MANUAIS DE EDUCAÇÃO CIENTÍFICA
E TECNOLÓGICA

ELLEN RICHMAN

MANUAL DE INTRODUÇÃO AOS COMPUTADORES

PUBLICAÇÕES DOM QUIXOTE / CÍRCULO DE LEITORES

MANUAL
DE INTRODUÇÃO
AOS COMPUTADORES

MANUAL DE INTRODUÇÃO AOS COMPUTADORES

Uma introdução à literacia computacional

Tradução de João Pinto Ferreira
Revisão técnica e adaptação do Dr. João Correia de Freitas

Ellen Richman

Publicações Dom Quixote/Círculo de Leitores
LISBOA
1990

Publicações Dom Quixote, Lda.
Rua Luciano Cordeiro, 116, 2.º
1098 Lisboa Codex — Portugal

Círculo de Leitores, Lda.
Rua Eng.º Paulo de Barros, 22
1599 Lisboa Codex — Portugal

Título original: *Spotlight on Computer Literacy*

Capa de Vítor Costa

Depósito legal n.º 40301/90
Fotocomposição: Textype - Artes Gráficas, Lda.
Impressão e acabamento: Gráfica Manuel Barbosa & Filhos, Lda.

Distribuição no mercado livreiro:
Digilivro — Rua Ilha do Pico, 3-B, Pontinha, Lisboa
Movilivro — Rua Gomes Leal, 93, Porto

ISBN: 972-20-0063-2 (Publicações Dom Quixote)

ÍNDICE

Bem-vindos ao mundo dos computadores

Provavelmente há muito pouco tempo você esteve em contacto com um computador: se fez um telefonema, um computador ajudou ao estabelecimento da chamada; se se divertiu com um jogo electrónico, um computador calculou as pontuações e ajudou a controlar a acção; se viajou de avião, o respectivo comandante teve a ajuda de computadores ao pilotar o aparelho; e saiba que, até neste livro, as palavras que está a ler foram escritas com a ajuda de um computador.

O que é que sabe sobre computadores?

De seguida está uma lista de afirmações sobre computadores. Indique as que são verdadeiras e as que são falsas:

1. Os computadores são mais inteligentes que os homens.
2. Os computadores têm cérebros.
3. Alguns computadores têm sentimentos.
4. Os computadores podem resolver qualquer problema.
5. Para se poder usar um computador é preciso saber-se muito sobre ciência e matemática.

Se considerou todas as afirmações falsas, acertou. Os computadores *não são* mais inteligentes do que os homens; *não têm* cérebros *nem* sentimentos; *não podem* resolver qualquer problema; e *não são* necessários muitos conhecimentos de ciência ou de matemática para se poder usar um computador.

Então, o que é um computador? O que pode fazer? É uma máquina que manuseia grandes quantidades de informação, que trabalha a velocidades espantosas e que está preparada para executar as seguintes quatro tarefas:

1. **Receber informação.** A informação é fornecida ao computador sob a forma de um conjunto de dados ou de valores, ou como um conjunto de instruções que lhe indicam o que ele tem a fazer.
2. **Armazenar informação.** O computador tem um dispositivo, a memória, que retém a informação enquanto se desejar.
3. **Processar informação.** Isto significa que o computador pode fazer algo com a informação: desde uma simples adição até comparar e ordenar nomes.
4. **Fornecer informação processada.** Em consequência das instruções que lhe foram dadas o computador fornece os resultados do processamento.

Os computadores são projectados para seguirem instruções de um ser humano e apenas resolvem os problemas que os homens lhes mandam resolver. Se se não souber solucionar um problema, um computador também o não pode fazer. Assim, para se indicar a um computador o que ele tem de executar, há que conhecer o problema que se quer resolver e ter um plano para a sua resolução.

Como os computadores não podem fazer nada sem instruções de um ser humano, o que os torna tão especiais é o facto de conseguirem executar algumas tarefas melhor que os seres humanos: têm uma maior velocidade de cálculo, são mais exactos, podem armazenar grandes quantidades de informação e não «esquecem» o que têm armazenado. São estas características que os tornam óptimas ferramentas auxiliares na resolução de problemas.

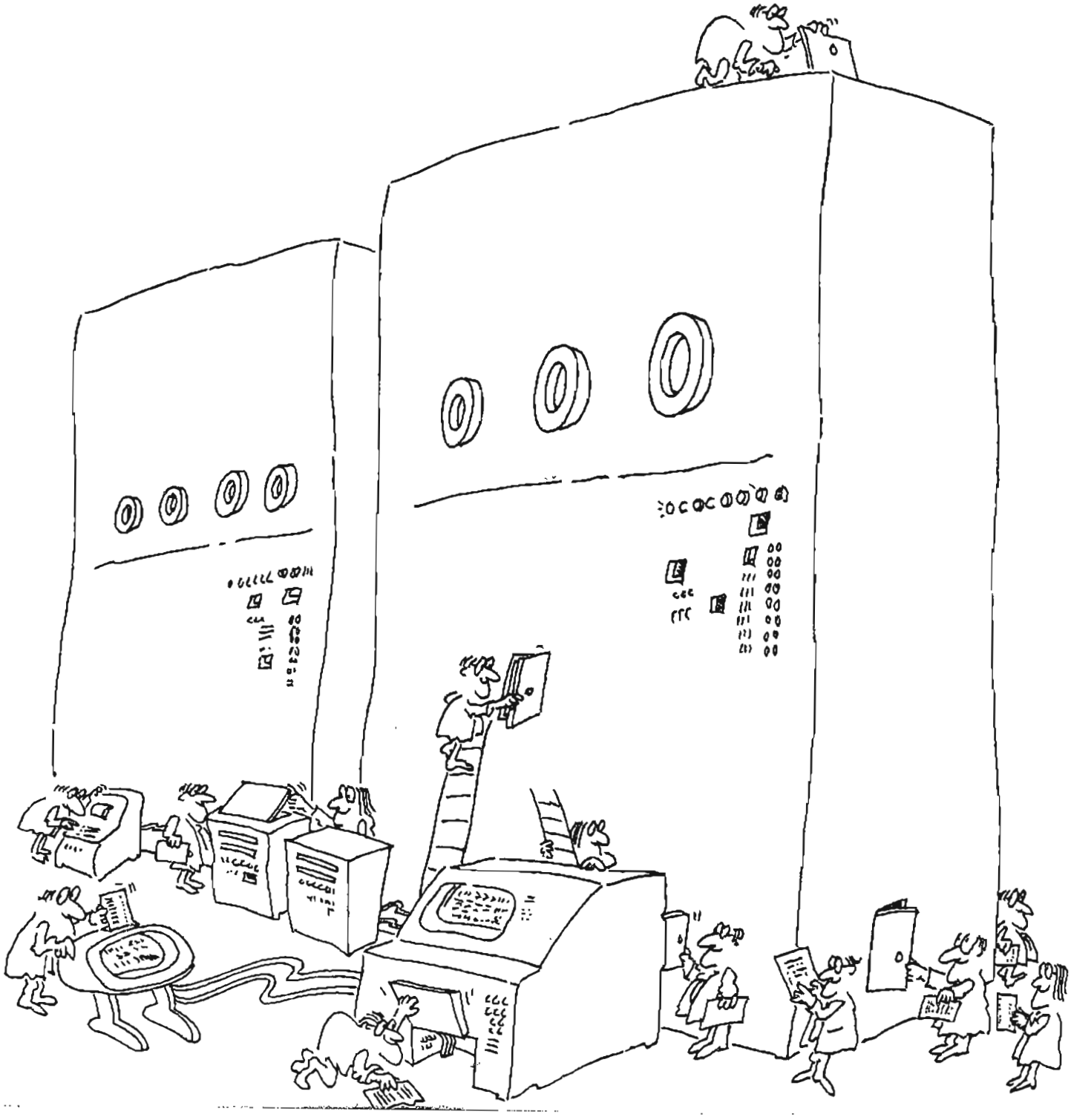
No mundo actual os computadores tornaram-se instrumentos importantes e valiosos. E porque eles nos afectam de tantos modos é importante estar informado sobre como são usados e como trabalham.

O objectivo deste livro é iniciar o leitor num caminho que o leve a tornar-se «instruído» em computadores. E o que significa isto? Considera-se alguém «instruído» se foi educado ou se tem conhecimentos ou experiência em determinadas áreas. Portanto, para se ser instruído em computadores será preciso entender-se de computadores e ter-se alguma experiência na sua utilização.

Neste livro o leitor poder-se-á instruir em computadores. Na primeira parte estudará os diferentes tipos de computadores, as várias partes que os constituem e também o que os faz funcionar. Na segunda parte poderá apreciar algumas das invenções históricas que conduziram ao desenvolvimento dos computadores actuais; informar-se-á como os computadores são utilizados na indústria ou nas empresas, na administração pública, nas escolas e em casa e verá também que espécie de oportunidades de carreira são criadas pelos computadores. Finalmente, na terceira parte, aprenderá a escrever programas para um computador.

Quando tiver abrangido estas três áreas principais — como funcionam os computadores, como é que os computadores afectam a nossa vida e como se programa um computador — terá dado um grande passo em frente para se tornar «instruído» em computadores. Bem-vindo ao mundo dos computadores!

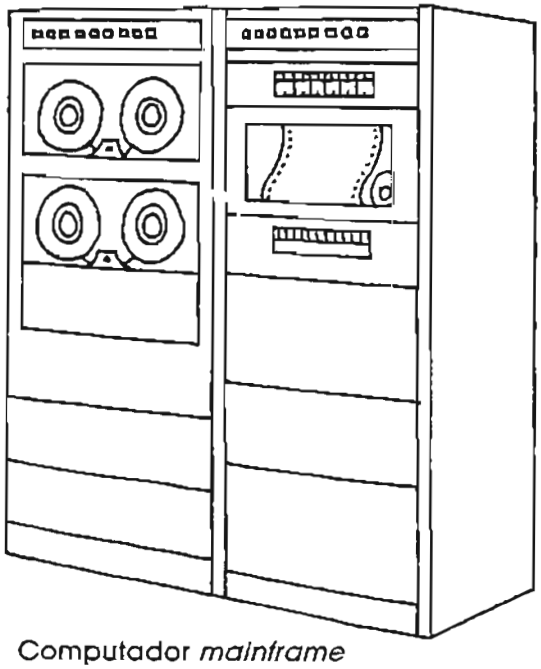
UNIDADE 1
COMO FUNCIONAM OS COMPUTADORES



Capítulo 1

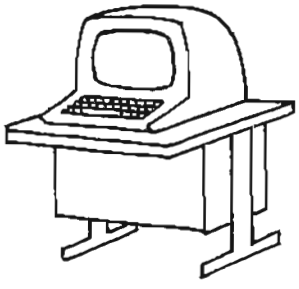
SISTEMAS DE COMPUTADORES

Há computadores de muitas formas e tamanhos, que vão desde os grandes sistemas computacionais, chamados computadores *mainframe*, até aos pequenos minicomputadores e aos ainda mais pequenos microcomputadores.



Computador *mainframe*

Minicomputador



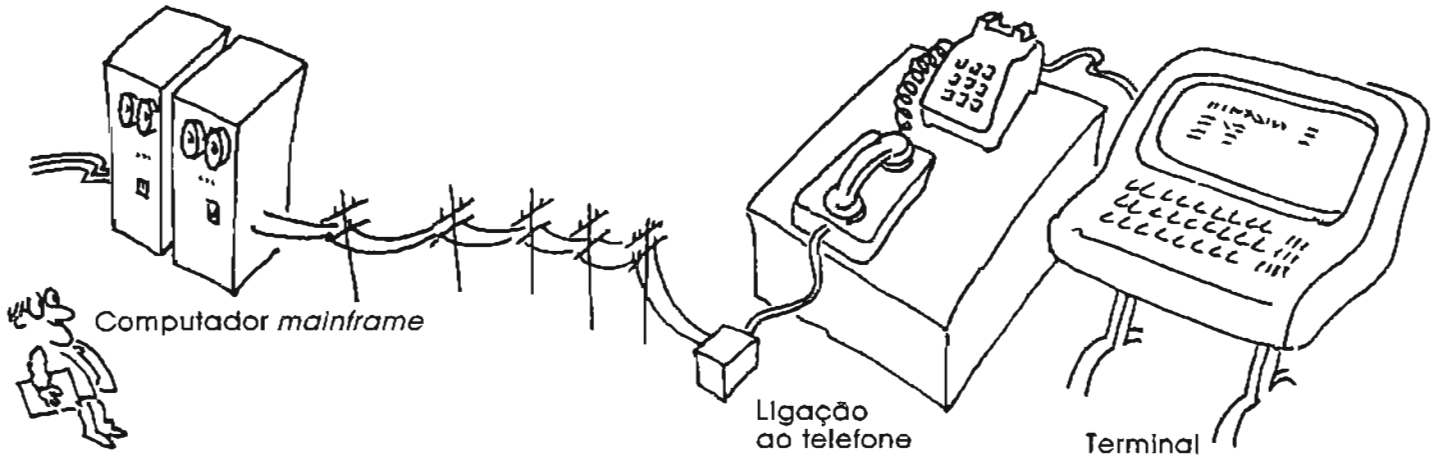
Microcomputador

Por vezes o equipamento ocupa diversas salas.

Os computadores *mainframe* são muito grandes e muitas vezes ocupam tanto espaço como uma sala de aula. Armazenam enormes quantidades de informação e custam dezenas de milhar ou mesmo centenas de milhar de contos. São utilizados pelas grandes empresas, pela administração pública e pelas universidades, que assim podem executar os seus trabalhos mais fácil e eficientemente. Por exemplo, uma companhia de telefones utilizará um computador deste tipo para um registo completo de todas as chamadas telefónicas interurbanas feitas pelos seus assinantes; a Marinha empregá-lo-á para registar os movimentos e a localização de todos os seus barcos e submarinos; uma universidade, por sua vez, utilizá-lo-á para ter um registo dos seus alunos e das respectivas notas.

Terminal significa «ponto final». Um terminal é o «ponto final» de um sistema de computadores.

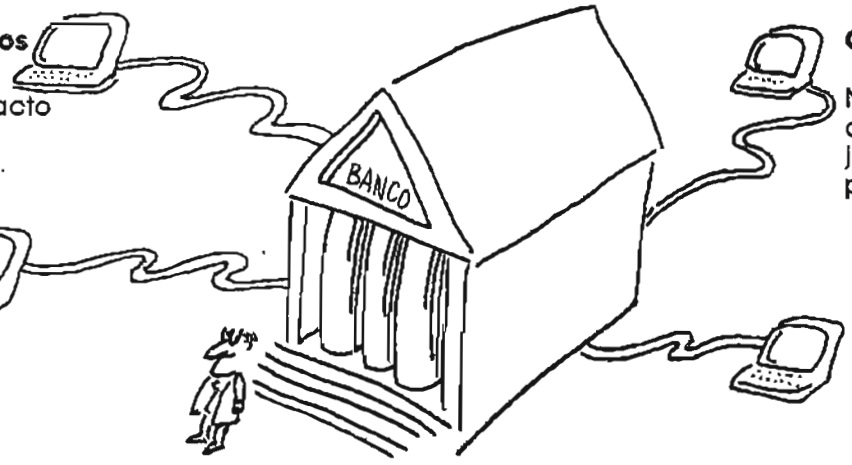
Os grandes sistemas podem ter, ligados a si, muitos postos ou terminais que mais parecem pequenos computadores, embora verdadeiramente não o sejam. Os terminais são dispositivos que enviam informação para o computador e recebem a que este lhes envia. Alguns deles localizam-se na mesma sala ou no mesmo edifício que o computador, estando a ele ligados por cabos; outros, por sua vez, localizam-se noutros edifícios ou noutras cidades, «comunicando» com o *mainframe* através de fios telefónicos.



Como este tipo de computador é muito grande e pode armazenar muita informação, é-lhe possível fazer, ao mesmo tempo, uma série de trabalhos diferentes: várias pessoas podem utilizar simultaneamente e para finalidades diversas os seus vários terminais. O armazém central de uma grande empresa em Nova Iorque, disporá, por exemplo, de um sistema de computadores com terminais colocados em todos os armazéns regionais e, num determinado momento, pode dar-se uma situação como a representada na figura:

Os **minicomputadores** são muito mais pequenos que os *mainframes.* Ocupam pouco espaço numa sala e, alguns deles até podem ser montados sobre uma mesa. O seu preço varia bastante. Têm capacidade para armazenar grandes quantidades de informação, ainda que não tanta como os *mainframes,* e tal como estes podem executar, ao mesmo tempo, mais do que uma tarefa. Os seus terminais, em número menor que os dos *mainframes* visto a sua capacidade não ser tão grande, situam-se na mesma sala ou no mesmo edifício do minicomputador. São utilizados não só por pequenas e médias empresas como também por grandes empresas e por escolas.

Realmente nalgumas grandes empresas, para além dos grandes sistemas trabalharem a informação respeitante ao conjunto da empresa, empregam-se minicomputadores para os trabalhos específicos de cada um dos seus departamentos. Os grandes bancos, por exemplo, têm vários departamentos (contas de depósitos à ordem, depósitos de poupança, empréstimos, operações internacionais e outros), cada um dos quais trabalha um conjunto de informações específicas que não diz respeito a outros departamentos: para trabalhar estas informações, cada departamento pode ter o seu próprio minicomputador, evitando recorrer à utilização do computador central.

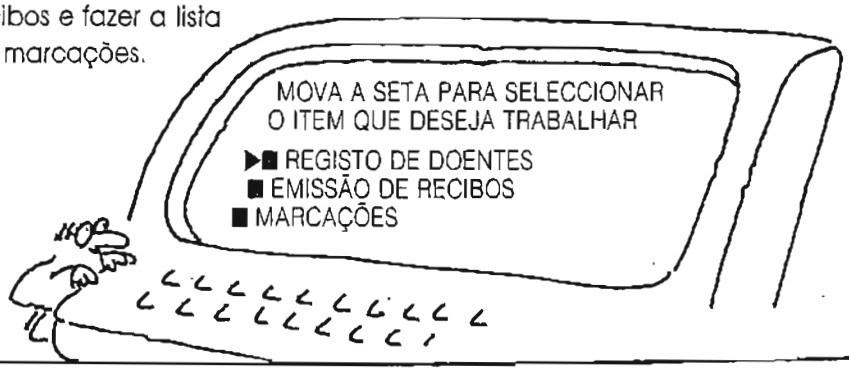


É possível, através de ligações telefónicas, ligar alguns tipos de microcomputadores a outros microcomputadores, a minicomputadores e a *mainframes*. Nesta situação o micromputador pode funcionar como um terminal

Os microcomputadores são mais pequenos que os minicomputadores. São conhecidos por computadores «pessoais» ou «domésticos», porque muitas pessoas os adquirem para essas utilizações. São leves e pequenos para se poderem transportar de um lado para o outro e cabem em cima de uma mesa ou de uma secretária. O seu preço é inferior a mil contos e muitos deles não ultrapassam algumas dezenas de contos. Um microcomputador não pode armazenar tanta informação como um *mainframe* ou um minicomputador e não permite também a ligação de terminais.

Recorde que os grandes sistemas podem executar várias tarefas ao mesmo tempo.

A maioria dos microcomputadores está preparada para realizar apenas um trabalho de cada vez. São utilizados por grandes e pequenas empresas e também por escritórios. As grandes empresas, embora processem os seus maiores trabalhos em *mainframes* e minicomputadores, também empregam microcomputadores em trabalhos de pequeno porte, de natureza individual. Muitas pequenas empresas que não se podem permitir dispor de um grande sistema computacional verificam que um microcomputador se ajusta às suas necessidades pois podem utilizar-se em diferentes tipos de trabalhos, embora não os executem em simultâneo. Num consultório médico, por exemplo, o microcomputador pode proceder ao registo dos doentes, imprimir os respectivos recibos e fazer a lista de marcações.



Os **programas** são conjuntos de instruções dadas ao computador que lhe dizem o que deve fazer.

Muitas escolas já têm microcomputadores nas salas de aula, o que permite aos alunos utilizarem-nos como apoio em línguas, nas Ciências, na Matemática, nos Estudos Sociais e noutras disciplinas; os professores, por sua vez, podem empregá-los para registar os progressos dos seus alunos ou para outros tipos de registos. Nas nossas casas os microcomputadores podem ser úteis para apoio às finanças domésticas, para elaboração de jogos ou simplesmente porque há quem goste de escrever programas como passatempo.

tente responder

1. Que tipo de computador (*mainframe,* minicomputador ou microcomputador) utilizaria para cada uma das seguintes tarefas?

   a. Uma grande empresa distribuidora de energia eléctrica deseja utilizar um computador para registar a electricidade consumida todos os meses pelos seus milhares de clientes.

   b. Uma pessoa utiliza-o para registar receitas culinárias, datas importantes e controlar os extractos das suas contas bancárias.

   c. No Hawai um corrector usa um terminal para comprar e vender obrigações na Bolsa de Valores de Nova Iorque.

   d. Na sede da NASA utiliza-se para o lançamento e acompanhamento das órbitas das centenas de satélites existentes em volta de Terra.

   e. Numa grande clínica médica um guarda-livros usa um terminal para introduzir as contas dos doentes no respectivo computador.

A **NASA** (Administração Nacional de Aeronáutica e do Espaço) é um departamento da administração dos Estados Unidos.

2. Segue-se um resumo dos três principais tipos de sistemas de computador. Diga o nome de cada sistema.

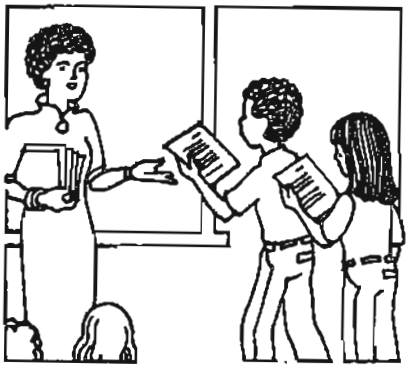
	Dimensão	Custo Aproximado	Características
a.	Chega a ocupar toda uma sala.	Muitos milhares de contos.	Executam muitas tarefas ao mesmo tempo; podem ter muitos terminais; usados nas grandes empresas, departamentos governamentais e universidades.
b.	Coloca-se no pavimento ou numa mesa grande.	De 1000 a 10 000 contos.	Executam várias tarefas ao mesmo tempo; podem ter diversos terminais; utilizados nas pequenas e médias empresas e em departamentos das grandes empresas.
c.	Portátil. Pode colocar-se numa secretária.	Menos de 1000 contos até algumas dezenas de contos.	Apenas executa uma tarefa de cada vez; usado em pequenas empresas, em trabalhos isolados nas grandes empresas, em casa e nas salas de aula.

Capítulo 2

COMO FUNCIONA UM COMPUTADOR

Quando se dão instruções a um computador para fazer um determinado trabalho, ele vai executá-las de uma forma muito especial: começa por receber a informação, armazena-a até poder ser utilizada, processa-a e finalmente fornece a informação devidamente trabalhada. Este método de trabalho é afinal idêntico ao que muitas pessoas empregam todos os dias. Vejamos um exemplo:

Numa escola os alunos da senhora Branca fazem muitos trabalhos escritos. Ela recebe-os, coloca-os numa pasta que tem na secretária, no fim do dia corrige-os e no dia seguinte devolve-os aos alunos. Esta sequência de operações é idêntica à de um computador.



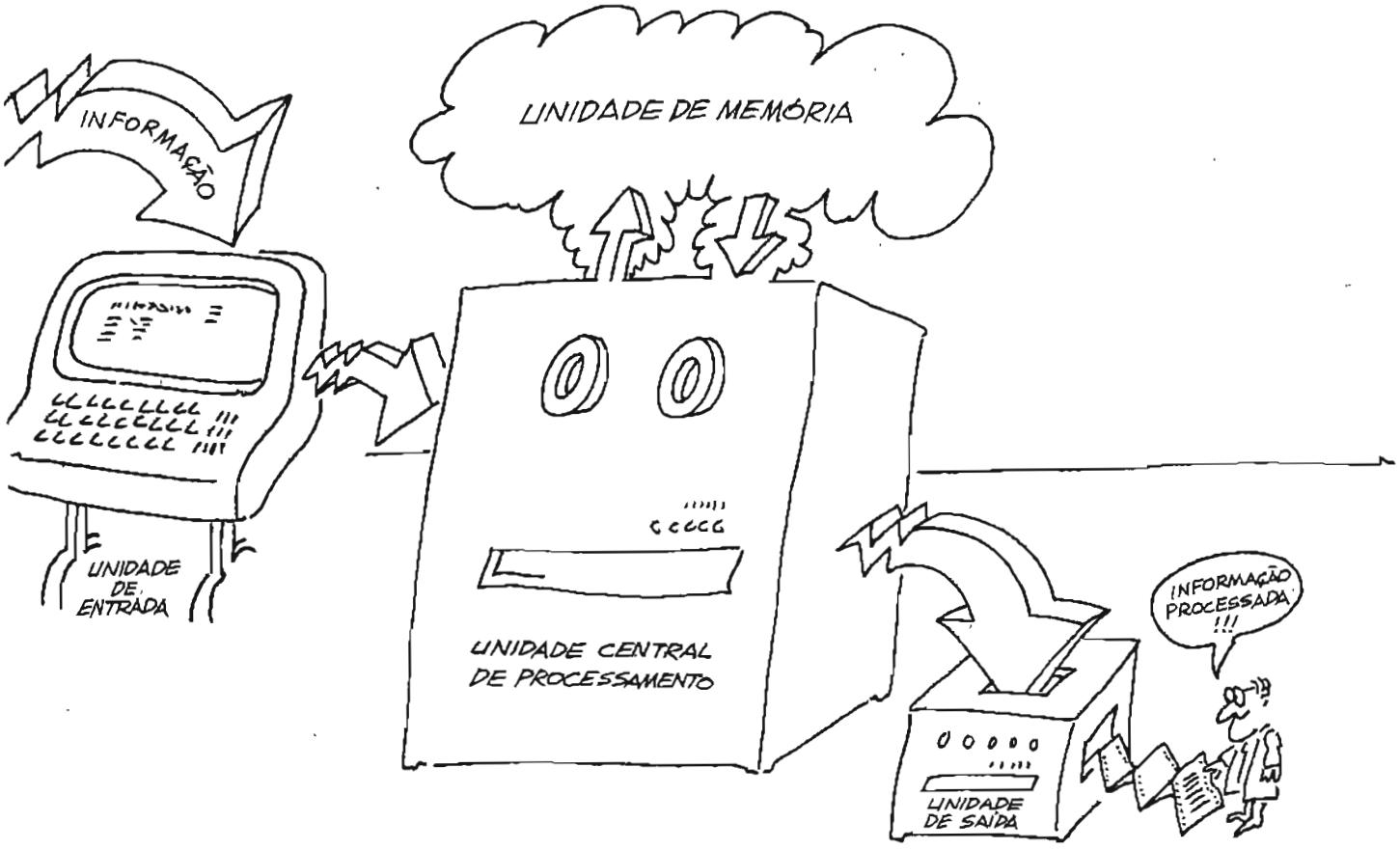
1. Recebe a informação.

2. Armazena a informação.

3. Processa a informação.

4. Fornece a informação processada.

A execução de cada uma destas tarefas é realizada numa determinada parte do computador. Todos eles, desde os grandes sistemas que custam centenas de milhar de contos aos pequenos microcomputadores de algumas de contos, são constituídos pelas seguintes quatro partes ou unidades: **unidade de entrada, memória, unidade central de processamento** e **unidade de saída**. Ver-se-á em seguida como estas quatro partes estão relacionadas.



A **unidade de entrada** recebe a informação.
A **unidade de memória** armazena a informação.
A **unidade central de processamento** (CPU ou UCP) processa a informação.
A **unidade de saída** fornece a informação processada.

Processar a informação significa «fazer alguma coisa com» a informação.

tente responder

1. Diga qual o trabalho executado pelo computador (recepção, armazenamento, processamento e saída de dados) e a parte do computador onde é executada (unidade de entrada, memória, unidade central de processamento e unidade de saída) em cada uma das seguintes actividades executadas na aula da senhora Branca:

   a. Os alunos entregam os seus trabalhos à professora.
   b. A senhora Branca guarda os trabalhos numa pasta.
   c. A senhora Branca corrige os trabalhos e classifica-os.
   d. A senhora Branca devolve os trabalhos corrigidos e classificados aos alunos.

2. Indicam-se em seguida algumas tarefas que se executam na *Pizzaria* do Sá. Faça-as corresponder às várias tarefas executadas por um computador e indique a parte do computador onde são realizadas:

   a. O Sá recebe um fornecimento de géneros para fazer *pizza*.

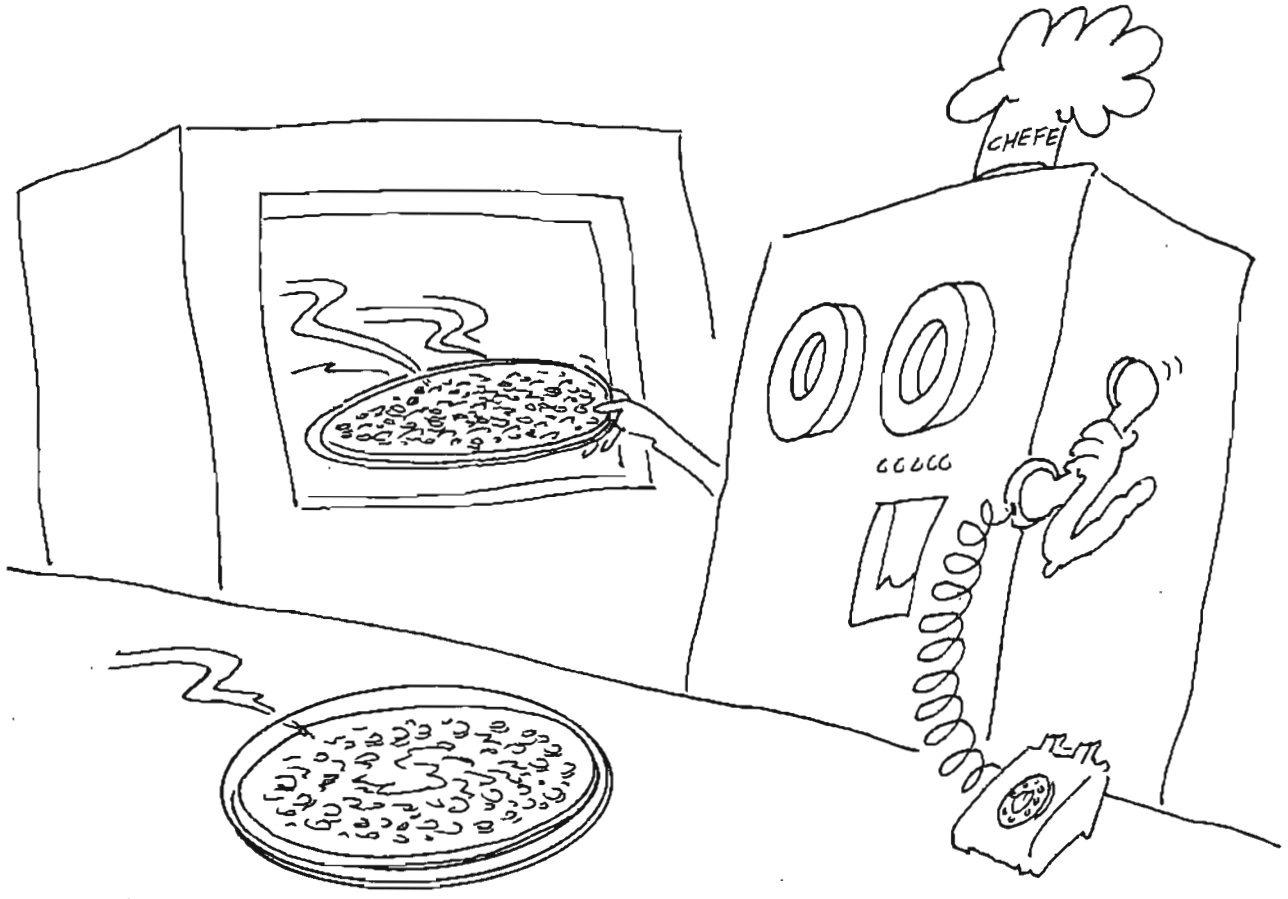
b. Na sua dispensa guarda a massa de farinha, o molho da *pizza,* o queijo, os condimentos e o *peperoni.*

c. Modela a massa numa forma circular, cobre-a com o molho, o queijo, os condimentos e o *peperoni* e coze a *pizza.*

d. O Sá serve a *pizza.*

Imagine por um momento!

Se um computador pudesse substituir o Sá na sua *Pizzaria,* seria a *unidade de entrada* que receberia os fornecimentos de géneros, a *memória* que guardaria os vários ingredientes e a *unidade central de processamento* que os misturaria nas devidas proporções e procederia à cozedura da *pizza;* à *unidade de saída* caberia a tarefa de servir a *pizza* aos clientes.

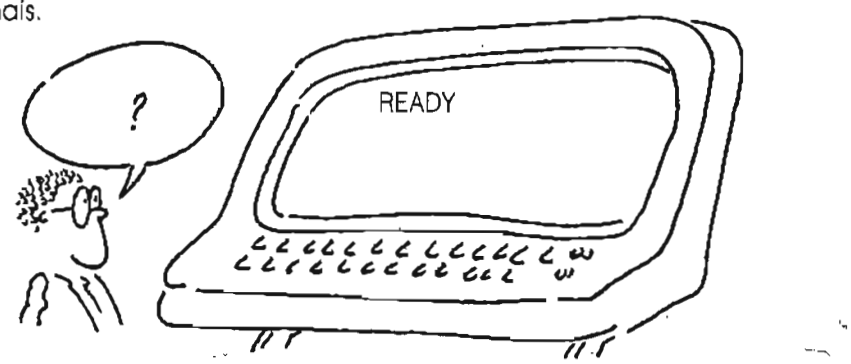


Capítulo 3

A COMUNICAÇÃO COM UM COMPUTADOR

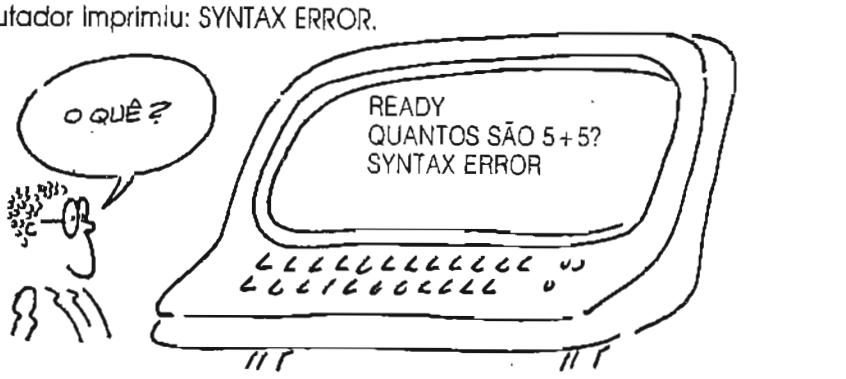
Os computadores destinam-se a executar diversas tarefas, das mais complexas como o controlo de uma nave espacial em vôo às mais simples como a adição de dois números. Seja qual for a tarefa que tenha de executar, o computador tem de receber um conjunto de instruções numa linguagem que ele «compreenda». Vejamos como se dão instruções aos computadores.

O Jorge pegou numa cadeira e sentou-se em frente do microcomputador. Ouvira dizer que os computadores faziam coisas maravilhosas e estava ansioso por vê-las com os seus olhos. Ligou o computador e esperou. No *écran* apareceu a palavra READY (que significa que o computador está «pronto»), mas nada mais.



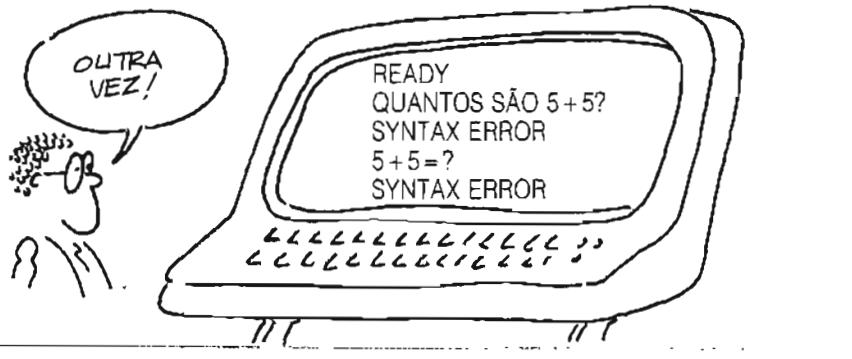
Nem todos os computadores quando são ligados fazem aparecer no *écran* a palavra READY: outros imprimem mensagens como *O.K.*, o nome do computador ou um símbolo especial.

O Jorge resolveu então escrever a seguinte pergunta: QUANTOS SÃO 5+5?
O computador imprimiu: SYNTAX ERROR.



SYNTAX (em português SINTAXE) relaciona-se com a estrutura de uma linguagem e as respectivas regras. Se escrever algo que o computador não entenda ele imprimirá a mensagem SYNTAX ERROR (erro de sintaxe).

Perplexo, o Jorge tentou fazer algo diferente. Cuidadosamente teclou: 5+5=?
E de novo o computador respondeu: SYNTAX ERROR.



Os problemas que o Jorge estava a ter na comunicação com o seu computador eram devidos ao facto de ele não compreender como os computadores trabalham. O computador não tem um «cérebro» para entender o que se lhe pergunta, nem pode executar qualquer trabalho sem instruções. Um computador é como um automóvel que sem gasolina não consegue mover-se: sem instruções, um computador também não consegue executar qualquer tarefa.

Todos os computadores — *mainframes*, minicomputadores e microcomputadores — têm de ter instruções ou **programas** para executarem as suas tarefas.

Chama-se **programa** a um conjunto de instruções que indicam ao computador o que ele tem de fazer. Assim há vários tipos de programas: os que «dizem» ao computador para calcular o saldo de uma conta bancária, os que procedem à emissão da folha de vencimentos de uma empresa, os que servem para efectuar um plano de vôo até Paris, os que perguntam a um aluno o que ele sabe, por exemplo, acerca de pronomes ou até os que nos fazem disputar um jogo de monstros do espaço. Todos os programas devem ser escritos numa linguagem que o computador possa entender.

Há muitas linguagens de computador, cada uma delas destinada a uma finalidade específica. As linguagens mais vulgarmente usadas nos computadores são as seguintes:

BASIC	Beginners All-purpose Symbolic Instruction Code
PASCAL	derivado do nome de Blaise Pascal (1623-1662)
FORTRAN	Formula Translator
COBOL	Common Business Oriented Language
Logo	palavra grega que significa «pensamento»

O BASIC é a linguagem mais vulgarizada nos microcomputadores existentes nas escolas. Esta linguagem emprega várias palavras correntes da língua inglesa, cuja utilização no entanto deve ser feita de uma maneira muito exacta. Por exemplo, para mandar executar a operação 5+5, na linguagem BASIC, o Jorge teria de escrever:

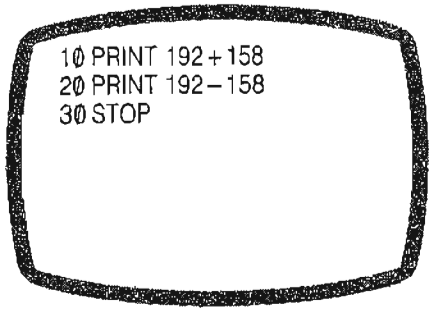
PRINT é um termo da linguagem BASIC.

```
PRINT 5+5
```

Então o computador imprimiria a resposta: 10.

Os programas consistem numa série de instruções que o computador vai ler e executar uma a uma. Vejamos um programa simples, escrito em BASIC, que indica ao computador para calcular a soma e a diferença de dois números. Uma vez o programa escrito no teclado do microcomputador, aparece no *écran* com o seguinte aspecto:

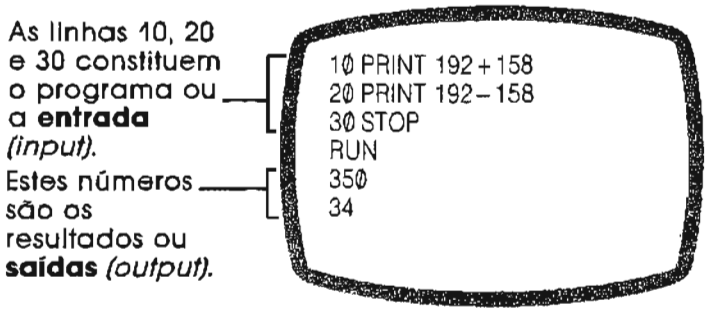
Na maior parte dos computadores o zero distingue-se da letra «O» por aparecer cortado com uma barra sob a forma «Ø».



RUN é uma palavra da linguagem BASIC que significa «processe as instruções» ou, por outras palavras, «execute o programa».

Cada linha do programa, depois de escrita no teclado, vai ser armazenada na memória do computador. Contudo o programa só se processa quando se escreve o comando RUN. Uma vez isto feito, o computador segue as várias

instruções do programa: «lê» a primeira linha (linha 10) e vai calcular e imprimir a soma de 192 com 158 que é 350; depois «lê» a segunda linha (linha 20) e calcula e imprime a resposta à operação 192−158 que é 34; finalmente «lê» a linha seguinte (linha 30) que lhe indica que deve parar: não há mais instruções.



Sempre que se digita e faz correr este programa de computador, são envolvidas no processo as quatro partes que o constituem:

Unidade de Entrada: o teclado utilizado para introduzir o programa no computador.

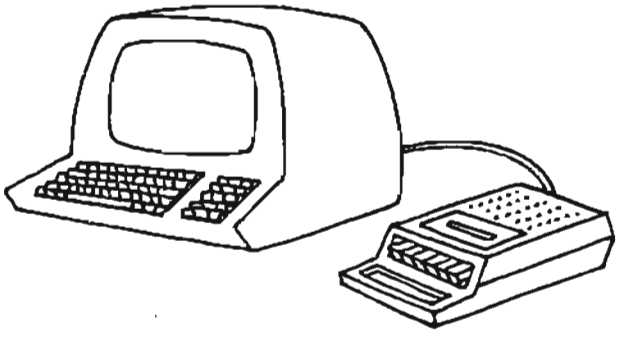
Memória: é aqui que o programa é armazenado.

Unidade Central de Processamento: onde são executadas as instruções do programa.

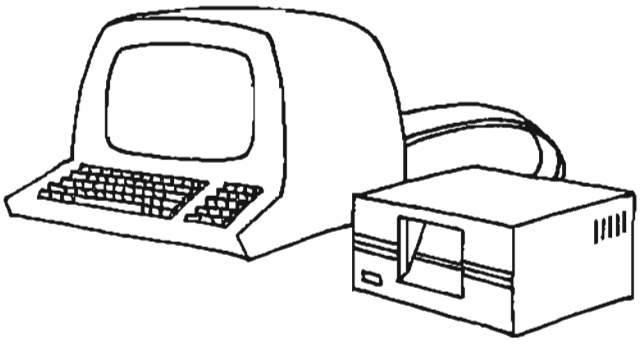
Unidade de Saída: o *écran* onde são apresentados os resultados do programa.

Existem várias formas de introduzir um programa num microcomputador: uma delas é escrever o programa no teclado respectivo; outra forma é usar um programa já escrito (por si ou por alguém) que esteja gravado numa fita de *cassette* ou num disco (disquete) e que são introduzidos no computador através de um gravador de cassettes ou de um leitor de discos. Estes dispositivos permitem usar o mesmo programa várias vezes sem ter de o escrever no teclado sempre que se quer usá-lo. Chama-se **hardware** a todo o equipamento do computador como o teclado, o *écran*, o gravador de *cassettes*, o leitor de discos e outros dispositivos a ele ligados.

No capítulo 4 há mais informações sobre fitas de gravação e discos



Gravador de *cassettes*



Leitor de discos

A aquisição de *software* deve ter em conta a marca e as especificações do computador em que vai ser utilizado, não podendo, geralmente, ser usado noutras marcas.

Aos programas dos computadores chama-se **software**. O preço de aquisição do *software* varia entre algumas centenas de escudos (caso dos jogos de computador) e largas centenas de contos (caso de programas complexos para a execução de tarefas múltiplas). Uma escola pode ter *software* próprio para os seus computadores, parte do qual pode ter sido adquirida e parte ter sido escrita por alunos ou professores. As duas soluções (adquirir *software* ou escrever os seus próprios programas) têm vantagens: com a primeira ganha-se muito tempo e permite-se a utilização de um computador mesmo a quem não saiba programar; com a segunda solução é possível indicar ao computador *exactamente* o que se deseja que ele execute.

1. Um computador não funciona sem um conjunto de instruções. Como se chama tal conjunto de instruções?
2. Indique o nome da linguagem de programação utilizada na maior parte dos microcomputadores das escolas.
3. O que é o *hardware* de um computador?
4. O que é o *software* de um computador?
5. Indique uma vantagem que tenha em comprar *software* para o seu computador.
6. Indique uma vantagem que tenha em escrever os seus próprios programas.
7. Eis um exemplo de um programa escrito em linguagem BASIC.

```
10 PRINT 6+6
20 PRINT 9-7
30 STOP
```

   a. Que instruções são dadas ao computador para ele executar?
   b. Que resultados (saídas) indicará o computador depois de obedecer às instruções?

Capítulo 4

ENTRADAS *(INPUT)* E SAÍDAS *(OUTPUT)*

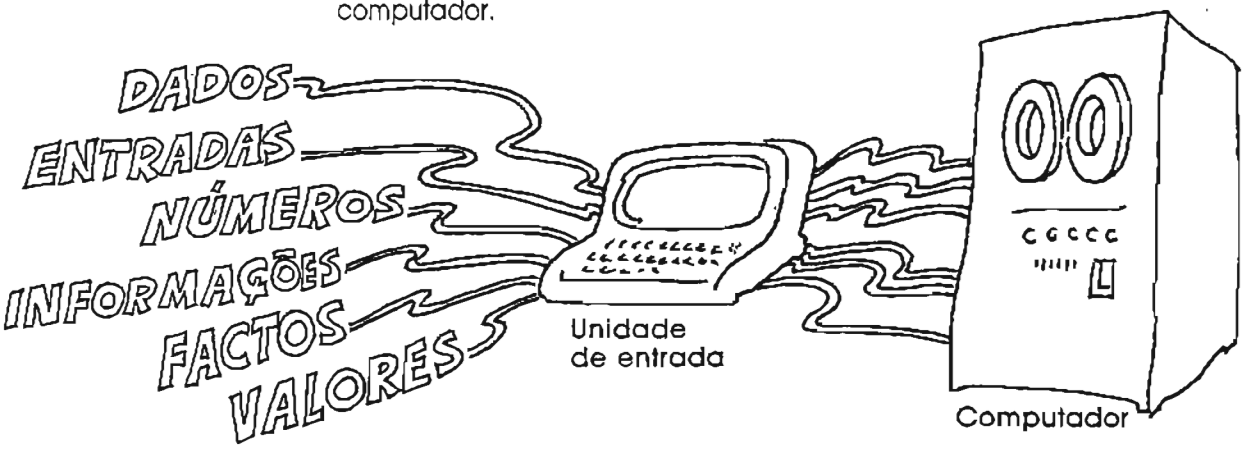
Vamos ver em seguida alguns dos diferentes tipos de dispositivos utilizados como **unidades de entrada** e como **unidades de saída**, que, como se deve recordar, são as partes de um sistema computacional que recebem informação e fornecem informação processada.

Unidades de Entrada

Se se partir a palavra *input* em sílabas e se se alterar a respectiva ordem, recordaremos imediatamente o papel de um dispositivo utilizado como unidade de entrada.

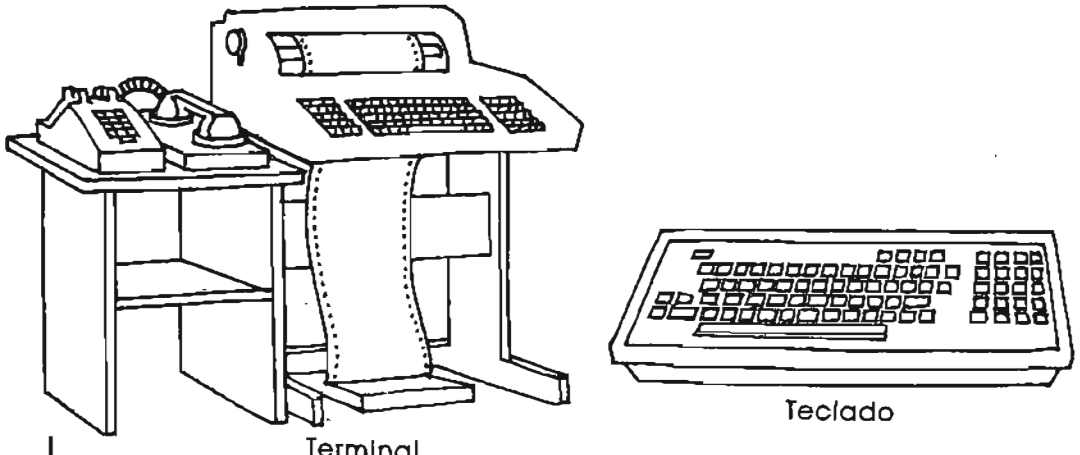
in-put → put in

Put in significa «introduzir», «pôr em», o que corresponde ao trabalho de uma unidade de entrada que é precisamente *introduzir* informação dentro do computador.



Como se introduzem as informações ou os dados num computador? Um processo é escrever os dados ou os programas no **teclado** do computador que é assim um dos tipos possíveis de unidades de entrada. Se o teclado estiver incorporado no **terminal** de um grande sistema então a unidade de entrada é o próprio terminal.

Os terminais por vezes parecem-se com microcomputadores.

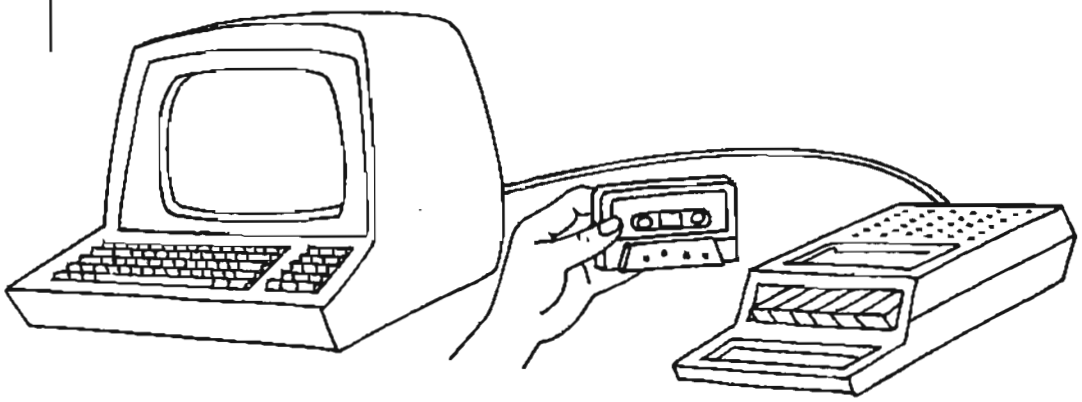


Geralmente os programas de computador são armazenados no exterior do computador em suportes apropriados. Este sistema permite a sua utilização repetidas vezes com a vantagem de demorar menos tempo a carregar no computador um programa longo gravado num desses suportes do que se se tivesse de o escrever totalmente no teclado sempre que se desejasse utilizá-lo.

Há muitos tipos de **unidades periféricas de entrada** que «carregam» programas guardados no exterior do computador. Nos microcomputadores funciona geralmente como tal o **gravador de cassettes**. A fita da cassette é uma banda de material magnético onde os programas são gravados sob a forma de impulsos magnéticos que o gravador entende ou «lê» à medida que a fita vai correndo, transferindo essa informação para o computador ao qual está ligado.

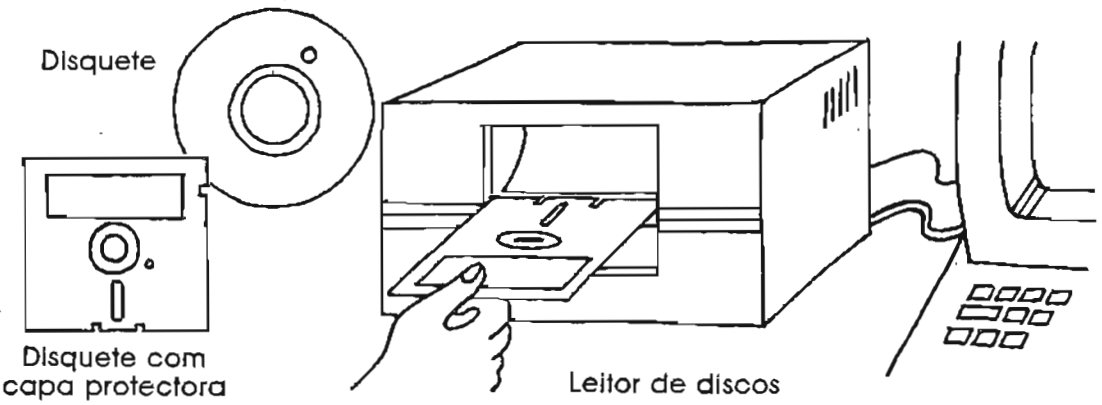
Periférico significa «relacionado com a parte mais exterior».

Se tiver um gravador desligue o cabo de ligação ao computador e ouça o programa gravado na fita. Os ruídos que ouve não soam nem a inglês, nem a BASIC, e muito menos a música.



Outra forma de suporte para gravação de programas destinados a microcomputadores ainda sob a forma de impulsos magnéticos são os **floppy disks** (discos flexíveis, ou simplesmente **disquetes**). Têm uma forma circular, parecendo-se com um pequeno disco, mas são flexíveis pois curvam-se facilmente (donde o termo *floppy* que significa «mole», «frouxo»). Uma disquete pode ser inutilizada por poeiras ou por uma simples impressão digital, pelo que se encontra permanentemente guardada num invólucro cartonado ou plastificado.

Existem **discos flexíveis** ou **disquetes** em diferentes dimensões, as mais vulgares das quais são as de 8 polegadas e de 5 ¼ polegadas de diâmetro.



As disquetes devem estar permanentemente encerradas na respectiva cobertura protectora evitando-se assim que se sujem e facilitando o seu manuseamento.

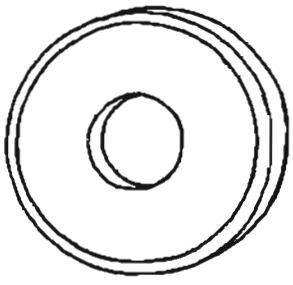
A disquete é introduzida num dispositivo de entrada chamado **leitor de discos** e que vai carregar o programa no computador. A disquete roda dentro da sua capa protectora da mesma forma que um vulgar disco roda num gira-discos. O leitor de discos dispõe de uma **cabeça de leitura/gravação** que desliza sobre a superfície magnética da disquete um pouco à maneira do braço de um gira-discos. Simplesmente, como a disquete não tem estrias, a cabeça de leitura/gravação não precisa de ter agulha, tendo, isso sim, umas

almofadas metálicas que procedem à leitura dos impulsos magnéticos registados e transmitem essa informação ao computador.

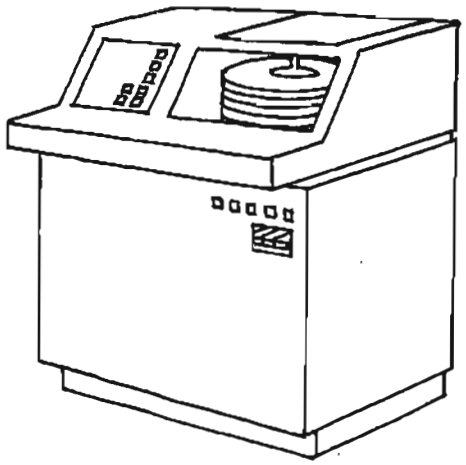
Uma disquete tem habitualmente maior capacidade de armazenagem que uma cassette e, para além disto, o leitor de discos é muito mais rápido a carregar a sua informação no computador que um gravador de cassettes. O grande inconveniente deste sistema é o seu preço que é muito mais caro do que um simples gravador.

Alguns microcomputadores usam também **discos rígidos.**

Como dispositivos de entrada dos minicomputadores e dos grandes sistemas usam-se geralmente as **unidades de discos rígidos.** Estes são maiores que as disquetes e, como o nome indica, são duros, não flexíveis. A sua superfície é magnética pelo que o processo de gravação da informação também é feito sob a forma de impulsos magnéticos. E também aqui é uma cabeça de leitura/gravação que entende ou «lê» a informação do disco e a envia ao computador.

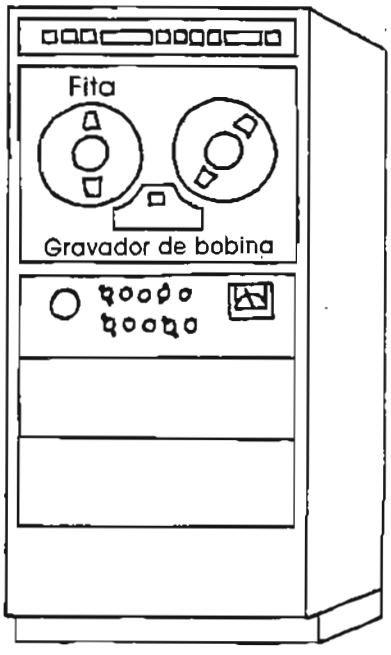


Disco rígido



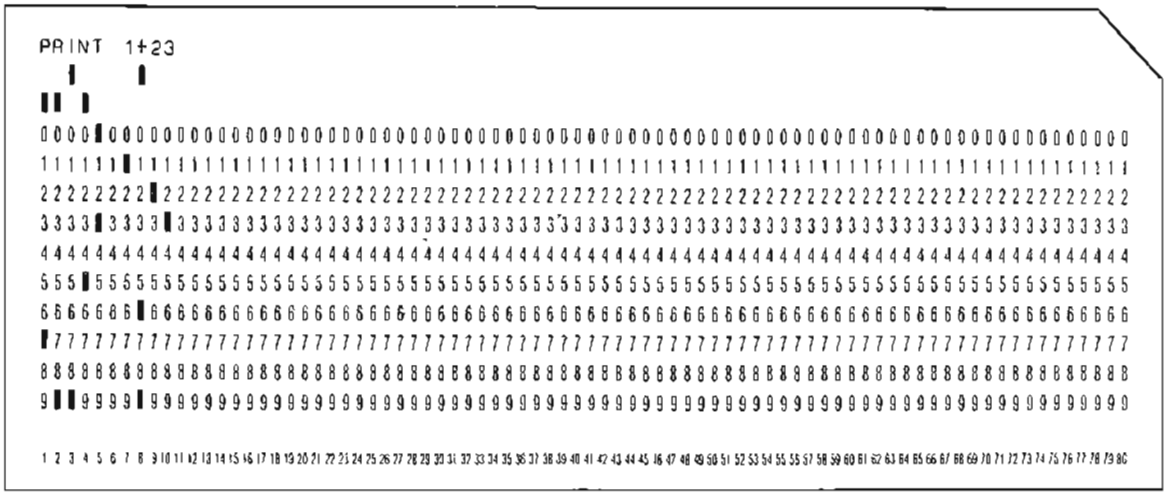
Unidade de discos rígidos

Alguns computadores *mainframe* têm unidades de entrada diferentes das anteriores: usam unidades de **fita magnética em bobinas.** Estas fitas são mais largas e mais compridas que as fitas das cassettes e os respectivos leitores carregam a informação mais rapidamente que os de *cassettes.*



Outro tipo de unidade periférica de entrada é o **leitor de cartões** que lê e ordena **cartões perfurados.** Os dados são perfurados em cartões especiais por alguém que trabalha com uma **máquina perfuradora,** utilizando-se um único cartão para cada instrução. Cada letra, número ou símbolo tem um determinado padrão de perfuração. No cartão da figura seguinte as perfurações representam a instrução PRINT 1+23.

Uma **máquina perfuradora** tem um teclado de máquina de escrever.



Este é um dos mais antigos métodos de introduzir informações.

Dados são a informação que vai ser processada pelo computador.

Este processo é por vezes conhecido como varrimento ou *scanning* óptico.

Os cartões perfurados são colocados no leitor respectivo ligado ao computador, que vai ler o padrão de furos existente e envia a informação respectiva para o computador. Os cartões perfurados usam-se dos microcomputadores aos grandes sistemas.

O leitor óptico é outro tipo de unidade de entrada; ele lê documentos marcados de uma forma especial e transfere os dados para o computador. Os leitores ópticos estão preparados para ler muitos documentos numa cadência muito rápida. Por exemplo, já alguma vez fez um teste em que tem de preencher pequenos círculos ou linhas numa folha de resposta? As respostas são pontuadas através da sua análise por um leitor óptico que compara o padrão das marcas feitas no papel com o padrão correcto armazenado na memória do computador.

Já alguma vez reparou que nos avisos de pagamento que recebe em sua casa ou nos cheques bancários existem dígitos especiais de «computador»? Estes dígitos, impressos com tinta magnética, representam geralmente o número de conta e, em certos avisos, a quantidade de dinheiro que está em débito. Há um tipo especial de dispositivo de entrada, o leitor MICR (de *Magnetic Ink Character Recognition*) ou seja **leitor de reconhecimento de caracteres em tinta magnética** — que pode «ler» a forma especial de cada dígito, transferindo os dados respectivos para o computador que regista o movimento da sua conta.

M. Garcia

No 701

Sept. 19 19 82

PAY TO THE ORDER OF Telephone Company $ 22.41

Twenty two and 41/100 DOLLARS

NBR
National Bank of Ridgeville

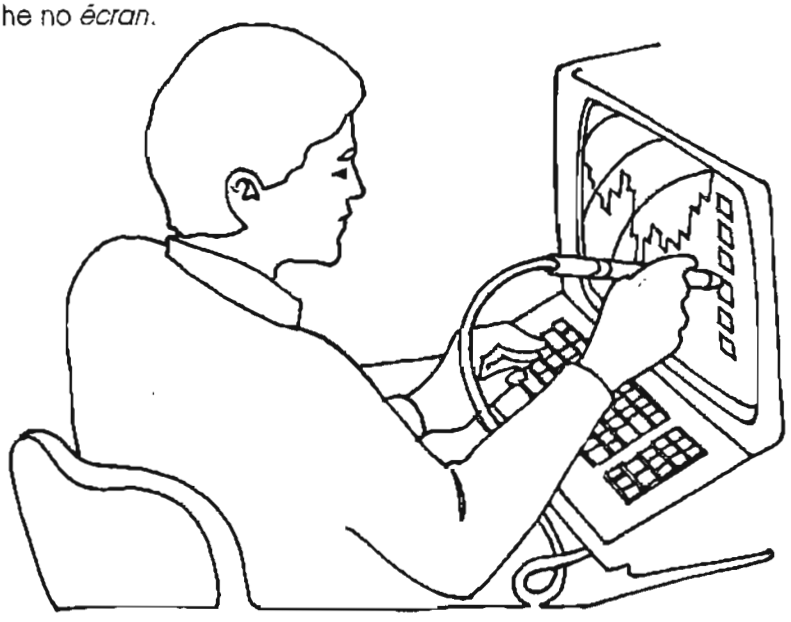
MEMO

M. Garcia

⑈03100531⑈ ⑈784 100 4⑈ 701

Caracteres magnéticos

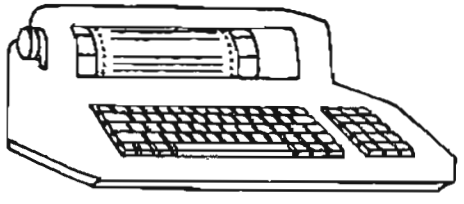
Outro tipo de dispositivo de entrada é a light pen ou caneta luminosa. Parece-se com uma caneta, está ligada ao computador por um cabo e é sensível à luz existente no *écran*, alimentando de informações o computador. Pode ser usada, por exemplo, para rectificar uma figura que o computador desenhe no *écran*.



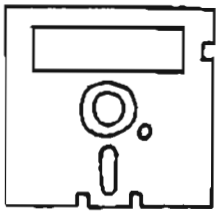
tente responder

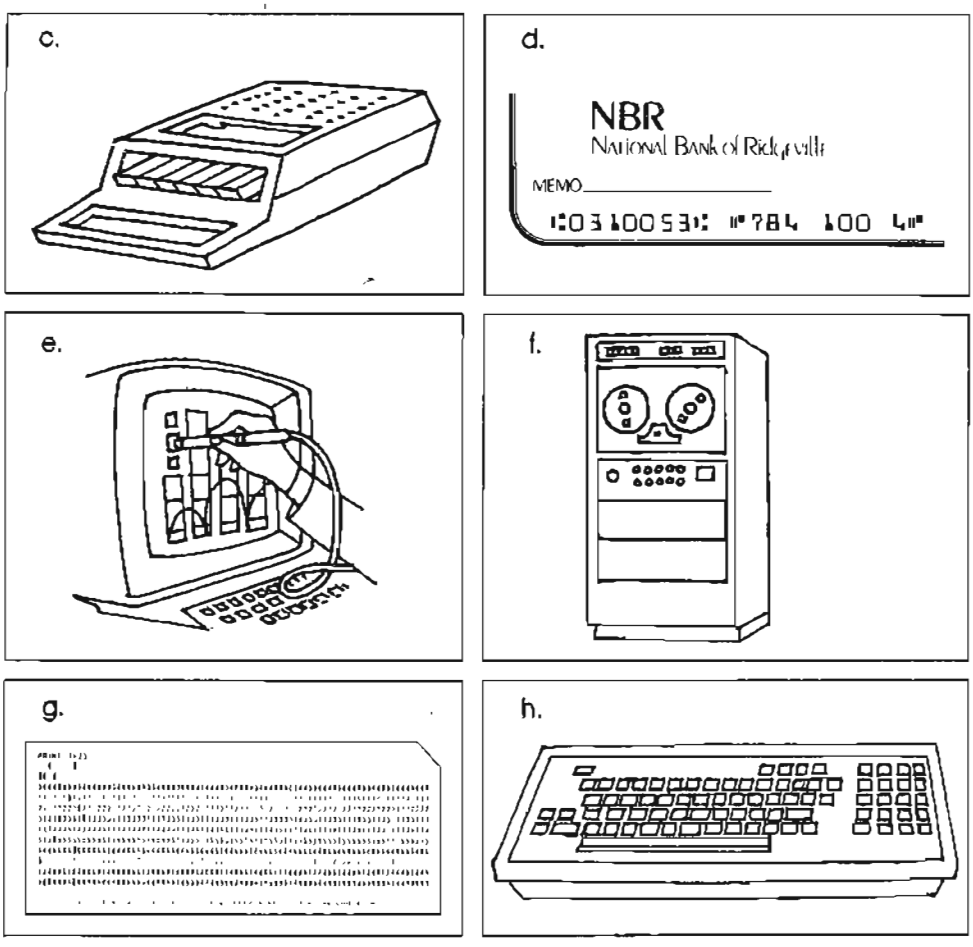
1. Qual é o papel de um dispositivo de entrada?
2. Diga o nome de cada um dos dispositivos representados. Uns são de entrada e outros de armazenamento.

a.



b.





Unidades de Saída

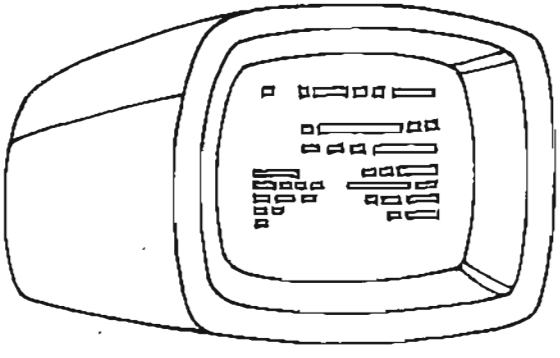
Poderemos recordar a função de uma unidade de saída da mesma forma que fizemos com a unidade de entrada, ou seja,

out-put → put out

o que significa «expelir, deitar para fora». Um dispositivo de saída *deita para fora* a informação que o computador processou.

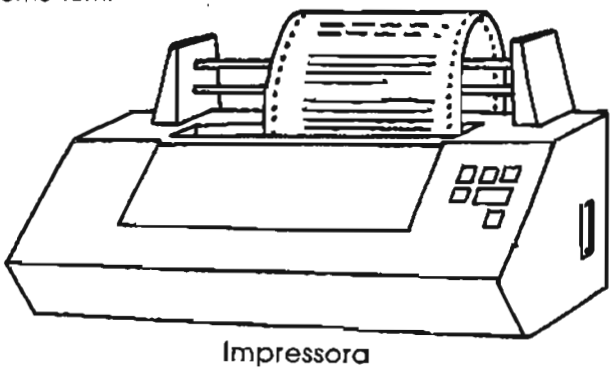
Há muitas formas de um computador fornecer a informação processada. O mais vulgar dos dispositivos de saída utilizados nos microcomputadores e nos terminais é o *écran vídeo*. Nos computadores domésticos este pode ser uma vulgar televisão ou então um monitor.

Um ***écran vídeo*** parece-se com uma televisão mas não capta o sinal dos canais de televisão. Também se lhe chama Tubo de Raios Catódicos (CRT ou TRC) ou monitor.



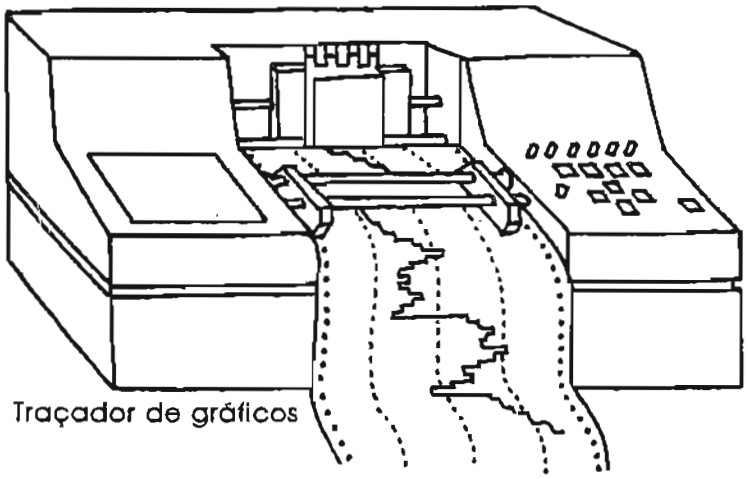
Écran vídeo ou
Tubo de raios catódicos
(CRT ou TRC)

A **impressora** é outra vulgar unidade periférica de saída. Em vez de ser apresentada num *écran*, a informação é enviada para a impressora que a imprime em papel. A velocidade das impressoras varia mas a maior parte delas imprime a informação mais rapidamente que a velocidade de leitura de uma pessoa. As saídas impressas, chamadas ***printouts*** ou ***hard copys***, para além de darem o *output* (ou seja a informação processada fornecida pelo computador como seria visualizada no *écran*), permitem o seu transporte para todo o lado; esta característica torna as impressões importantes para todas as pessoas que têm necessidade de apresentar ou utilizar *outputs* sem ser junto ao computador. Alguns exemplos: quando um vendedor visita um cliente pode precisar de levar consigo uma impressão com os registos das vendas; uma empresa de cartões de crédito usa impressoras para emitir os extractos das contas que envia aos seus clientes. Se não produzissem *hardcopys*, os computadores não teriam tanta utilidade como têm.



Impressora

Outros periféricos de saída que também permitem a produção de *hardcopys*, mas que em vez de imprimirem palavras e números traçam gráficos e desenhos, são os **traçadores de gráficos** *(plotters)*. São muito úteis para engenheiros, arquitectos, matemáticos e outros profissionais para quem os gráficos e os desenhos fazem parte integrante do seu trabalho.



Traçador de gráficos

Outros periféricos de saída são os **gravadores de cassettes, as unidades de disquetes, os gravadores de bobinas e as unidades de discos rígidos.** Se reparar bem estes termos são-lhe familiares porque já falámos deles como unidades de entrada. Isto significa que, relativamente a estes periféricos o computador, para além de poder *receber* a informação neles gravados, também pode *enviar* informação para aí ficar armazenada.

Um **bug** é um erro de programação.

Os periféricos que executam ambas as funções — de entrada e de saída de dados — são conhecidos pelos nomes de **unidades de entrada/saída** ou **unidades I/O**, derivando I/O de *Input/Output*. Por exemplo é possível escrever um programa num microcomputador utilizando como unidade de entrada o teclado; quando se prime o comando RUN os resultados aparecem no *écran*, a unidade de saída. Depois de fazer «correr» o programa várias vezes e de corrigir quaisquer erros ou defeitos que contenha (e que são conhecidos pelo termo inglês *bug*) pode guardá-lo em cassette (utilizando o comando SAVE); nesta operação o gravador de cassettes funciona como unidade de saída. Pode-se então desligar o computador e ir calmamente para casa. No dia seguinte, se se quiser mostrar o programa aos colegas, a cassette e o gravador de cassettes vão funcionar como unidades de entrada. Dá-se ao computador a instrução para o programa ser carregado da cassette para o computador (LOAD) e em seguida a de fazer correr o programa (RUN). Como vê, o gravador de cassettes foi utilizado como unidade I/O. Estas unidades são por vezes conhecidas como **memórias secundárias** ou **memórias extensíveis**, porque contêm e armazenam informação fora da memória principal do computador, ou ainda por **memórias de massa**.

tente responder

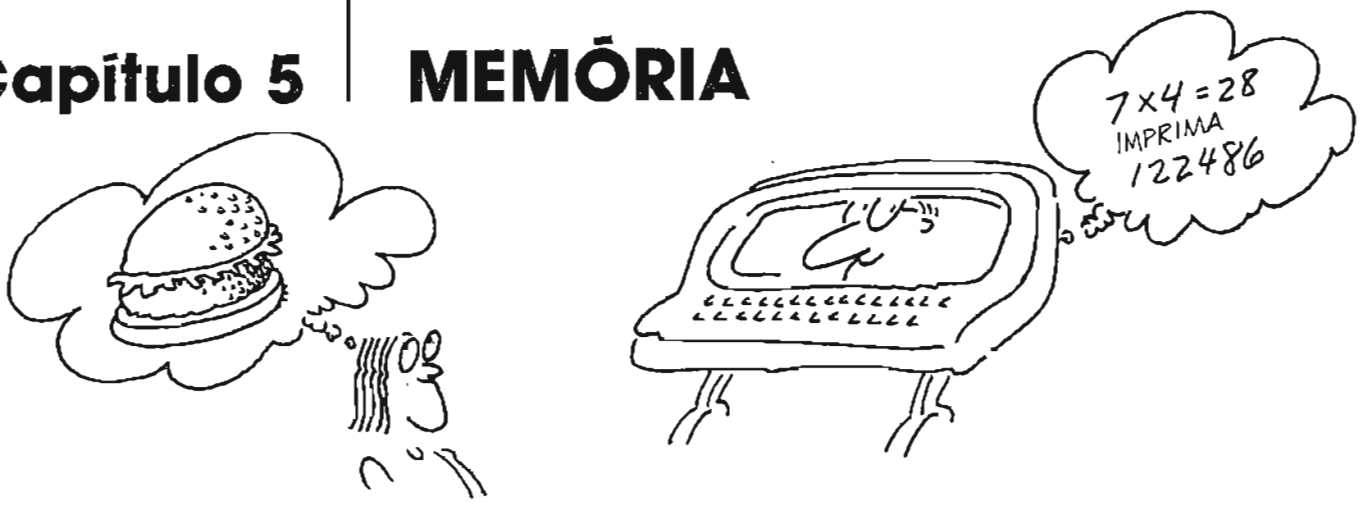
1. Dos dispositivos seguintes diga quais os que são unidades de entrada (I), unidades de saída (O) ou unidades de entrada/saída (I/O):

   a. Teclado
   b. Gravador de *cassettes*
   c. Unidade de disquetes
   d. Gravador de bobina
   e. Unidade de discos rígidos
   g. Caneta luminosa
   h. Leitor óptico
   i. *Écran vídeo*
   j. Impressora
   k. Traçador de gráficos

2. Das unidades de entrada indicadas na segunda coluna escolha a que melhor se adapta a cada uma das situações referidas na primeira coluna:

a. Um professor carrega um programa de ortografia num microcomputador.	1. Leitor de cartões
b. Uma empresa de estudos de mercado classifica 10 mil respostas a um inquérito num grande sistema.	2. Leitor de discos
c. Um engenheiro rectifica num *écran* de computador o desenho do sistema de direcção de um automóvel	3. Teclado
d. Um estudante escreve um programa para um microcomputador.	4. Leitor óptico
e. Um operador carrega num computador dados em cartões perfurados.	5. Caneta luminosa

Capítulo 5 MEMÓRIA



Todas as pessoas têm memória. Você pode recordar-se bem de algumas coisas e mal de outras. Recorda-se dos nomes de todos os professores que já teve? E lembra-se do último *hamburger* que comeu?

Um computador também tem memória. Não precisamente idêntica a uma memória humana, pois apenas pode «recordar» dados armazenados (factos, números e programas) e não sensações como o delicioso cheiro do seu bolo predilecto a cozer no forno ou a frescura da água quando se salta para dentro de uma piscina num dia quente.

Realmente um computador tem dois tipos de memória: uma é a chamada RAM, ou **memória de acesso aleatório** (Random Access Memory), e a outra é a ROM, ou **memória apenas de leitura** (Read Only Memory). Vejamos em primeiro lugar a RAM.

Recorde que *dados* significa informação.

Recorde que a informação pode ser armazenada no exterior do computador em discos, bandas, fitas, etc. e carregada no computador através de uma unidade de entrada.

Memória de Acesso Aleatório (RAM)

A **Memória de Acesso Aleatório, RAM** ou **memória viva** pode ser imaginada como uma caixa vazia onde se vão guardar as informações que se escrevem ou carregam no computador. A RAM é a memória onde são guardados os programas. Ao contrário da memória humana, a RAM nunca esquece as informações que tem armazenadas, enquanto o computador estiver ligado. Para demonstrar o poder de «recordação» da RAM experimente teclar este pequeno programa:

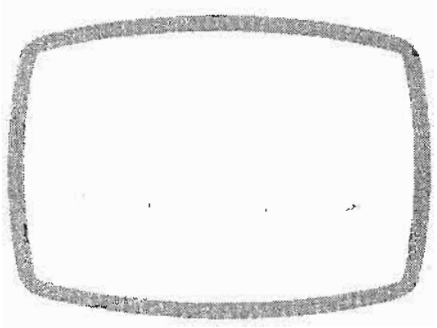
Os programas em BASIC podem ser limpos ou «apagados» da RAM escrevendo a instrução NEW.

```
10 PRINT "LEMBRA-TE DE MIM!"
20 STOP
```

O programa dá ao computador a instrução de imprimir no *écran* a frase «LEMBRA-TE DE MIM!». No final de cada instrução é necessário premir as teclas RETURN ou ENTER que indicam ao computador que se acabou de escrever uma instrução e que agora o processador pode tentar executá-la.

Alguns computadores têm teclas ou comandos que uma vez premidos ou escritos limpam o *écran*, por exemplo o CLS.

Depois limpe o *écran* e deixe o computador ligado.



Mesmo com o *écran* limpo o programa continua na RAM.

Mais tarde (ou mesmo no dia seguinte) quando voltar ao computador prima o comando RUN. É agora que o computador vai demonstrar a sua notável memória, recordando as instruções que lhe foram dadas e imprimindo no *écran* o respectivo *output*:

Recorde que RUN é o comando que indica ao computador para executar cada uma das instruções do programa.

Vamos agora supor que se desejava ver o programa em memória. Um programa carregado na RAM pode ser apresentado no *écran* escrevendo o comando LIST. Se o fizer poderá certificar-se do programa existente na RAM:

LIST é um termo da linguagem BASIC.

A RAM é uma memória *temporária* o que significa que as informações aí guardadas se podem alterar. Pode, por exemplo, alterar-se parte de um programa ou dos dados de um programa ou até apagar-se todo um programa e substituí-lo na RAM por um novo. São estas características que permitem que um computador seja usado para muitas tarefas.

Os programas que se compram são normalmente protegidos, de tal forma que não podem ser alterados.

Como a RAM é uma memória temporária todas as informações aí armazenadas se perdem ou se apagam quando o computador se desliga. Para evitar que todo o trabalho de escrita de um programa no teclado de um computador, que por vezes pode demorar horas, se perca pelo desligar, acidental ou não, do computador, utilizam-se periféricos de saída onde o

programa fica gravado em fita ou disco através da instrução SAVE. Quando se quiser usar o programa novamente, carregar-se-á do disco ou da fita para a memória RAM do computador.

tente responder

1. Qual o significado de RAM?
2. Quando se liga um computador por que razão a RAM é como uma «caixa vazia»?
3. Por que razão se chama à RAM memória temporária?

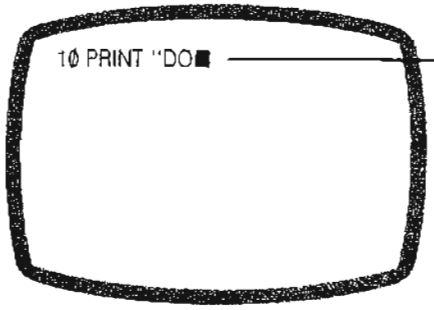
Memória Apenas de Leitura (ROM)

A **ROM** ou **Memória Apenas de Leitura** é o outro tipo de memória do computador. A ROM armazena os programas de que o computador necessita para poder funcionar e que são aí introduzidos quando da montagem do computador na fábrica. A ROM é uma *memória permanente* pois os seus programas conservam-se em memória esteja o computador ligado ou desligado e não podem ser alterados.

O comando NEW não apaga os programas da ROM.

Os diferentes tipos de computador têm diferentes tipos de programas armazenados na ROM. A maior parte destas memórias contêm programas destinados:

a fazer aparecer no *écran* símbolos especiais, como um cursor;



O cursor é um quadrado cintilante ou outro símbolo que indica onde vai ser colocado o carácter seguinte.

a emitir, quando necessário, mensagens de erro;

O comando RUN foi escrito incorrectamente e o computador «reagiu» imprimindo uma mensagem que significa «erro de sintaxe».

Acerca deste tema veja-se o capítulo 7

a traduzir o BASIC para a linguagem própria dos computadores, em numeração binária.

1. Qual o significado de ROM?
2. Por que razão a ROM é uma memória permanente?
3. Dê um exemplo do que pode fazer um programa da ROM.

RAM e ROM

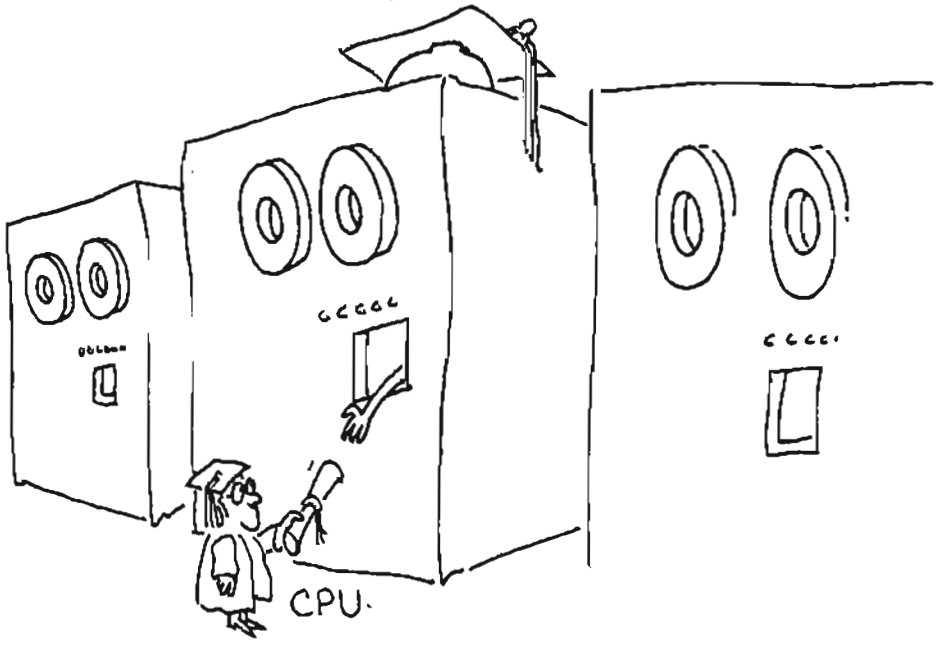
Para funcionarem devidamente os computadores carecem dos dois tipos de memória (RAM e ROM): a ROM guarda as instruções permanentes de operação do computador e a RAM os programas e os dados que se introduzem no computador. Façamos agora uma revisão dos dois tipos de memória do computador.

RAM Memória de acesso aleatório	ROM Memória apenas de leitura
Memória temporária	Memória permanente
Armazena os programas que se escrevem	Armazena os programas introduzidos na memória aquando da montagem do computador
Os programas armazenados podem ser alterados	Os programas armazenados não podem ser alterados

1. Das características das memórias adiante descritas indique as que se referem à RAM ou à ROM:
   a. Armazena os programas que você escreve.
   b. Armazena programas permanentemente.
   c. Os seus programas podem apagar-se ou até perder-se.
   d. Podem fazer-se alterações num programa aí armazenado.
2. Em que é que a memória de um computador difere da memória de uma pessoa?

Capítulo 6

UNIDADE CENTRAL DE PROCESSAMENTO



Porque pensa que a CPU é chamada de «cérebro»? É-o realmente?

O «cérebro» do computador é a **Unidade Central de Processamento, CPU** ou **UCP**. É aí que têm lugar todo o processamento e todos os cálculos. Embora chamemos à CPU um «cérebro» ela não é nada como um cérebro humano pois não pode pensar ou raciocinar, apenas executando as instruções que lhe são dadas. A CPU é constituída por duas partes — a unidade aritmética e a unidade de controlo.

Unidade Aritmética

A **unidade aritmética** é a parte de um computador que executa os cálculos. Mesmo quando programado para fazer problemas aritméticos muito complicados, na unidade aritmética de um computador apenas se **adicionam** e **comparam** números. Então como pode resolver outros tipos de problemas como as multiplicações? Imagine, por um momento, que apenas sabe somar — e que nunca aprendeu a multiplicar. Como poderia resolver um problema como o seguinte?

$$8 \times 6$$

Se se lembrar que a multiplicação é uma adição sucessiva, poderá adicionar

$$6+6+6+6+6+6+6+6$$

ou...
8+8+8+8+8+8

É possível assim resolver o problema adicionando o número 6 oito vezes, embora se demore um pouco mais do que multiplicando 8 por 6.

E você arrisca-se ainda a fazer alguns erros ao longo deste processo de cálculo.

A unidade aritmética também pode subtrair, dividir e fazer outras operações, mas redu-las todas a adições efectuando assim os cálculos.

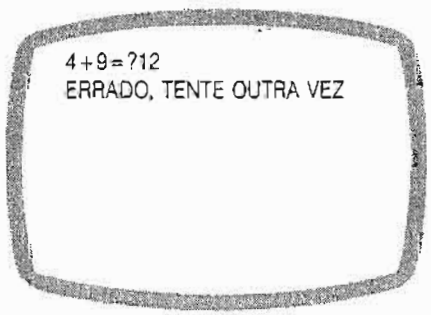
Retenha que para resolver este tipo de problemas o computador tem de ser previamente programado.

Num computador a velocidade de comparação de números é tão grande como a de cálculo.

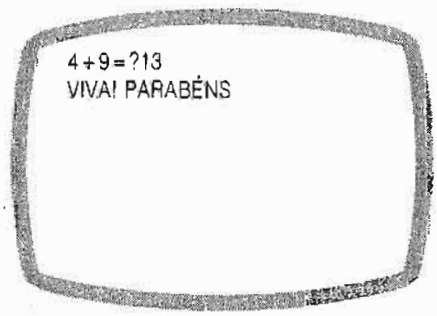
E se o problema for 34×589? Se apenas souber somar e não souber multiplicar poderá adicionar 589+589+589+ . . . 34 vezes mas demorará multíssimo tempo. Um computador reduz a execução de todas as multiplicações a adições sucessivas, mas praticamente não demora tempo: aquela operação demora a executar menos de um milionésimo de segundo! Mesmo resolvendo o problema da multiplicação pelo caminho «mais longo» das adições sucessivas, o tempo demorado pelo computador é mínimo.

Os cientistas medem a velocidade de um computador em **nanosegundos** ou seja um espaço de tempo 1000 milhões de vezes mais pequeno que o segundo. Para resolver 25 692×587 104 999 um computador demora 4 nanosegundos. É esta velocidade fantástica da unidade aritmética que faz com que o computador seja uma óptima ferramenta. Os cientistas que planeiam os voos espaciais têm de resolver problemas muito complexos que, naturalmente, levariam meses ou, talvez anos a concluir e que com os computadores resolvem rapidamente em minutos, ou mesmo em segundos.

A unidade aritmética também é conhecida por **unidade lógica**, pois não resolve apenas problemas aritméticos: executa ainda o trabalho «lógico» de comparar números. Como exemplo há *software* para computadores escolares que exercita os alunos em cálculos aritméticos: neste tipo de programas pode-se perguntar quanto são 4+9 e como parte do programa a unidade aritmética ou lógica compara a sua resposta com o cálculo correcto (13); se as respostas forem iguais o computador imprime a mensagem «VIVA! PARABÉNS» e se não for imprime «ERRADO, TENTE OUTRA VEZ».

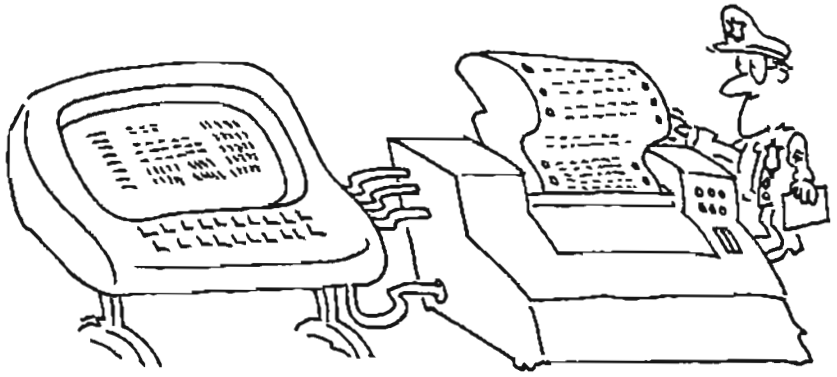


O computador compara a sua resposta com 13.



O computador compara a sua resposta com 13.

Outro exemplo da capacidade da unidade aritmética e lógica na comparação de números é quando um agente da polícia deseja verificar se um carro é roubado. O agente comunica o número da matrícula para o quartel-general onde é introduzido num terminal de computador e é comparado com os das matrículas de carros roubados, registados na sua memória: se o número introduzido for igual a algum dos registados, o computador fornece as informações necessárias.



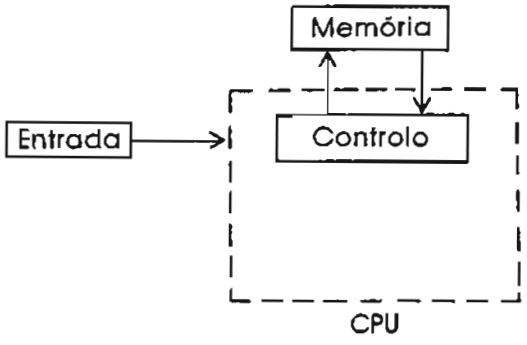
tente responder

1. Seleccione os dois processos de a unidade aritmética processar números:
   a. Somar
   b. Subtrair
   c. Comparar
   b. Multiplicar
2. Porque se chama também unidade lógica à unidade aritmética?
3. Diga como a unidade aritmética resolve o seguinte problema de multiplicação: 4×5.

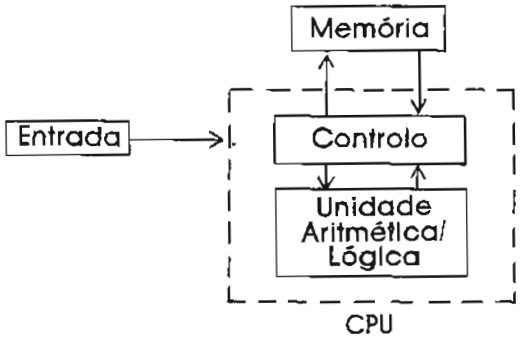
Unidade de Controlo

Quando através de uma unidade de entrada se introduzem dados num computador, o fluxo desta informação é dirigido pela **unidade de controlo** que age um pouco como um polícia de trânsito no controlo do tráfego.

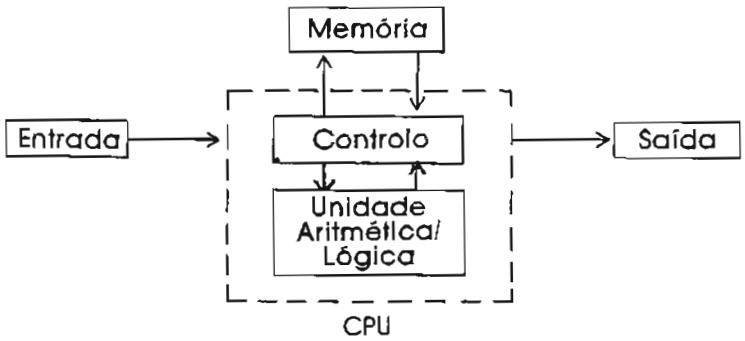
A unidade de controlo envia para a memória a informação a ser aí guardada e vai buscar à memória a informação para ser processada.



Dirige a informação para e da unidade aritmética e lógica onde são feitos cálculos e comparações.

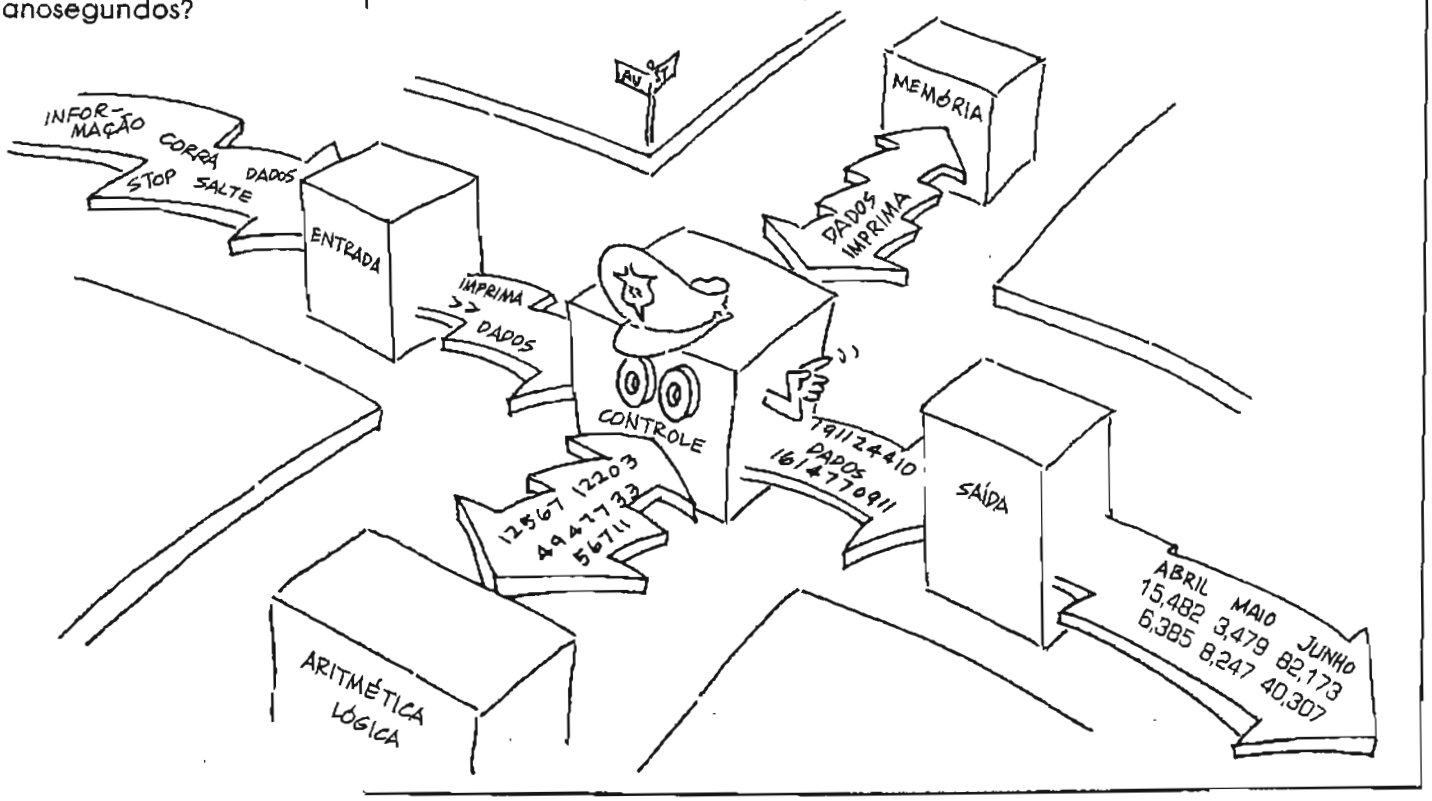


A unidade de controlo tem ainda o papel de dirigir as informações processadas para a unidade de saída.



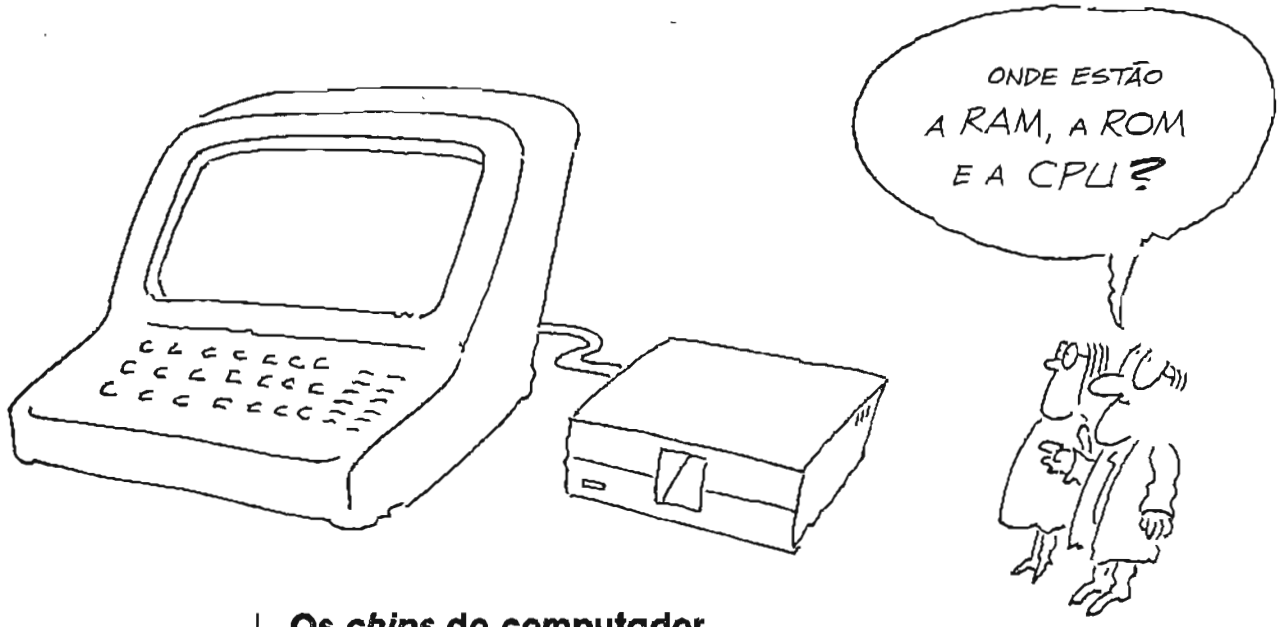
Como vê a unidade de controlo encarrega-se, dentro do computador, de mover a informação para os locais correctos. Este movimento da informação é tão rápido que a sua velocidade também se mede em nanosegundos. Sem unidade de controlo nada se poderia mover no computador: haveria como que um congestionamento de tráfego.

Qual a outra unidade cuja velocidade se mede em nanosegundos?



1. Descreva a função da unidade de controlo.
2. Porque se chama «cérebro» do computador à unidade central de processamento?
3. Que unidade de medida é utilizada para descrever a velocidade a que a CPU faz os seus cálculos?
4. Um estudante está a fazer um exercício de ortografia no seu microcomputador. Diga em que parte da CPU (se na unidade aritmética, se na unidade de controlo) se efectua cada tarefa:
   a. A resposta do aluno é orientada da unidade de entrada para a memória.
   b. A resposta do aluno dirige-se da memória para a unidade aritmética.
   c. A resposta do aluno é comparada com a resposta correcta.
   d. A resposta do aluno é correcta e é enviada para a unidade de saída a mensagem de PARABÉNS.
   e. A resposta do aluno é incorrecta e é enviada para a unidade de saída a mensagem TENTE NOVAMENTE.
   f. No final da lição esta unidade soma o número total de respostas correctas.
   g. Esta unidade envia o número total de respostas correctas para a unidade de saída.

Capítulo 7 CHIPS, BITS e BYTES

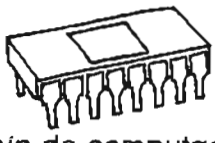


Os *chips* do computador

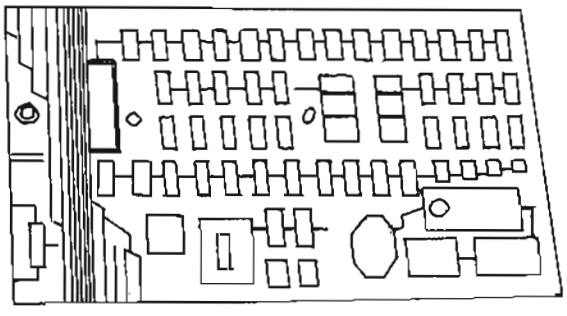
Quando se olha para um computador podemos ver unidades de entrada e unidades de saída como o teclado, o *écran,* o gravador de bobinas e o leitor de discos. Mas onde estão a memória e a unidade central de processamento? Num *mainframe* como num minicomputador estas unidades estão geralmente numa peça de *hardware* à parte que as pessoas referem como sendo o computador, e à qual estão ligados os terminais e outros periféricos I/O. Nalguns microcomputadores a memória e a CPU estão na caixa do teclado, debaixo deste. Mas qual o aspecto da RAM, da ROM e da CPU? Todas elas são ***chips*** **de computador.** O que são *chips* de computador e como funcionam?

Alguns *chips* têm uma cobertura cerâmica em vez de uma cobertura plástica.

Um *chip* de computador parece-se com uma peça de plástico negro com pinos de metal salientes: o plástico negro é a cobertura protectora do *chip* e os pinos tornam possível a ligação do *chip* à placa de circuitos do computador.



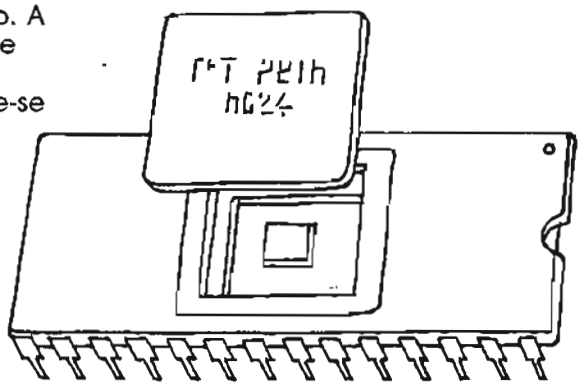
Chip de computador

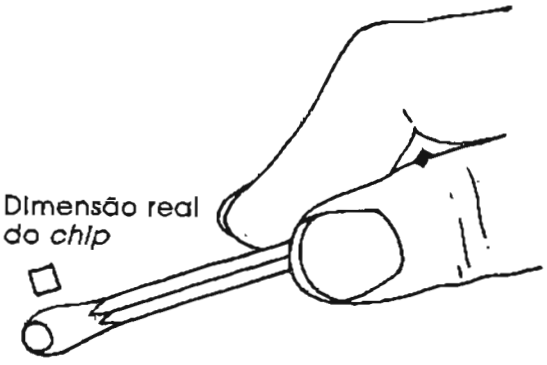


Placa de circuitos do computador

O **silício** é um dos elementos mais vulgares à superfície da Terra encontrando-se na areia, no quartzo e no granito. A pastilha de silício de um vulgar computador parece-se com uma pequena lasca de metal.

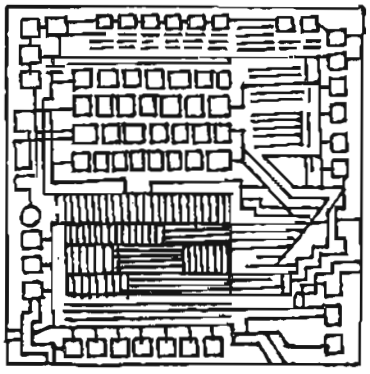
Actualmente, o *chip* é uma delgada pastilha de silício com cerca de 1 cm de cada lado. Se olhar atentamente para a placa de circuitos de um *chip* poderá ver o que parecem ser uns arranhões ou gravações no metal.





Um **circuito** é um caminho através do qual flui a electricidade.

Se conseguir olhar ainda mais atentamente, verá que tais «gravações» são minúsculos circuitos que formam uma teia, uns à volta dos outros, havendo cerca de 10 a 20 mil num único *chip!*



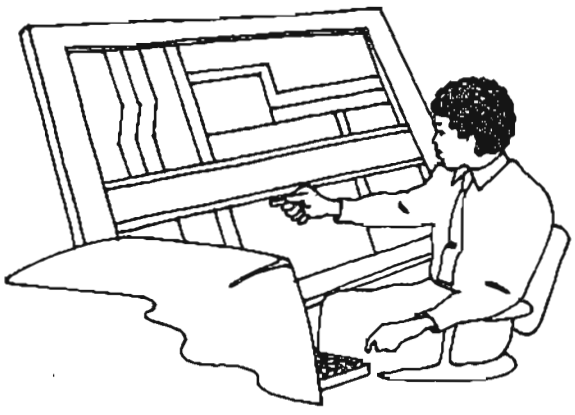
Chip ampliado

Todos estes circuitos são interligados ou integrados uns nos outros. Por essa razão estes *chips* chamam-se *circuitos integrados.*

Como podem estes circuitos ser colocados num único *chip?* À técnica de pôr tantos circuitos num único minúsculo *chip* chama-se **integração em grande escala** (LSI). A integração em grande escala envolve muitos processos complexos de que a seguir se indicam os principais passos.

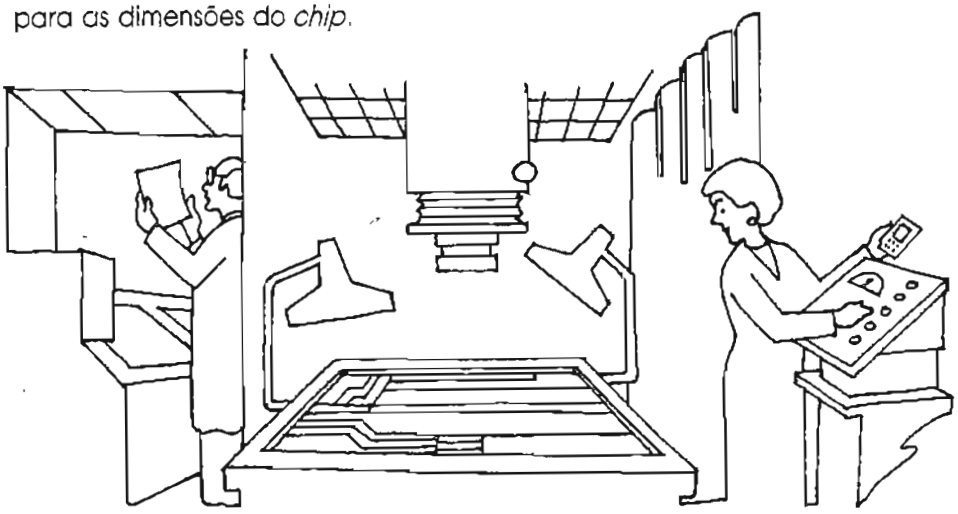
Recorde que um **traçador de gráficos** é uma unidade de saída que desenha figuras e gráficos.

O *chip* começa por ser planeado num computador por um engenheiro. Depois o projecto é impresso num traçador de gráficos.



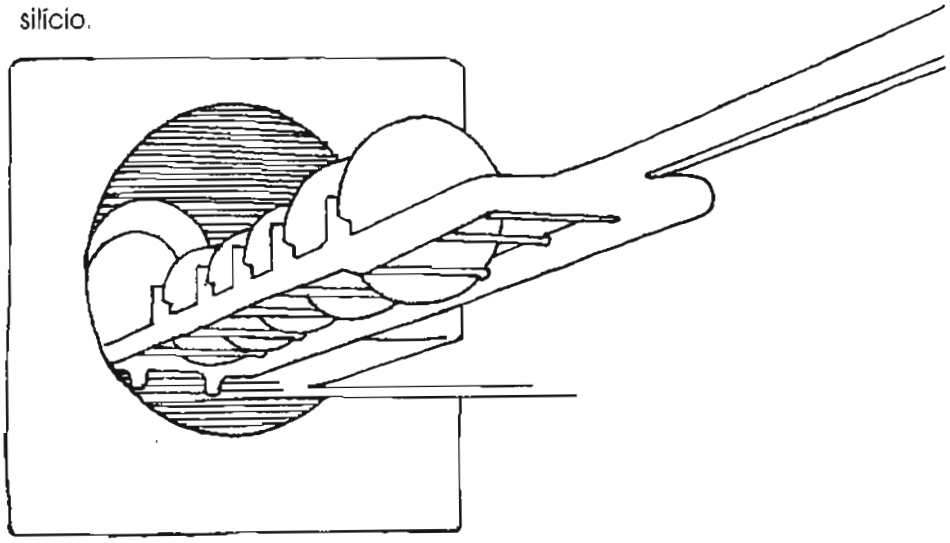
O desenho dos circuitos, extremamente preciso, é fotografado e reduzido para as dimensões do *chip*.

É pela fotografia de raios *laser* que se consegue o desenho pormenorizado dos circuitos.



Esta fotografia é «cosida» numa fina pastilha de silício. Durante este processo, as linhas marcadas na fotografia fazem um sulco, ou gravação, no silício.

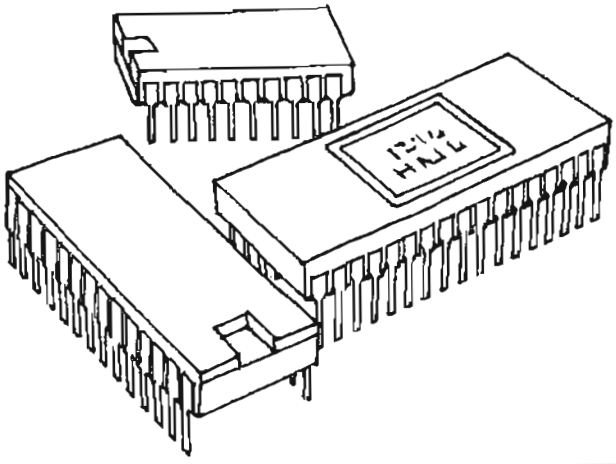
Cada pastilha de silício tem dúzias de circuitos gravados. Depois de passar através de uma câmara magnética ela vai ser dividida em *chips* individuais de cerca de 1 cm de lado.



A pastilha de silício assim gravada passa por uma câmara magnética onde as gravações são magnetizadas, passando agora a ser capazes de conduzir electricidade — tornaram-se circuitos eléctricos.

Coloca-se então o *chip* num invólucro plástico com pinos, os quais vão ser colocados na placa de circuitos do computador.

Os *chips* são encerrados num invólucro plástico ou cerâmico.



Num microcomputador toda a CPU é apenas um único *chip*, a que se chama **microprocessador.**

A maior parte dos *chips* de circuitos integrados custa menos de 1500 escudos.

Como é evidente nem todos os *chips* são iguais. Um *chip* da ROM é diferente de um *chip* da RAM e qualquer destes diferente de um *chip* da CPU. E os próprios fabricantes de computadores fazem diferentes tipos de *chips* ROM, RAM e CPU. A invenção da LSI (integração em grande escala) levou todos estes tipos de *chips* a ser produzidos em grandes quantidades e a custo muito reduzido, conduzindo ao desenvolvimento dos computadores mais pequenos e mais baratos que muitas empresas, escolas e pessoas, usam hoje em dia.[1]

1. De que elemento são feitos os *chips* do computador?
2. Qual é o tamanho aproximado de um *chip* de computador?
3. Qual o número aproximado de circuitos que há num *chip?*
4. Qual é o nome completo de um *chip* com muitos circuitos interligados?
5. Qual é o nome do processo de colocação de milhares de circuitos num único *chip?*

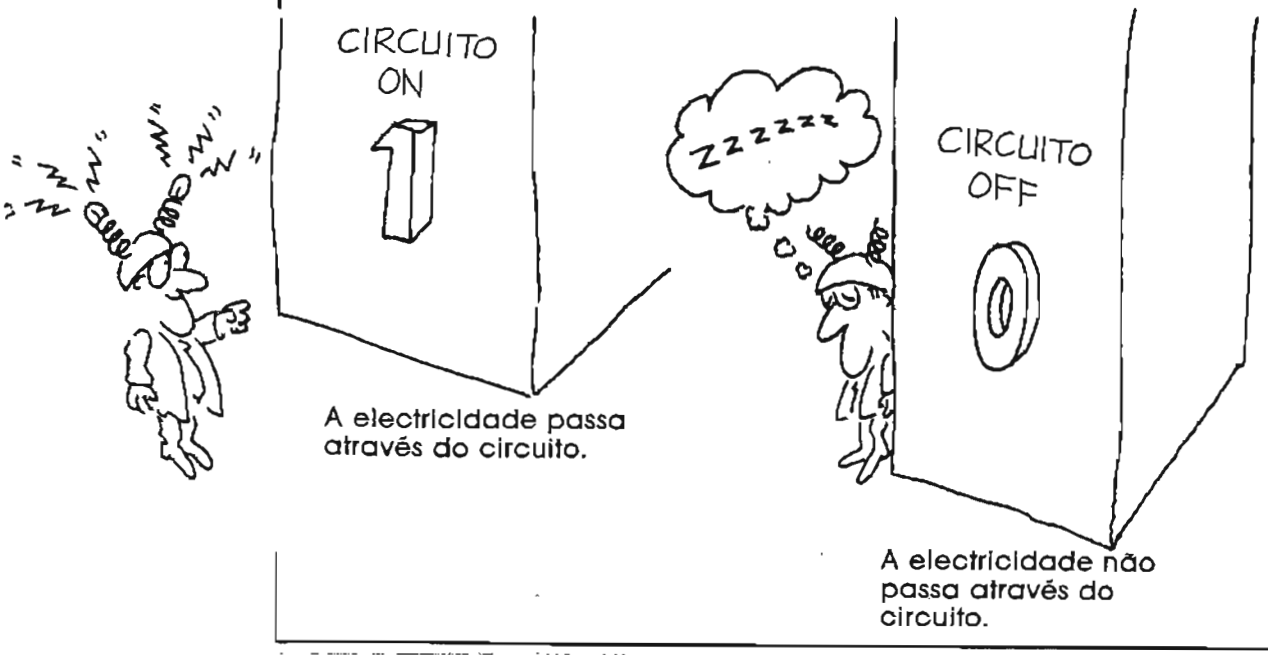
Bits e Bytes

Vejamos agora como funciona um *chip* de computador. Para cada circuito de um *chip* de computador há duas possibilidades:

— através do circuito passa a corrente eléctrica; ou

— através do circuito não passa a corrente eléctrica.

Quando o circuito está ligado *(on)* a electricidade passa através dele; quando o circuito está desligado *(off)* a electricidade não passa. Um circuito *on* é representado pelo dígito 1; um circuito *off* pelo dígito 0.



Em binário, o prefixo **bi** significa «dois». O sistema de numeração binária tem apenas dois dígitos 0 e 1. O sistema de numeração corrente que usamos diariamente é um sistema de *numeração decimal* em que há dez dígitos do 0 ao 9.

O computador faz todos os seus cálculos e o respectivo processamento utilizando apenas os dígitos 0 e 1 que constituem o **sistema de numeração binária**. E só precisa daqueles dois dígitos pois tudo o que ele «lê» é traduzido numa série de *zeros* e *uns*. Uma instrução lida como 0 ordena ao computador para desligar um circuito; uma instrução lida como 1 ordena ao computador para ligar um circuito.

Aos dois dígitos 0 e 1 chamam-se **bits**, palavra derivada de «dígito binário» (em inglês *binary digit*).

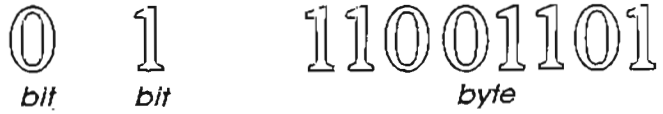
binary digit → bit

Sempre que um computador «lê» uma instrução converte-a numa série de *bits* (1s e 0s). Na maior parte dos computadores todas as letras, números e símbolos são traduzidos em *oito* bits, ou seja numa combinação de oito zeros e uns. Por exemplo quando se prime a tecla da letra A o computador traduz A em 01000001; a letra B converter-se-á em 01000010. A seguir apresentam-se alguns outros:

X 01011000	Z 01011010	? 00111111	2 00110010
Y 01011001	$ 00100100	1 00110001	3 00110011

Byte pronuncia-se «baite».

Cada um dos caracteres do teclado converte-se numa determinada combinação de oito *bits*. A cada um destes grupos de oito *bits* chama-se **byte**. Um *byte* são oito dígitos binários; ou sejam oito *bits*, ou ainda oito *zeros* e *uns*.



Suponhamos que você tecla esta instrução no seu microcomputador:

PRINT "BOO"

O computador converte cada carácter e cada símbolo em um *byte* (oito dígitos binários). Cada *byte* indica ao computador o que fazer com os seus circuitos — colocá-los em «on» (a electricidade circula) ou em «off» (a electricidade não circula).

Byte	Circuitos							
	1	2	3	4	5	6	7	8
P 01010000	off	on	off	on	off	off	off	off
R 01010010	off	on	off	on	off	off	on	off
I 01001001	off	on	off	off	on	off	off	on
N 01001110	off	on	off	off	on	on	on	off
T 01010100	off	on	off	on	off	on	off	off
" 00100010	off	off	on	off	off	off	on	off
B 01000010	off	on	off	off	off	off	on	off
O 01001111	off	on	off	off	on	on	on	on
O 01001111	off	on	off	off	on	on	on	on
" 00100010	off	off	on	off	off	off	on	off

Recorde que o nanosegundo é mil milhões de vezes menor que um segundo.

Lembre-se que é um programa da ROM que ordena ao computador a emissão deste tipo de mensagem.

Uma memória de 16 K armazena, mais precisamente, 16 384 bytes.

Há computadores de grande porte com capacidades de vários milhares de bytes ou seja, 1000 × 1000 = 1 000 000 de bytes.

O computador traduz cada um dos caracteres num *byte* ligando e desligando circuitos de acordo com as instruções «1» e «0». Embora o processo possa parecer lento e fastidioso tem lugar em poucos nanosegundos.

As memórias de computador medem-se em *bytes*. A RAM do seu computador, por exemplo, pode ter aproximadamente 16 000 *bytes* o que significa que a respectiva capacidade de armazenagem é aproximadamente de 16 000 *bytes* de informação, ou sejam 16 000 grupos de oito *bits* ou 16 000 grupos de oito circuitos ligados ou desligados. Recorde que cada *byte* representa um carácter tal como uma letra ou um número. Se tentar escrever no teclado ou carregar um programa com mais de 16 000 caracteres o computador imprimirá a mensagem OUT OF MEMORY (memória ultrapassada) ou algo equivalente.

Outra forma de indicar a dimensão de uma memória RAM de 16 000 bytes é 16 K *bytes*. «K» significa *quilo* ou *um milhar*, pelo que 16 K representa 16 000. (Em rigor, *quilo* significa, em contexto computacional, 1024 e não 1000). As memórias de acesso aleatório dos microcomputadores disponíveis no mercado têm capacidades que variam entre os 16 e os 640 K ou mesmo mais de 1000 K e a sua aquisição deve ter em conta essa característica. Também se pode, em alguns computadores, adicionar mais memória; se, por exemplo, tiver um computador de 16 K, com a adição de *chips* RAM suplementares aumenta a sua capacidade de memória.

tente responder

1. Qual o nome do sistema de numeração usado nos computadores?
2. Qual o significado dos dígitos 0 e 1 em termos de corrente de electricidade através de um circuito num *chip*?
3. Os dígitos binários 0 e 1 têm um nome mais pequeno. Qual é?
4. Quando um computador lê o carácter C converte-o em oito *bits* (01000011). Como se chama a este grupo de oito *bits*?
5. Que significa o facto de um computador ter uma memória RAM de 16 K ou 16 000 *bytes*?
6. Se tiver acesso a um computador, verifique qual a capacidade da sua memória de acesso aleatório (RAM).

FICHA DE AVALIAÇÃO

Escolha a melhor alternativa para cada questão.

1. Que tipo de sistema computacional poderia usar uma grande companhia de aviação para planear os seus vôos e fazer as respectivas marcações de lugares?

   a. computador *mainframe*
   b. minicomputador
   c. microcomputador

2. Qual das seguintes declarações é verdadeira para um microcomputador?

   a. pode fazer muitos trabalhos ao mesmo tempo
   b. pode colocar-se numa secretária e deslocar-se facilmente
   c. custa mais de 15 mil contos

3. Que parte do computador processa a informação?

   a. unidade de entrada
   b. memória
   c. unidade central de processamento

4. Escolha a afirmação verdadeira:

   a. os computadores carecem de instruções para resolverem problemas
   b. os computadores têm cérebros
   c. os computadores são mais inteligentes que os seres humanos

5. Das seguintes, qual é uma linguagem de computador?

   a. Inglês
   b. BASIC
   c. RAM

6. Um exemplo de unidade de entrada periférica é:

   a. uma impressora
   b. um leitor de cartões perfurados
   c. um *écran vídeo*

7. Um programa de computador é:

   a. um *chip*
   b. uma unidade de saída periférica
   c. um conjunto de instruções dizendo ao computador o que tem de executar

8. Os programas de computador são conhecidos por:

   a. *software*
   b. *hardware*
   c. microprocessadores

9. Qual das seguintes afirmações se aplica à ROM (memória apenas de leitura):

   a. é uma memória temporária
   b. os seus programas podem ser apagados
   c. os seus programas não podem ser apagados

10. A unidade aritmética e a unidade de controlo são partes da:

    a. integração em grande escala
    b. unidade central de processamento
    c. programa de computador

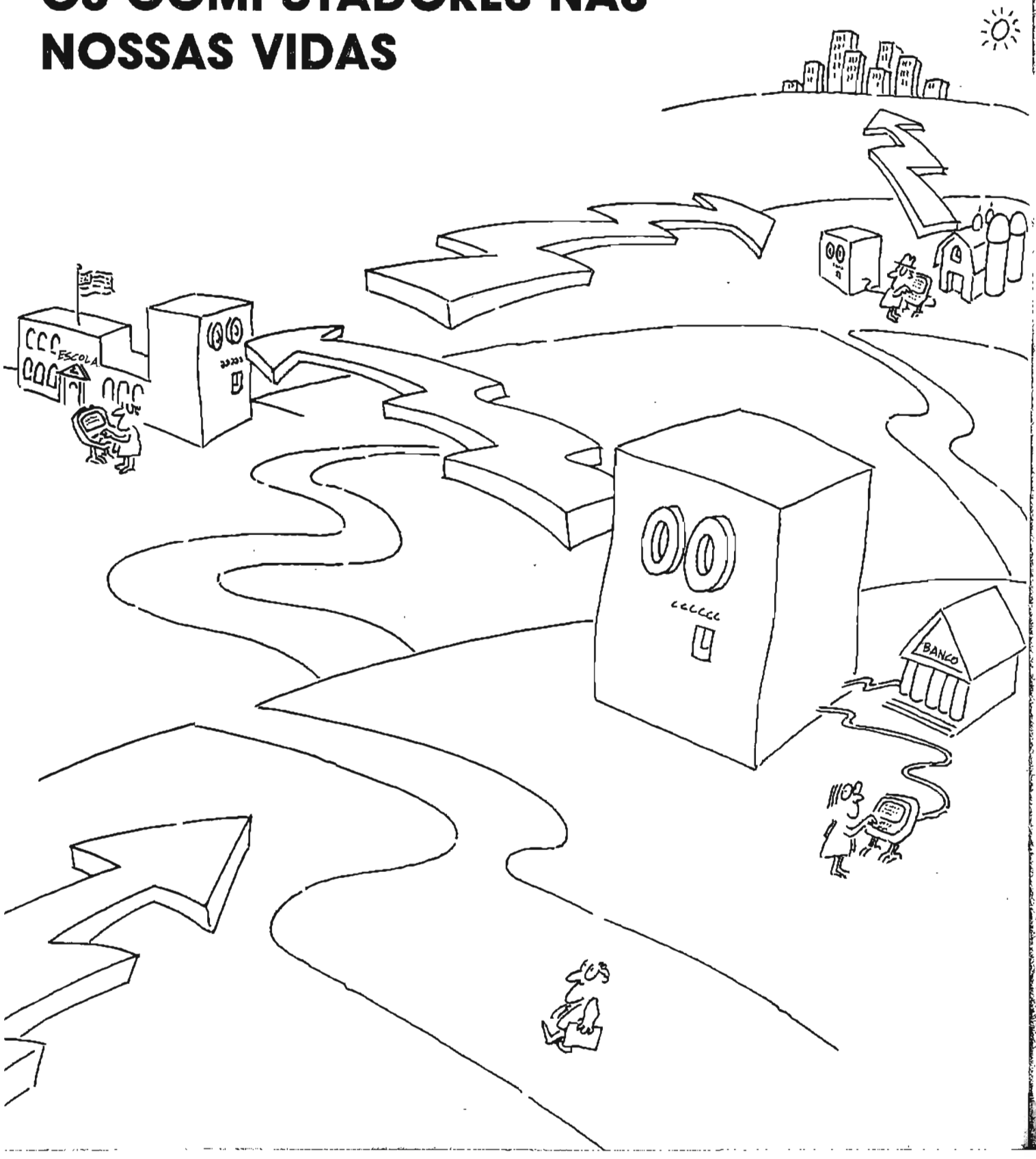
11. O sistema de numeração usado nos computadores é designado por:

    a. sistema binário de numeração
    b. sistema decimal de numeração
    c. sistema métrico

12. Num microcomputador, RAM, ROM e CPU são:

    a. dispositivos de entrada
    b. *chips* de circuitos integrados
    c. dígitos binários

UNIDADE 2
OS COMPUTADORES NAS NOSSAS VIDAS



Capítulo 8

DO ÁBACO AO IBM

Na unidade 1 vimos o que é um computador e como funciona. As suas quatro funções — receber informação, armazenar informação, processar informação e fornecer informação processada — não são ideias novas: desde o início da civilização que o homem tem vindo a desenvolver instrumentos ou mecanismos que o auxiliem nessas tarefas.

Neste capítulo iremos até ao passado, ver alguns dos desenvolvimentos que levaram à invenção dos computadores modernos.

Na Antiguidade

Desde os tempos mais antigos que os homens usaram os *dedos* quando queriam demonstrar «quantos são»: era assim que exprimiam o número de animais mortos numa caçada ou o número de moradores de uma habitação; era fácil referirem-se a grandes números como grupos de dez para o que levantavam ambas as mãos abertas. Por isso, o dez se tornou a base do nosso sistema actual de numeração.

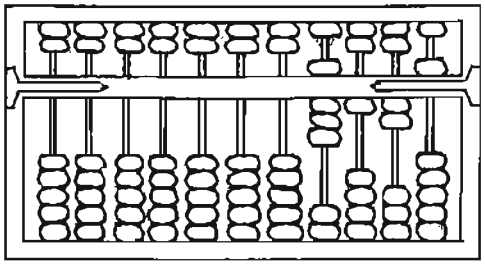
Os números são referidos como **dígitos,** palavra que deriva do latim *digitus* que significa «dedo».

O nosso sistema de numeração é chamado **decimal.** O prefixo *deca* significa «dez».

Os povos primitivos careciam também de um processo de cálculo e de armazenamento de informações para utilização posterior: o número de animais mortos era registado reunindo pequenas pedras ou calhaus num monte em que cada pedra representava um animal; mais tarde começaram a registar e armazenar informações, fazendo entalhes e símbolos em pedra ou madeira.

O **ábaco** foi um dos primeiros instrumentos utilizados para simbolizar números. O ábaco chinês, desenvolvido há cerca de 5000 anos, era construído com madeira e contas e podia-se segurar e transportar facilmente. O ábaco teve tanto sucesso que, a partir da China, o seu uso espalhou-se por muitos outros países ainda se utilizando hoje em dia nalguns deles.

Os egípcios, os romanos e os gregos também tinham versões do **ábaco,** palavra que deriva do grego *abax* e que significa «tábua ou prancha de cálculo».



Ábaco

O ábaco é um dispositivo de armazenagem de números, tal como um lápis e um papel também servem para os registar quando se fazem cálculos.

O ábaco não faz realmente o cálculo, como as calculadoras actuais. Ajuda as pessoas a informarem-se dos números à medida que os cálculos são feitos. Os peritos na utilização de um ábaco podem fazer cálculos tão rapidamente como uma pessoa que disponha de uma calculadora!

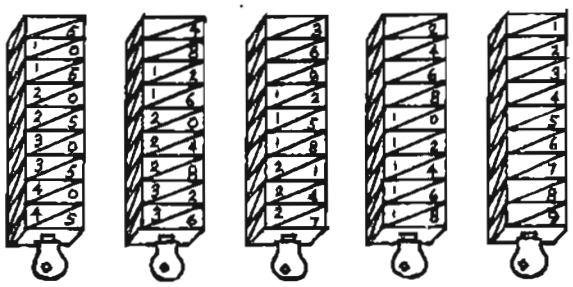
tente responder

1. Qual foi o mais antigo método de contagem utilizado pelo homem?
2. Porque é que este método levou à utilização do sistema de numeração decimal?
3. Que dispositivo inventado pelos antigos chineses é usado para ajudar a fazer cálculos?
4. Por que motivo este antigo instrumento ainda é usado hoje?

No século XVII

Através da história os povos foram desenvolvendo sistemas de numeração e processos de contagem, procurando maneiras de tornar os cálculos mais fáceis. Em 1617 o matemático escocês John Napier inventou umas barras de cálculo conhecidas pelo nome de ossos de Napier e que foram utilizadas como auxiliares da multiplicação de grandes números.

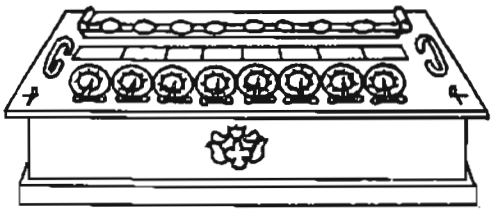
As barras originais eram feitas de osso ou marfim.



Ossos de Napier

Cada uma das barras continha os múltiplos sucessivos de um número. Movendo as barras em roda e lendo as filas de números, era possível com poucas adições, obter o produto de dois números grandes. As barras efectivamente não faziam multiplicações; elas ajudavam uma pessoa a calcular um produto de uma forma rápida e fácil.

Alguns anos mais tarde, em 1642, um jovem matemático francês chamado Blaise Pascal inventou uma máquina que podia adicionar e subtrair números. Pascal trabalhava no escritório do pai onde tinha de fazer enormes somas relativas a impostos; fazer tantas adições era enfadonho e consumia tempo, e por isso Pascal procurou uma maneira de fazer o trabalho mais rapidamente, chegando assim ao seu invento cujas dimensões eram as de uma caixa de sapatos.



A Máquina Aritmética de Pascal

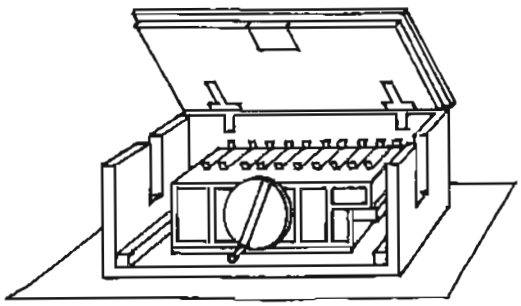
Um conta-quilómetros mede o número de quilómetros que um carro percorre.

Pascal tinha apenas 19 anos, quando inventou esta máquina.

Leibnitz era também advogado e filósofo.

A **máquina aritmética** de Pascal usava rodas dentadas que se moviam cada vez que se adicionava ou subtraía um número. O funcionamento da máquina era muito semelhante ao de um conta-quilómetros de um automóvel moderno. Os números 0 a 9 estavam impressos nas bordas de cada uma das rodas. Quando uma delas completava uma rotação desde o 0 até ao 9 um pequeno dente encarregava-se do respectivo transporte, isto é, movia a roda colocada imediatamente à esquerda do correspondente a uma unidade. A máquina aritmética foi uma das primeiras máquinas que realmente executava cálculos.

Nos cinquenta anos seguintes inventaram-se outras máquinas de calcular que, no entanto, não passaram de simples aperfeiçoamentos da máquina aritmética de Pascal. Em 1694 o matemático alemão Gottfried Wilhelm von Leibnitz construiu uma máquina de calcular muito engenhosa, conhecida por **calculador graduado**, que podia multiplicar e dividir para além de adicionar e subtrair. Para fazer os seus cálculos estes calculadores dispunham de cilindros graduados em vez de rodas e engrenagens.



O calculador graduado de Leibnitz

As máquinas inventadas por Pascal e Leibnitz eram complicadas. Tinham muitas partes móveis. Naquele tempo era difícil construir exemplares perfeitos das máquinas. Assim, e embora as máquinas originais funcionassem bem, poucas foram construídas.

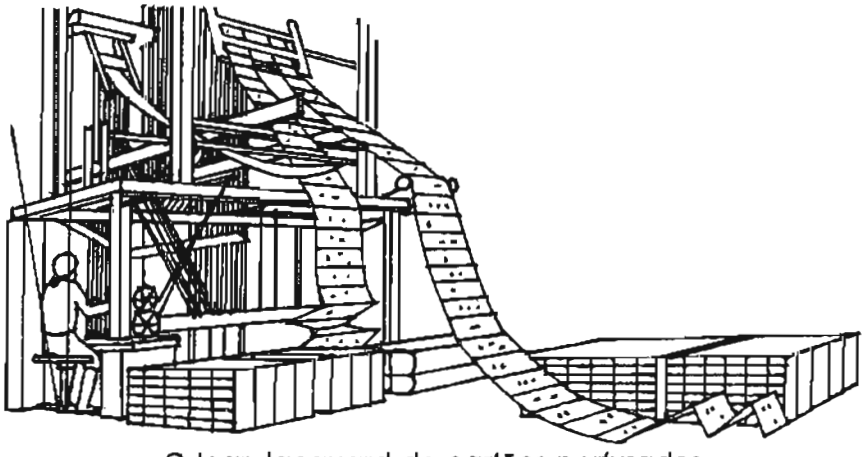
Tente responder

1. Indique o nome do inventor dos seguintes dispositivos:
   a. ossos de Napier
   b. máquina aritmética
   c. calculador graduado
2. Indique o nome do invento que foi usado para fazer cada um dos seguintes conjuntos de operações:
   a. adicionar, subtrair, multiplicar e dividir números
   b. multiplicar números grandes
   c. adicionar e subtrair números
3. Por que razão não teve muita expansão o emprego da Máquina Aritmética e do Calculador Graduado?

No século XIX

Em 1801 o francês Joseph Jacquard inventou um novo tipo de tear que empregava **cartões perfurados** no controlo da operação. Os orifícios do cartão permitiam a passagem de agulhas que puxavam um fio: se num ponto houvesse orifício o fio passava e tornava-se parte do desenho do tecido; se não houvesse nada era desenhado. O processo podia ser repetido quantas vezes se quisesse. O padrão de orifícios de cada novo cartão introduzido debaixo das agulhas determinava quais as que podiam atravessar o cartão, traccionando fios de diferentes cores e espécies. Se um tecelão desejava repetir um desenho bastava-lhe simplesmente fazer passar os mesmos cartões e pela mesma ordem através do tear.

Os tecidos franceses eram muito populares em todo o mundo. Os tecelões franceses eram exímios em procurar sempre maneiras melhores e mais rápidas de tecer os seus modelos.



O tear Jacquard de cartões perfurados

Jacquard, quando projectou o seu tear de cartões perfurados não pensava em computadores. Mas a sua ideia de armazenar informações em cartões perfurados iria ser usada anos mais tarde por uma série de inventores de computadores.

A primeira máquina de calcular que talvez possa verdadeiramente ser chamada de computador foi inventada em Inglaterra por Charles Babbage em 1835. O sonho de Babbage era construir uma máquina que pudesse fazer algo mais do que calcular grandes números. Ela deveria receber instruções, processá-las, armazená-las e finalmente imprimir os resultados. Pensou chamar-lhe **Máquina Analítica**. Babbage planeou usar cartões perfurados para a informação numérica e planeou também que os resultados fossem impressos.

Babbage era um cientista respeitado mas a maior parte das pessoas não entendiam as suas novas e invulgares ideias. Era considerado um excêntrico. Demorou muito tempo a encontrar alguém que quisesse emprestar-lhe dinheiro para construir a sua Máquina Analítica. Foi uma talentosa matemática, Lady Ada Augusta Lovelace que viu como a máquina analítica poderia ser importante e apoiou Babbage na tentativa de juntar dinheiro para a sua construção. Uma das suas mais importantes contribuições foi convencer Babbage a usar na sua máquina o *sistema de numeração binária* em vez do sistema de numeração decimal, o que iria tornar o trabalho da máquina analítica muito mais eficiente.

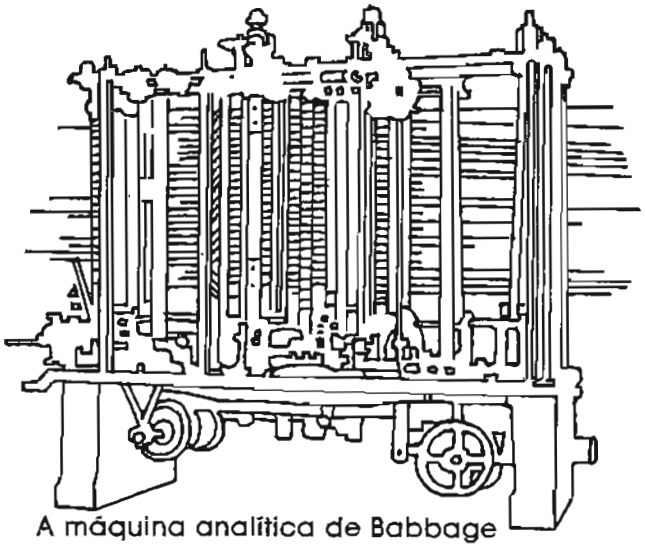
Inicialmente Babbage queria construir a sua máquina para calcular tabelas de informações para a navegação.

Isto não lhe parece familiar? Estas são as quatro funções executadas por um computador.

Babbage fundou na Universidade de Cambridge, em Inglaterra, a Analytic Society, cujos membros espalharam muitas ideias novas por todo o país.

Actualmente todos os computadores usam o sistema binário ou de base 2. Recorda-se do que significam os dois dígitos deste sistema, o 0 e o 1?

A Máquina Analítica está em exposição no British Science Museum, em Londres.



A máquina analítica de Babbage

Ada Augusta Lovelace também escreveu sobre os planos de Babbage para a Máquina Analítica. Destes escritos conclui-se que os seus planos eram como os de um moderno computador: a Máquina Analítica tinha as quatro partes de um sistema de computadores — entradas, saídas, memórias e unidade central de processamento.

Infelizmente, Ada Augusta Lovelace era a única pessoa que apreciava os planos de Babbage. A falta de dinheiro impediu o progresso e a falta de instrumentos de precisão tornou muito difícil a construção da Máquina Analítica. Nos princípios do século XIX não se usava electricidade; apenas se dispunham de instrumentos mecânicos para trabalhar com engrenagens, dentes de engrenagens e rodas, e as ferramentas existentes não eram suficientemente precisas para construir uma tão complicada máquina. Por isso, a Máquina Analítica nunca trabalhou: Babbage morreu considerando-se um fracassado e nunca soube que as suas ideias seriam usadas mais de 100 anos depois no primeiro computador «moderno».

O Departamento de Defesa dos Estados Unidos tem a sua própria linguagem de programação chamada «Ada», nome derivado de Ada Lovelace.

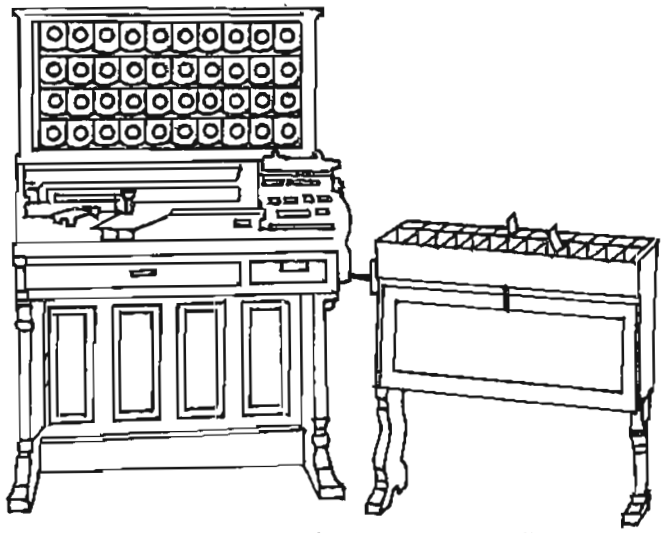
Cerca de 50 anos mais tarde foi construída uma outra máquina importante, a **Máquina de Tabulação** de Herman Hollerith, que se destinava a auxiliar o Governo dos Estados Unidos no censo de 1890. O Governo dos Estados Unidos realiza de dez em dez anos um recenseamento da população onde regista dados acerca de todos os habitantes do país. No censo de 1880 haviam sido recolhidos tantos dados que no momento em que foram classificados e ordenados já se estava quase no momento de realizar o de 1890! Um engenheiro militar, Herman Hollerith pensou numa maneira melhor de registar os dados do censo, indo buscar tal ideia ao tecelão francês Joseph Jacquard — os **cartões perfurados**.

Um **censo** é uma contagem da população.

As informações relativas a cada pessoa eram perfuradas em cartões que eram então colocados na máquina de tabulação projectada por Hollerith. A máquina de tabulação fazia empurrar uma série de pinos contra os cartões: se um pino atravessava um orifício do cartão, contactava com uma superfície metálica situada debaixo do cartão e fechava um circuito eléctrico que fazia a tabuladora adicionar mais uma unidade ao item que estava a ser contado: se não houvesse orifício no cartão o circuito eléctrico não se fechava e nada era adicionado ao total.

Os cartões perfurados são também actualmente chamados de cartões de Hollerith.

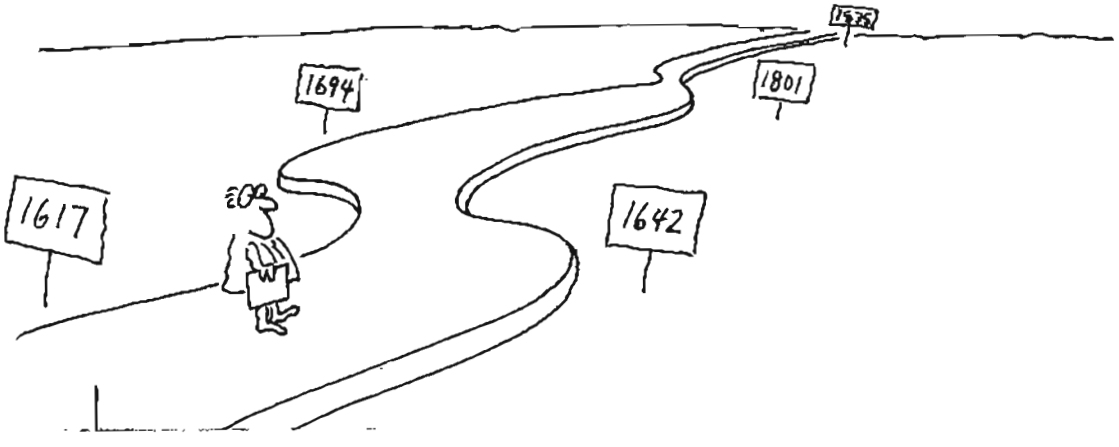
Nesta altura já se usava a electricidade.



A máquina de tabulação de Hollerith

A máquina de Hollerith permitiu fazer uma «rápida» contagem da população dos Estados Unidos. De facto, os dados do censo de 1980 demoraram pouco menos de três anos a ser trabalhados. Era um grande avanço sobre o censo de 1880! O uso da Máquina de Tabulação foi um êxito tão grande que Hollerith formou uma empresa que mais tarde veio a ser conhecida por International Business Machine Company ou seja a IBM.

Marcos miliários na história dos computadores			
Ano	Invenção	Inventor	País
cerca de 3000 a. C.	Ábaco	desconhecido	China
1617	Ossos de Napier	John Napier	Escócia
1642	Máquina Aritmética	Blaise Pascal	França
1694	Calculador graduado	Gottfried Leibnitz	Alemanha
1801	Tear de cartões perfurados	Joseph Jacquard	França
1835	Máquina Analítica	Charles Babbage	Inglaterra
1887	Máquina de Tabulação	Herman Hollerith	Estados Unidos



tente responder

1. Babbage (Máquina Analítica) e Hollerith (Máquina de Tabulação) foram buscar a ideia da forma de introduzir informações nas suas máquinas a um tecelão francês Jacquard. Que ideia era essa?

2. Qual a finalidade de Hollerith ao projectar a Máquina de Tabulação?

3. Em que era a Máquina Analítica de Babbage semelhante a um computador moderno?

4. Com que ideia importante contribuiu Lady Ada Augusta Lovelace para a Máquina Analítica?

5. Algumas pessoas dizem que Babbage «nasceu no século errado». Está de acordo? Porquê?

6. Faça corresponder cada descrição à invenção apropriada:

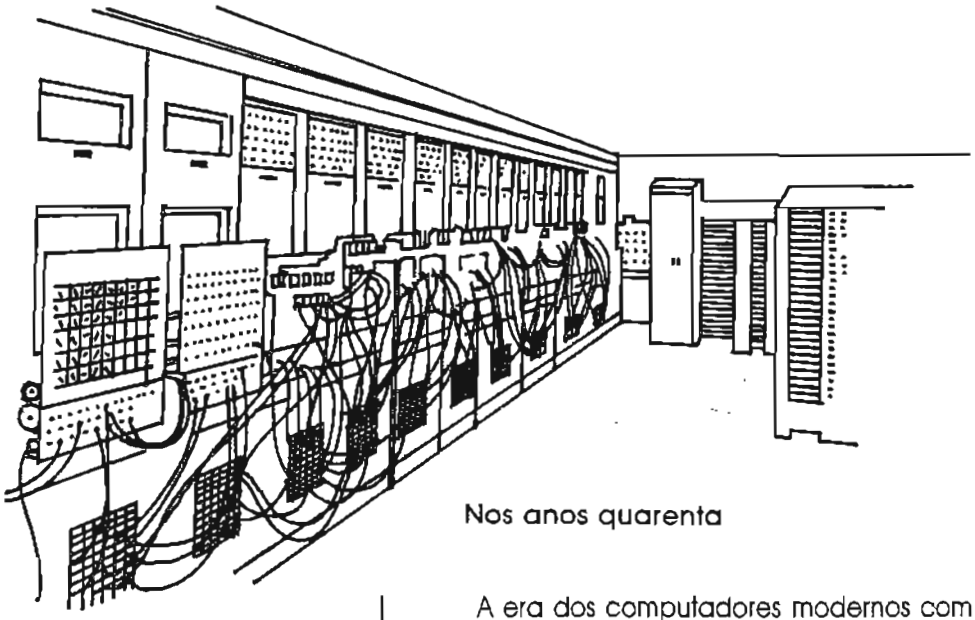
a. usado no censo de 1890 dos Estados Unidos.	1. Calculador graduado.
b. utilizava cartões perfurados para tecer desenhos em tecidos	2. Ossos de Napier.
c. inventada há cerca de 5000 anos	3. Máquina de Tabulação.
d. ajudavam apenas a fazer longos problemas de multiplicação	4. Máquina Analítica.
e. primeira máquina capaz de adicionar, subtrair, multiplicar e dividir	5. Máquina Aritmética.
f. apenas podia adicionar e subtrair	6. Tear de cartões perfurados.
g. projectada para receber, armazenar, processar e fornecer informação processada	7. Ábaco.

Capítulo 9

AS QUATRO GERAÇÕES DOS COMPUTADORES MODERNOS

O homem, à medida que aprende novas coisas, vai usando estes conhecimentos para gerar novas informações. E quanto mais informação tem, melhores maneiras tenta encontrar de a armazenar, de a processar e de a recuperar. Nos últimos quarenta anos deu-se um gigantesco salto no tratamento da informação. Homens e mulheres desenvolveram computadores de alta velocidade que recebem, armazenam, processam e fornecem informação. Os computadores trabalham mais depressa do que pessoas como Pascal e Hollerith alguma vez sonharam ser possível.

Recuperar informação significa «obter informações que estão armazenadas algures».



Nos anos quarenta

Nos anos oitenta

A era dos computadores modernos começou em 1944. Foi nesse ano que um engenheiro americano da Universidade de Harvard, **Howard Aiken** construiu um computador. O seu funcionamento era muito parecido com o da Máquina Analítica de Babbage, projectada cem anos antes. O computador de Aiken, chamado Mark I recebia a informação através de cartões perfurados, armazenava-a, processava-a e imprimia os resultados numa máquina de escrever eléctrica. O Mark I, que permitia a execução de diferentes tarefas, era uma máquina enorme que ocupava o espaço do ginásio de uma escola. Resolvia um problema de matemática em apenas alguns segundos — o que era uma proeza para 1944! O Mark I é hoje considerado como o primeiro **computador electromecânico** do mundo.

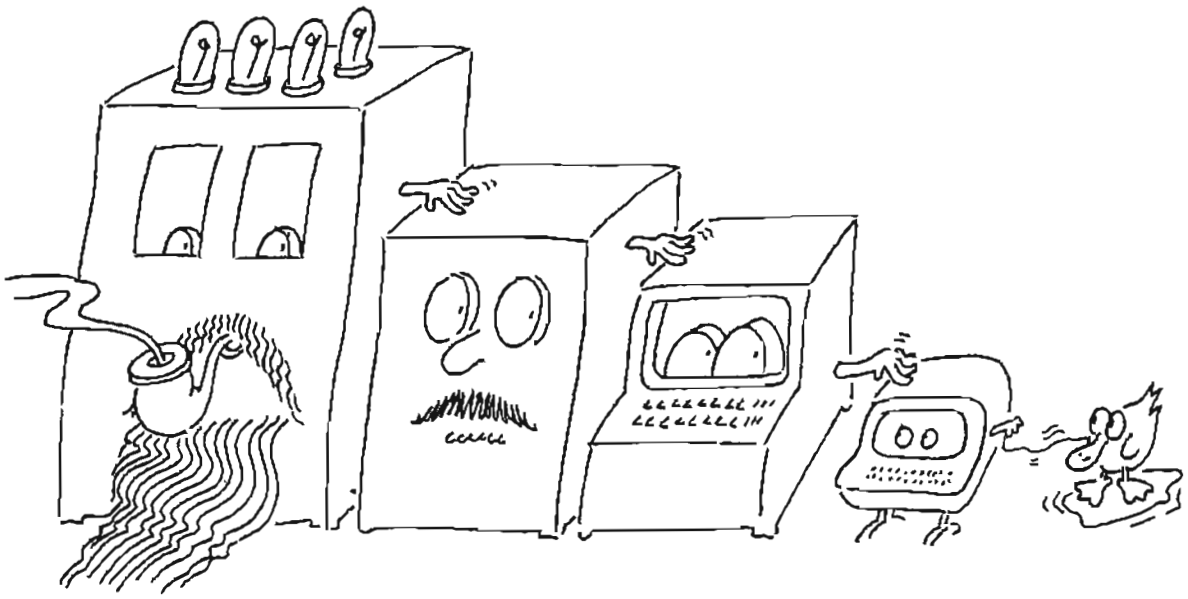
Lembre-se que Babbage não conseguiu construir a sua Máquina Analítica.

Logo após a invenção do Mark I os cientistas começaram a construir computadores sem partes móveis, ou seja electrónicos em vez de mecânicos. A maior parte dos computadores sobre os quais já leu alguma coisa são os chamados **computadores digitais**, que transformam a informação em dígitos, sendo nesta forma que a armazenam e posteriormente processam.

Um **computador electromecânico** é composto de partes eléctricas e mecânicas (móveis).

Digital significa «ter dígitos ou números».

Os computadores electrónicos digitais rapidamente substituiram o Mark I. De facto, poucos anos após a sua construção, os computadores electromecânicos tornaram-se obsoletos e deixaram de ser utilizados. Nos últimos quarenta anos, houve diversas alterações nos computadores digitais, cada uma das quais introduziu uma nova «geração» de computadores. Tal como numa família há pessoas de diferentes gerações, o mesmo acontece com os computadores.



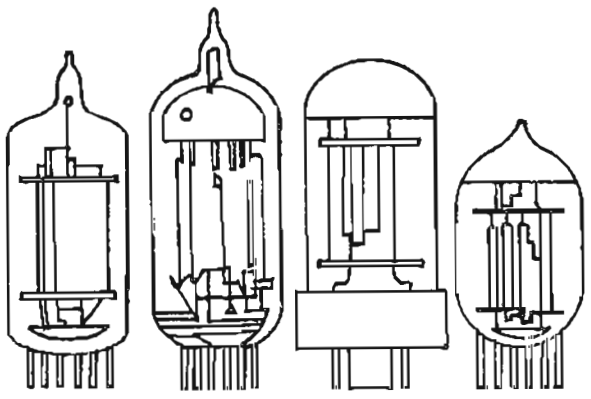
Os computadores da Primeira Geração

À medida que no interior do computador as partes móveis foram substituídas por circuitos eléctricos, os computadores começaram a trabalhar mais depressa e mais eficientemente. O primeiro computador totalmente digital foi concluído em 1946 na Universidade da Pensilvânia sob a direcção de dois engenheiros, John W. Mauchly e J. Presper Eckert. Este computador, o ENIAC era ainda maior que o Mark I e pesava mais de 30 toneladas! A passagem da corrente eléctrica fazia-se através de válvulas e o computador tinha mais de 18 000!

Num circuito eléctrico, o único «movimento» é o fluxo de electricidade.

ENIAC deriva de **E**lectronic **N**umerical **I**ntegrator **a**nd **C**alculator ou seja Calculador e Integrador Numérico Electrónico.

Nos anos cinquenta os receptores de rádio e de televisão usavam válvulas.



Válvulas

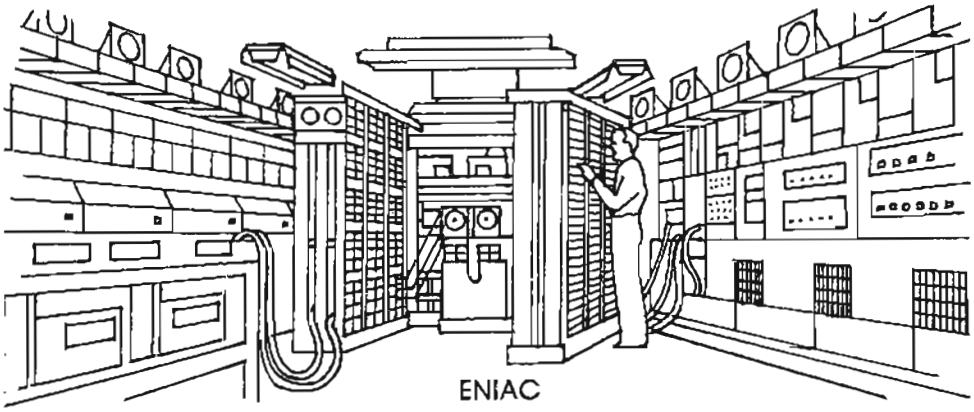
Em 1946 a conta de electricidade do ENIAC era de 1800 dólares por mês! (270 contos ao câmbio actual).

Como as válvulas aquecem, 18 000 válvulas libertavam uma grande quantidade de calor. Para evitar o aquecimento do ENIAC eram necessárias unidades especiais de ar condicionado.

O computador tem realmente um cérebro?

Os computadores actuais podem resolver o problema em poucos minutos ou ainda menos.

O ENIAC era considerado quase um «cérebro». Era 300 vezes mais rápido que o Mark I e trabalhava um milhar de vezes mais rápido que uma pessoa com uma calculadora de secretária. O ENIAC resolveu em duas horas um problema que 100 engenheiros, trabalhando 8 horas por dia teriam demorado um ano inteiro a resolver.



ENIAC

Logo após a construção do ENIAC, **John von Neumann** teve a ideia de armazenar um programa na memória do próprio computador. Até essa altura apenas se guardavam em memória os números utilizados no programa. Esta ideia permitiu a construção de computadores que trabalhavam mais depressa que o ENIAC e nela se baseiam os computadores actuais que também armazenam programas na própria memória do computador.

Alguns anos mais tarde, em 1951, Eckert e Mauchly projectaram um outro computador chamado **UNIVAC.** Era ainda maior que o ENIAC e foi vendido pelos seus criadores ao Serviço de Recenseamento dos Estados Unidos. Foram depois construídos e vendidos outros modelos do UNIVAC que assim se tornou o primeiro computador comercial.

UNIVAC deriva de **Univ**ersal **A**utomatic **C**omputer (Computador automático universal).

Um produto **comercial** é algo que pode ser comprado e vendido.

tente responder

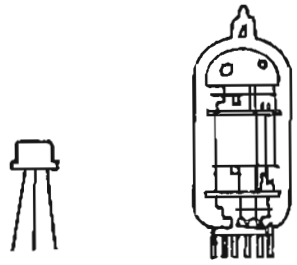
1. Qual o nome do primeiro computador electromecânico do mundo?
2. Quem inventou este computador? Por que razão é considerado «electromecânico»?
3. Que nome é dado aos computadores electrónicos que traduzem as informações em dígitos e é sob esta forma que os armazenam e processam?
4. Diga o nome de dois computadores da primeira geração.
5. Quem foram os inventores destes computadores?
6. Que dispositivos eram usados nos computadores da primeira geração para conduzir electricidade?
7. Imagine que é presidente de uma grande empresa nos anos cinquenta. Indique uma vantagem e um problema que poria a compra de um UNIVAC para a sua empresa.

Um transistor pode fazer o trabalho anteriormente executado por 10 válvulas, usando apenas 1/100 da energia. Assim os gastos em electricidade dos computadores da segunda geração eram muito mais pequenos!

Outras melhorias introduzidas foram uma memória maior e dispositivos de entrada/ saída mais aperfeiçoados.

Os computadores da Segunda Geração

Em finais dos anos cinquenta, as válvulas dos computadores foram substituídas por transistores, que conduzem a corrente eléctrica de uma forma mais rápida e eficiente. E eram também mais seguros: as válvulas «queimavam-se» muitas vezes e necessitavam de ser substituídas; os transistores, pelo contrário, raras vezes precisavam de ser substituídos. Eram ainda mais pequenos que as válvulas e praticamente não aqueciam.



Transistor Válvula

Com a utilização de transistores em vez de válvulas, os computadores tornaram-se mais pequenos. A sua velocidade de resolução de problemas aumentou dez vezes relativamente à dos computadores da primeira geração.

É difícil atribuir a uma única pessoa o êxito da construção de um computador da segunda geração. Eles são tão complexos que foram necessários os esforços de muitas pessoas com diferentes especialidades para projectar todas as suas partes constituintes. Foram várias as grandes empresas que construíram computadores da segunda geração, alguns destinados ao seu próprio uso e alguns para venda a outras empresas.

tente responder

1. Na segunda geração de computadores que dispositivos passaram a ser usados para a condução de electricidade?
2. Indique duas vantagens do novo dispositivo eléctrico, sobre as válvulas.
3. Porque quereria uma empresa adquirir um computador da segunda geração em vez de um UNIVAC?

Recorde-se que um circuito é um «caminho» através do qual flui a corrente eléctrica. Este caminho é geralmente constituído por um fio metálico fino. Os circuitos «integrados» são circuitos que estão ligados uns aos outros.

Os computadores da Terceira Geração

Em 1964 foram desenvolvidos minúsculos circuitos integrados que tomaram o lugar anteriormente ocupado pelos transistores. Estes minúsculos circuitos eram ainda mais rápidos e seguros que os transistores. Os circuitos integrados ou IC's eram muito pequenos ocupando consequentemente muito pouco espaço.

Os computadores de primeira e segunda geração tinham sido máquinas grandes que assentes no pavimento ocupavam um volume grande. Mas os computadores da terceira geração eram muito mais pequenos; alguns podiam

mesmo ser colocados em cima de uma mesa. E a sua velocidade de operação era 100 vezes superior à dos da segunda geração (e 1000 vezes maior que os da primeira geração)!

Os circuitos integrados eram produzidos em série a baixo custo. Assim quantos mais computadores fossem construídos com circuitos integrados mais o preço dos computadores ia baixando. Os computadores da terceira geração eram suficientemente baratos e pequenos para poderem ser adquiridos por milhares de empresas em todo o mundo.

Produção em série significa «fabricação em quantidades muito grandes».

tente responder

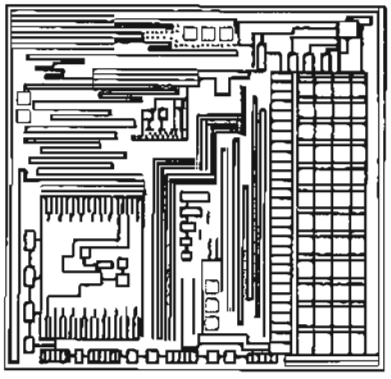
1. Que dispositivos eram usados para conduzir electricidade nos computadores da terceira geração?
2. Que vantagens tinham estes dispositivos sobre os transistores?

Os computadores da Quarta Geração

Em meados dos anos setenta os cientistas desenvolveram um método de colocar milhares de circuitos integrados num único e fino *chip* ou pastilha de silício. O *chip*, em si mesmo, é tão pequeno que alguns podem até passar pelo buraco de uma agulha, pelo que é difícil imaginar tantos circuitos numa

Recorde que no capítulo 7 já se disse que a RAM, a ROM e a CPU são ***chips* de circuitos integrados.**

Tamanho real de um *chip*



Chip ampliado

superfície tão minúscula. Foram necessários instrumentos muito delicados e técnicas especiais para criar o «miraculoso» *chip de circuitos integrados.*

Os computadores construídos com estes *chips* podem fazer cerca de 10

Os *chips* de circuitos integrados levaram ao desenvolvimento dos microcomputadores.

milhões de cálculos em 1 segundo. São 10 vezes mais rápidos que os da terceira geração ou seja 1000 vezes mais rápidos que os da segunda e 10 000 vezes mais rápidos que os da primeira geração!

1.ª Geração	2.ª Geração	3.ª Geração	4.ª Geração
Válvulas	Transistores	Circuitos integrados	*Chips* de circuitos integrados
1000 cálculos por segundo	10 000 cálculos por segundo	1 000 000 cálculos por segundo	10 000 000 cálculos por segundo

Como se vê, cada geração de computadores é caracterizada pelo emprego de uma nova invenção como dispositivo de condução de electricidade através do computador. Porque estes novos dispositivos eram sucessivamente mais pequenos, os computadores também diminuíam de tamanho, tornando-se mais potentes que os primitivos e enormes computadores. E o seu preço diminuiu de tal forma que permitiu a aquisição de computadores por pequenas empresas, escolas e até pessoas.

tente responder

1. Que dispositivo é usado para conduzir a electricidade nos computadores da quarta geração?
2. Um *chip* de circuitos integrados geralmente encerra quantos circuitos?
   a. 10 a 20 b. milhões c. milhares
3. Porque é que os computadores da quarta geração são usados por muitas empresas?

Os computadores das próximas gerações

Como serão as próximas gerações de computadores? Os computadores responderão à voz humana, ao contrário do que sucede com as vulgares unidades de entrada que são os teclados, as unidades de discos e os gravadores de cassettes. Já há computadores aptos a «compreender» um pequeno vocabulário falado e alguns outros podem «falar» certos sons através de um sintetizador de voz. As melhorias que estão a ser introduzidas nestas unidades de saída são de tal forma que os computadores ficarão aptos a «falar» numa voz quase humana.

Um **sintetizador de voz** produz o som de uma voz.

Os cientistas já estão a tentar colocar milhões de circuitos num único *chip*. Isto tornará os computadores ainda mais pequenos e mais rápidos do que são agora. Já se vendem pequenos computadores de bolso. Virá o dia em que computadores mais potentes que os actuais serão tão comuns e tão pequenos como as pequenas calculadoras.

Passaram milhares de anos desde o tempo em que as pessoas contavam

pelos seus dedos até ao momento em que foi construído o primeiro computador electromecânico. Mas demoraram apenas trinta anos desde a invenção do enorme, dispendioso e «lento» ENIAC até aos pequenos, baratos e rápidos microcomputadores. Qual a causa das rápidas mudanças destes trinta anos?

As novas invenções geralmente permitem a exploração de novas áreas e fazem aparecer outras novas invenções. Assim aconteceu com os computadores. Já se empregam mesmo computadores para projectar novos computadores: logo que construído, um novo e melhor computador pode ser usado para ajudar a projectar um ainda melhor. Que significam para nós computadores mais pequenos, mais potentes e mais baratos? Talvez que algum dia haja um computador em cada casa ou na pasta de cada estudante!

1. Porque é que os aperfeiçoamentos nos computadores têm lugar a uma velocidade cada vez maior?

2. Faça corresponder ao dispositivo eléctrico indicado a respectiva geração de computadores em que foi utilizado:

a. circuito integrado	1. primeira geração
b. válvula	2. segunda geração
c. *chip* de circuitos integrados	3. terceira geração
d. transistor	4. quarta geração

3. À medida que cada geração de computadores foi aparecendo eles tornaram-se:
   a. maiores e mais caros
   b. mais pequenos e menos potentes
   c. mais pequenos e mais baratos

4. Se os computadores fossem suficientemente pequenos e baratos para ter um e o levar para a escola, em que o utilizaria?

Capítulo 10

O COMPUTADOR COMO FERRAMENTA DE TRABALHO

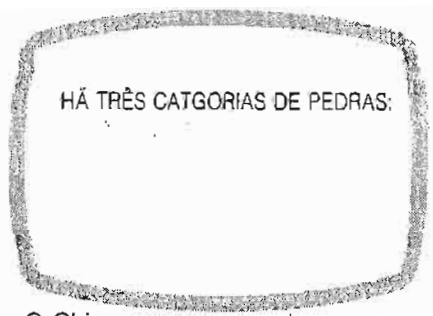
Durante muitos anos homens e mulheres dedicaram-se — e ainda se dedicam — a trabalhar no sentido de tornar os computadores cada vez mais úteis. Os computadores são como ferramentas — utensílios que servem para ajudar as pessoas. Neste capítulo iremos ter uma visão dos diversos tipos de programas de computador (ou *software*) concebidos para tornar o computador um utensílio muito útil em casa, na escola e no trabalho.

Processamento de texto

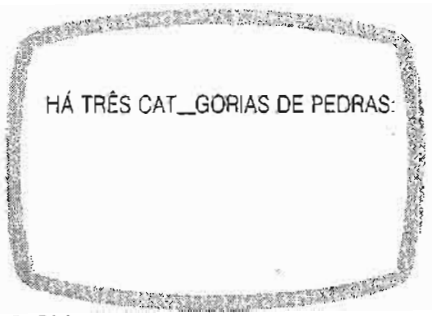
Qualquer aluno ocupa sempre muito do seu tempo a escrever trabalhos para a escola. Se tem de preparar uma boa redacção provavelmente escreve-a — e reescreve-a — até conseguir o texto que melhor o satisfaz. Depois o professor corrige o exercício e devolve-o com anotações para emendar a ortografia e erros de gramática. Haverá mesmo algumas vezes em que o texto terá de voltar a ser todo escrito de novo. É frequente estes esforços de escrita ocuparem muito tempo e serem acompanhados de muita frustração.

Vejamos agora o que fará um estudante que usa um microcomputador e um programa de **processamento de texto** para escrever o seu trabalho.

O Chico está a fazer um trabalho de Ciências e precisa de escrever um relatório sobre a sua colecção de rochas. Carrega no seu computador um programa de processamento de texto e em seguida escreve o relatório no teclado. As palavras, em vez de aparecerem no papel, vão aparecendo no *écran* do computador; se o Chico notar um erro de dactilografia move o cursor para o ponto do *écran* onde ele está e emenda-o escrevendo por cima; se omitiu uma letra dá instruções ao computador para a inserir no local correcto, procedendo o computador automaticamente à «abertura» do espaço necessário para a letra; e o Chico pode igualmente dar instruções ao computador quer para criar espaço para uma palavra, uma frase ou para tanto texto quanto desejar introduzir em determinado ponto. Se quiser apagar partes do texto poderá apagar tanto quanto queira e neste caso o computador automaticamente «elimina» o espaço deixado vazio.



O Chico esqueceu-se do «e» em «categorias».



O Chico moveu o cursor para o local onde falta o «e» e carrega na tecla que significa *inserir*. O computador abre um espaço para o «e» e move todas as outras letras um espaço em frente.

Lembre-se que *Software* se refere aos programas geralmente armazenados em disco ou fita magnética.

A operação de transferir o programa armazenado em disco para a memória de acesso aleatório do computador» chama-se fazer o **boot up** do disco.

Recorde-se que o cursor é um indicador que assinala no *écran* onde será escrito o carácter seguinte.

Inserir significa «pôr em».

Apagar significa «tirar, limpar».

Quando num *écran* de computador não podem ser apresentadas todas as linhas de um relatório, o texto no écran pode «desfilar» (em inglês *scroll*), isto é, subir ou descer de forma a que se tenha acesso a qualquer parte do texto.

Realçar uma palavra significa «evidenciá-la» geralmente apresentando-a em caracteres mais carregados ou em «invertido» (branco sobre preto em vez de preto sobre branco, por exemplo).

Os tipos de impressão podem dizer respeito às diferentes dimensões de um tipo de letra, à utilização de caracteres itálicos, negros e outros. Algumas impressoras dispõem de mais tipos de impressão do que outras.

O Chico pode não conseguir acabar o trabalho, apenas o fazendo mais tarde. Se desligar o computador todo o trabalho armazenado na RAM se perderá. Assim, e antes de o desligar tem de dar instruções ao computador para guardar *(save)* o seu trabalho num disco. Para tal ele não vai usar o disco que contém o programa de processamento de texto, mas sim um outro, preparado para gravar informações ou dados, a que se chama um *disco de dados*. Em rigor ele até deverá armazenar o seu trabalho em dois discos de dados separados, servindo o segundo disco de *backup* (reserva, auxiliar) apenas usado se o primeiro acidentalmente se estragar ou danificar.

Quando quiser recomeçar o seu trabalho volta a carregar no computador o programa de processamento de texto dando-lhe instruções para carregar o trabalho gravado no disco de dados. Agora pode continuar o seu trabalho. O Chico acaba de dactilografar o relatório e junta alguns pormenores finais «dizendo» ao computador para centrar o título e numerar cada uma das páginas. Se o programa de tratamento de texto contiver um programa de «dicionário», pode também dar instruções ao computador para verificar cada uma das palavras do relatório e ver se elas constam do dicionário: se o computador não encontrar no «dicionário» alguma das palavras, realçá-la-á e o Chico poderá verificar o que está mal escrito. (É evidente que se se enganar escrevendo oca em vez de boca o computador não realçará essa palavra, uma vez que oca também existe no dicionário.)

Agora tudo está pronto para se imprimir o relatório. Liga a impressora e dá instruções ao computador para imprimir. Estas instruções incluem as respeitantes aos espaços entre linhas (um ou dois espaços) e, nalguns programas e nalgumas impressoras, à escolha do tipo de letra de impressão. O relatório é impecavelmente impresso, pronto a ser entregue ao seu professor. Antes de desligar o computador, o Chico grava novamente o trabalho nos seus discos de dados.

Depois de ler o relatório o professor devolve-o ao Chico e sugere-lhe um novo texto para o parágrafo introdutório. Com um programa de processamento de texto este tipo de correcção é fácil: tudo o que há a fazer é utilizar o computador e introduzir as alterações sugeridas.

Os estudantes gostam de usar programas de processamento de texto porque assim podem corrigir facilmente erros. Os professores, por sua vez, também vêem neles vantagens pois sabem que assim os alunos podem empregar melhor o seu tempo concentrando-se em encontrar um melhor estilo de escrita em vez de terem de reescrever páginas e páginas de trabalho. E ambos, estudantes e professores, apreciam as páginas produzidas por um programa de tratamento de texto com uma impressão clara e sem erros.

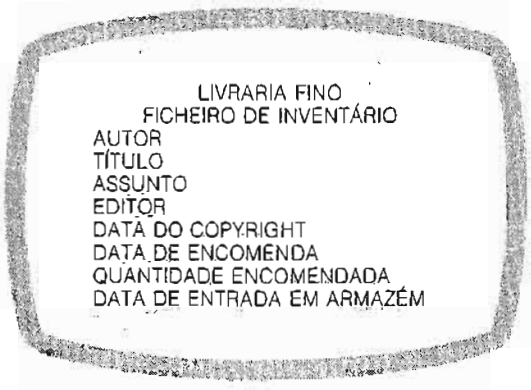
O processamento de texto é fácil de usar e muito popular em casa, nos escritórios e também nas escolas.

Gestão de bases de dados

A Livraria Fino é um local muito popular. Os clientes costumam perguntar ao senhor Fino se tem determinado livro na sua loja: às vezes conhecem o título do livro, mas não o autor; outras vezes sabem o nome do autor, mas não o título; e outras vezes ainda, não conhecem nem o título, nem o autor, mas apenas o tema geral do livro. Enquanto a livraria foi pequena, o senhor Fino não tinha dificuldade em encontrar os livros que os seus clientes desejavam, mas à medida que a livraria cresceu, deixou de poder lembrar-se de todos os livros

que tinha em armazém; passou então a recorrer a fichas para encontrar as informações que pretendia. Mas como no ficheiro os livros estavam classificados pelo nome dos autores, quando era apenas dado o título ou o assunto a sua utilidade não era grande. Foi então que o senhor Fino se decidiu a comprar um **computador** e um programa de **gestão de bases de dados** para o ajudar.

Um programa de gestão de base de dados é um programa que organiza a informação. O senhor Fino começou por decidir o tipo de informações que necessitava, utilizando o programa de gestão de bases para dados para construir um ficheiro de registo da informação. Para delinear o ficheiro, o senhor Fino começou por escrever as várias informações que desejava. Eis como surgiu no *écran* do computador o ficheiro que criou.



Agora, sempre que compra novos livros o senhor Fino regista no seu ficheiro as informações apropriadas. Eis um exemplo de uma ficha que teria podido juntar ao seu ficheiro depois de receber uma encomenda:

Chama-se um registo ou ficha a cada novo conjunto de informações adicionado ao ficheiro. Os ficheiros podem ter centenas de milhar de registos.

Desta forma o computador foi usado como um instrumento auxiliar para guardar informações. Antes o senhor Fino fazia estes registos com papel e lápis; agora usa a capacidade de classificação de informações do computador para, de uma forma rápida e precisa, fazer trabalhos que sozinho não poderia fazer.

Um cliente pergunta agora os nomes de alguns livros de um determinado assunto. O senhor Fino escreve o assunto e o computador faz aparecer no *écran* todos os livros sobre esse assunto que tem registados no ficheiro.

Um cliente pergunta por determinado título de um livro de que não sabe o

Lembre-se que **dados** significa «informação».

Uma **base de dados** é uma reserva, um fundo de informações.

Cada vez que se introduz no computador uma nova informação, aparecem no *écran* apenas as categorias que constituem o ficheiro.

Este **ficheiro** chama-se INVENTÁRIO

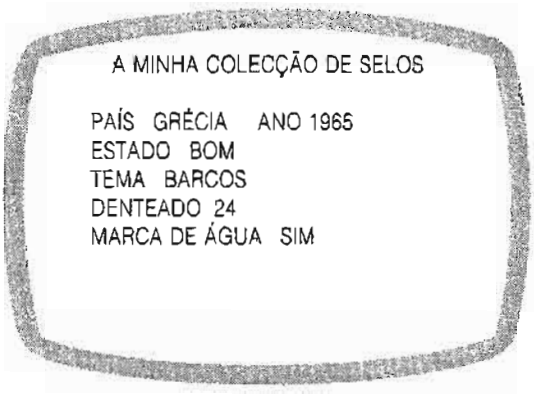
autor. O senhor Fino escreve o título e o computador fornece o nome do autor e, se forem desejadas, também outras informações sobre o livro.

O senhor Fino deseja saber quanto gastou na compra de novos livros durante o mês de Abril. Introduz a informação que deseja e o computador fornece o *valor total*.

A vantagem do programa de gestão de bases de dados é poder procurar informações segundo várias categorias.

Lembre-se que antes de utilizar o computador o senhor Fino apenas podia pesquisar as informações por autores.

Os programas de gestão de bases de dados são usados para armazenar, fornecer e recuperar todos os tipos de informação. A Maria, por exemplo, usa um destes programas para ir registando a sua colecção de selos: com o computador fica a saber rapidamente quantos selos gregos tem, o número de selos emitidos antes de 1930 ou até o número dos seus registos.



O dr. Benjamim usa um programa de gestão de bases de dados para listar os seus doentes com sintomas similares podendo assim comparar os tratamentos ou listar todos os efeitos secundários que tiveram com um determinado medicamento.

A maior parte das pessoas que registam informações de qualquer tipo consideram muito úteis os programas de gestão de bases de dados porque são rápidos, precisos e permitem «procurar» o informação usando vários critérios.

Folhas de cálculo

Um orçamento é um plano de receitas e de despesas.

A senhora Silva, directora de um colégio, é responsável pela elaboração do respectivo orçamento, tarefa em que todos os anos gasta semanas e semanas, pois tem de ter em consideração todos os materiais que os professores requisitam, os materiais de que os vigilantes também carecem, e o custo do aquecimento, dos telefones e da electricidade. Quando os valores daquilo que poderia gastar são superiores ao disponível, tem que fazer cortes nalgumas das rubricas. Se durante o ano ocasionalmente conseguir alguns fundos-extra pode então comprar materiais inicialmente não previstos. Mas sempre que pretende fazer uma alteração no orçamento, tem que gastar o lápis, quantidades de papel, uma boa borracha e utilizar uma calculadora.

No ano em que a senhora Silva fez o orçamento no computador da sua escola com uma **folha de cálculo** não precisou nem de lápis, nem de papel, nem de borracha, nem de calculadora e em vez de várias semanas gastou apenas algumas horas para o elaborar. Uma folha de cálculo é um programa

que se apresenta no *écran* do computador como uma tabela, constituída por linhas e colunas. A intersecção de cada linha com cada coluna constitui uma «célula», onde pode ser colocado determinado tipo de informação. Eis uma parte da folha de cálculo com algumas células preenchidas pela senhora Silva.

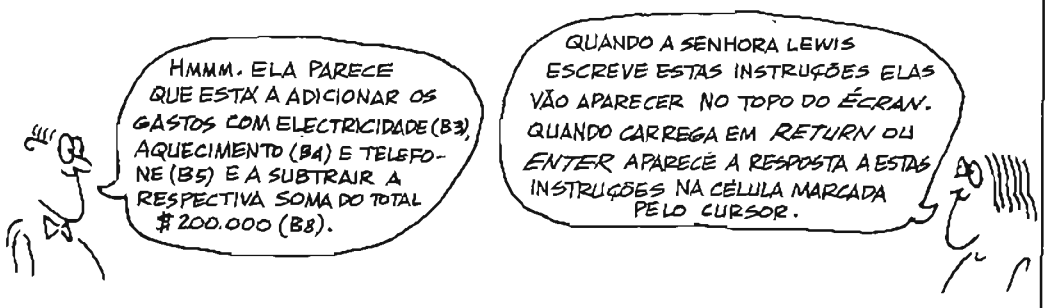
B5 indica a célula da coluna B, linha 5. Pode encontrar esta célula movendo o seu dedo sobre a coluna B, até ao seu cruzamento com a linha 5.

	A	B	C	D
1	RUBRICAS	VALORES		
2	----------	----------		
3	ELECTRICIDADE	6 000$00		
4	AQUECIMENTO	12 000$00		
5	TELEFONE	3 000$00		
6	MATERIAL DE ENSINO			
7		----------		
8	TOTAL	200 000$00		
9				

A senhora Silva introduziu na coluna A as várias rubricas do orçamento. Os valores relativos à electricidade, aquecimento e telefones são os valores que sabe que vai gastar e que não podem ser alterados. O valor de 200 000$00 na célula B8 é a quantidade total de dinheiro de que a senhora Silva dispõe e pode gastar. Na célula B6 ela escreveria:

Quando se escreve esta fórmula, ela vai aparecer no canto superior esquerdo do *écran*. Pressionando-se as teclas ENTER ou RETURN o resultado da aplicação da fórmula é deslocado para o ponto indicado pelo cursor.

```
B8-(B3+B4+B5)
```



A senhora Silva deu instruções ao computador para adicionar os valores de B3 a B5 e subtrair a respectiva soma de B8. Desta forma sabe exactamente o dinheiro que pode gastar em material de ensino.

E — pensa a senhora Silva — se pudessemos cortar os custos de aquecimento baixando os nossos termostatos no inverno? Ao modificar o valor da célula B3 (à frente de ELECTRICIDADE), o valor da célula B6 (MATERIAL DE ENSINO) automaticamente altera-se.

	A	B	C	D
1	RUBRICAS	VALORES		
2	----------	----------		
3	ELECTRICIDADE	5 000$00		
4	AQUECIMENTO	12 000$00		
5	TELEFONE	3 000$00		
6	MATERIAL DE ENSINO	180 000$00		
7		----------		
8	TOTAL	200 000$00		
9				

A senhora Silva pode fazer os ensaios que quiser alterando uma célula e observando como as outras mudam em consequência da primeira alteração.

Quando tiver chegado a um valor para o «material de ensino», a senhora Silva pode proceder ao preenchimento de uma nova folha, similar, listando todos os materiais de que precisa e obedecendo à condição do valor total ser aquele a que chegou na célula B6 da primeira folha. Sempre que fizer uma alteração numa das rubricas todas as outras mudarão automaticamente ajustando-se ao orçamento. Ela pode perguntar «E se?...» quantas vezes desejar e determinar em segundos a forma como isso afectaria o orçamento.

Quase todas as pessoas que costumam perguntar «E se?...» podem substituir o lápis, o papel, a calculadora e a borracha por uma folha de cálculo. Com estes programas, os presidentes de empresas e os gestores planeiam os seus orçamentos, as famílias prevêem as despesas domésticas, os cientistas demonstram relações entre fórmulas e os estudantes de matemática observam como alterações nas variáveis de um problema modificam o respectivo resultado.

Variável é o que se pode alterar, que pode variar.

Software integrado

Vimos três tipos de *software* ou programas vulgarmente usados em empresas, escolas e em casa. Há *software* que combina dois ou três destes programas (ou de outros) num programa ainda maior. A este tipo de *software* chama-se **software integrado**. Por exemplo um *package* de *software* integrado que inclua tratamento de texto, gestão de bases de dados e folha de cálculo torna possível a uma escola gerir os ficheiros dos seus alunos (gestão de bases de dados), registar os resultados de testes (folha de cálculo) e emitir cartas às famílias comunicando as notas dos alunos (tratamento de texto).

Integrado significa «ter partes diferentes combinadas num conjunto».

Para os utilizadores dos computadores o *software* integrado poupa muito tempo e elimina uma série de papéis.

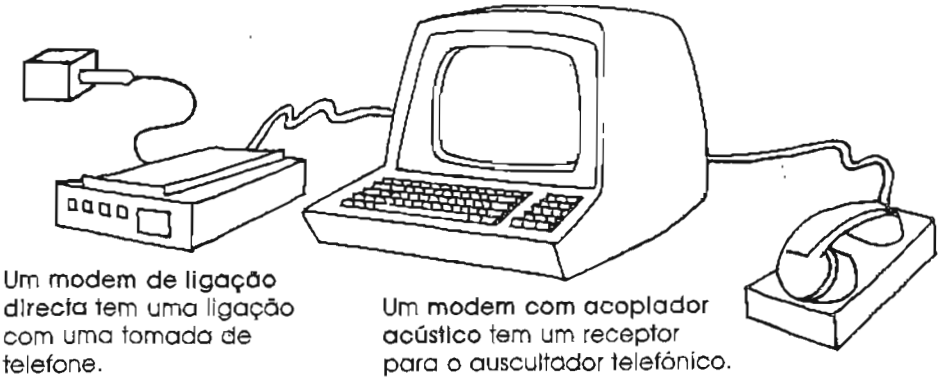
A comunicação com outros computadores

Embora um computador seja uma poderosa ferramenta, a sua utilidade é função das informações que lhe forem dadas. Todas as informações armazenadas num computador constituem a sua base de dados. Há algumas empresas cujos computadores têm enormes bases de dados a que podem aceder pessoas do exterior da empresa. Para tal apenas é necessário um computador, um disco com *software* de comunicações e um *modem*, dispositivo que liga um computador com um telefone.

Aceder significa «ter acesso».

Software de comunicações é um programa que dá instruções a um computador para comunicar com outro computador.

O termo **modem** deriva de «modulador--desmodulador». Um **modulador** converte os sinais do computador em sinais telefónicos; o **desmodulador** faz a conversão inversa.



Um modem de ligação directa tem uma ligação com uma tomada de telefone.

Um modem com acoplador acústico tem um receptor para o auscultador telefónico.

Log on significa «fazer uma ligação».

O número de identificação (ID) e a palavra-senha *(password)* para acesso à base de dados são recebidos através do correio.

É evidente que se a chamada telefónica para a empresa da base de dados for uma chamada interurbana (ou internacional) terá de pagar mais essa tarifa.

Há *software* de comunicações que permite gravar as informações em disco, mas há outro que não.

O operador do sistema é conhecido como o **sysop.**

O **menú** funciona como o índice do programa.

Para se fazer a ligação a uma base de dados, deve-se:

- ligar o computador a um *modem* e um telefone;
- introduzir o *software* de comunicações no computador;
- marcar o número de telefone da base de dados;
- marcar no teclado o número de identificação pessoal do utilizador e a palavra de senha *(password)*.

A empresa da base de dados debitá-lo-á por cada minuto que o seu computador esteve ligado à base de dados.

Quando se liga a uma base de dados, procuram-se informações. Algumas bases de dados têm um tipo de informação muito específica: por exemplo, uma base de dados médica tem informações sobre doenças, medicamentos e tratamentos e a base de dados de uma Bolsa tem as últimas informações e notícias sobre as cotações de acções e obrigações. Há ainda bases de dados de «informações gerais» que contêm informações sobre os mais diversos assuntos como os preços de carros estrangeiros, sugestões de jardinagem, e até o conteúdo de toda uma enciclopédia. Para encontrar a informação que deseja, tem de se consultar o índice da base de dados e indicar o tópico que se quer ver. Uma vez encontrado o que se deseja, tem-se a alternativa de dar instruções ao computador para guardar a informação em disco ou de a imprimir na sua impressora.

Outro meio de comunicação com outros computadores é o sistema BBS (bulletin board system), um serviço de mensagens e informações que é geralmente oferecido por uma pessoa a partir de sua casa. A maior parte dos BBS são gratuitos: liga-se simplesmente e comunica-se com o operador do sistema. A utilização do BBS tem muitas finalidades: procurar ajuda para um problema de programação, anunciar uma reunião local sobre computadores, ou simplesmente deixar uma mensagem para outra pessoa que a «buscará» quando por sua vez ligar ao BBS.

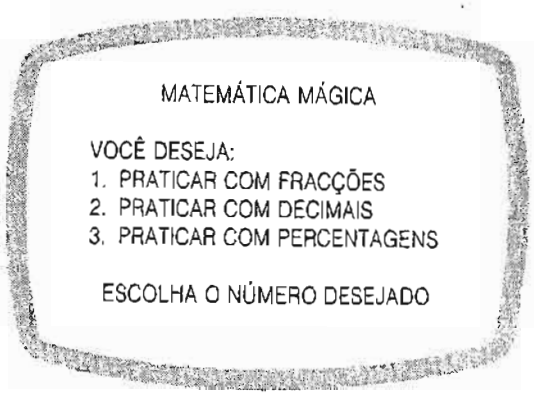
Enfim, com um *modem,* um telefone e *software* de comunicações o mundo abrir-se-á aos utilizadores dos computadores.

A escolha do *software* adequado

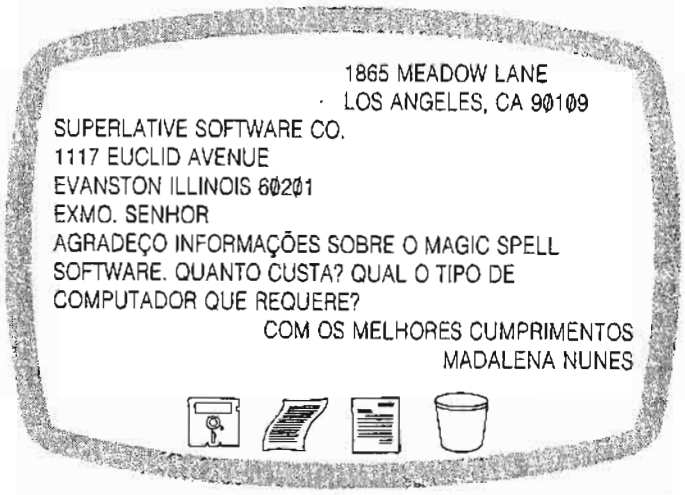
Muitas empresas vendem *software* para processamento de texto, gestão de bases de dados, folhas de cálculo e muitos outros usos. Há milhares de programas à escolha. Como escolher o melhor *software*? É impossível examinar cada programa e por isso precisamos de ter algumas directrizes.

Primeiro o programa deve fazer exactamente o que precisamos que faça. Se precisa de um programa para o ajudar a praticar multiplicações não compre um programa que o exercite em somas e subtrações, por muito bom que seja. Há programas que fazem mais do que aquilo que necessitamos e, geralmente, quanto mais um programa faz, mais caro é. Assim pode não compensar a compra de um programa que o exercite em multipilcações e também em fracções, decimais e percentagens.

Em segundo lugar o programa deve ser fácil de usar e ter instruções claras. Muitos programas têm um **menú** para ajudar o utilizador a seguir para as várias partes do programa.



Outros programas em vez de menú usam desenhos ou imagens. O utilizador move o cursor para a imagem que representa o trabalho que deseja ser feito.



Neste programa de processamento de texto, o utilizador move o cursor para a figura que representa o trabalho que vai ser feito: o disco, para armazenar o texto em disco; o papel para imprimir o texto; a página dactilografada para alterar as marginações; ou o cesto de papéis para eliminar o texto da memória do computador.

O movimento do cursor no écran é feito pressionando as teclas de setas existentes no teclado ou usando um **rato** *(mouse)*, dispositivo móvel ligado ao teclado, que movido sobre uma superfície plana faz o cursor mover-se na mesma direcção.

Um bom programa deve poder ser usado por qualquer pessoa que saiba ler, não sendo necessário que o utilizador seja um programador. Na realidade, a maior parte das pessoas que usam *software* nunca viram as instruções do programa. Os programas que são fáceis de usar são chamados **amigos do utilizador** *(user-friendly)*.

Um **bug** é um erro no programa.

Em terceiro lugar o programa não deve ter defeitos ou erros *(bugs)* e mesmo que o utilizador pressione teclas erradas ele não deve deixar de correr normalmente.

Uma vez seleccionado o programa que pensa poder responder às suas necessidades, há algumas coisas que deve fazer para ver se é o mais adequado. Muitas lojas de computadores permitem aos clientes utilizarem o programa na loja antes de o comprarem; outras permitem a devolução do *software* dentro de um certo prazo se não ficar satisfeito. Uma boa ideia é verificar a opinião dos seus amigos e ver se estão satisfeitos com o seu *software;* as opiniões deles podem ser uma das melhores directrizes na compra de *software*.

1. Indique duas profissões em que o processamento de texto seria útil. Explique porquê.

2. Quais das afirmações seguintes são verdadeiras acerca dos programas de gestão de bases de dados?

   a. Pode procurar informações segundo várias categorias.

   b. Estes programas são usados apenas para armazenar informações.

   c. Estes programas não foram projectados para uso doméstico.

   d. As pessoas usam estes programas para armazenar, classificar e recuperar informação.

3. Como é que uma folha de cálculo responde à questão «E se...?»

4. Qual o tipo de *software* que combina dois ou três programas diferentes num programa mais completo?

5. Em que medida acha que o BBS *(bulletin board system)* poderá vir a ser-lhe útil?

6. O que deve verificar sempre quando está a escolher *software*?

Capítulo 11 OS COMPUTADORES ESTÃO EM TODO O LADO

Os computadores afectam as nossas vidas todos os dias e de muitas formas. Como? Onde são usados? Vamos ver.

Os computadores na administração pública

O governo dos Estados Unidos foi um dos primeiros utilizadores dos computadores, durante a II Guerra Mundial e com a finalidade de decifrar os códigos militares secretos do inimigo. Hoje, as forças militares utilizam computadores em tarefas como o rastreio e a orientação de aviões, barcos oceânicos e tanques e no planeamento de estratégias de defesa, para além de também os utilizarem para registar os postos, tarefas, vencimentos e outras informações importantes relativas aos militares.

Lembra-se que o Serviço de Recenseamento usou a Máquina de Tabulação e os cartões perfurados de Hollerith?

No Serviço de Recenseamento, que é um departamento governamental, utilizam-se os computadores na ordenação das informações recolhidas de dez em dez anos sobre os habitantes dos Estados Unidos. No censo de 1980 obtiveram-se mais de 3 biliões de respostas: o tratamento manual de todos aqueles dados ocuparia milhares de pessoas durante mais de 10 anos; os computadores fizeram este trabalho em menos de um ano.

Centenas de outros departamentos empregam os computadores noutras espécies de trabalho.

No Serviço de Contribuições, um organismo governamental cuja missão é cobrar os impostos, os computadores registam os formulários das declarações dos vários contribuintes, verificam se têm erros, emitem cheques destinados a pessoas que têm de ser reembolsadas e imprimem os avisos para as pessoas que ainda têm impostos em débito.

A administração regional e local também confia aos computadores determinadas tarefas: por exemplo, os registos de propriedade de todos os veículos (automóveis, camiões e outros), tarefa que nos Estados Unidos é função de cada Estado, e o comando dos semáforos de uma cidade.

tente responder

1. Diga se um computador o poderia ou não ajudar a executar melhor a tarefa indicada:
   a. determinar o número de eleitores inscritos nos respectivos cadernos.
   b. escolher um novo chefe dos bombeiros.
   c. processar as ordens de pagamento para todas as pessoas que trabalham numa Câmara Municipal.
   d. cuidar do campo de *baseball* do parque da cidade.

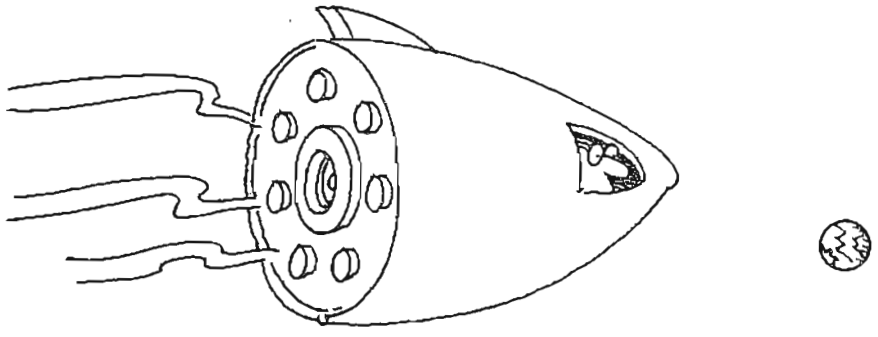
2. Para que pensa que o departamento da polícia local poderia usar um computador?

3. Um dos trabalhos do Serviço de Recenseamento é determinar a população de cada cidade, concelho, estado e país. Faça corresponder a cada uma das fases do trabalho a parte do computador que o executa:

a. As informações são recolhidas e introduzidas no computador.	1. Memória
b. As informações são armazenadas no computador para serem usadas quando necessário.	2. CPU
c. Somam-se os números respeitantes à população.	3. Unidade de Entrada
d. Imprimem-se os resultados.	4. Unidade de Saída

Os computadores no programa espacial

Se não fossem os computadores as naves espaciais não conseguiriam ser lançadas. O seu emprego nos programas espaciais remonta ao lançamento da primeira nave espacial em 1959 sendo usados no planeamento da trajectória das naves, para as manter na rota fixada e para planear a respectiva aterragem.



Foi equipamento informático que detectou a fenda no tanque de combustível.

Os computadores provaram ser muito importantes em emergências. Há alguns anos foi encontrada uma fenda no tanque de oxigénio do módulo de comando da Apolo 13. Havia apenas uma pequena quantidade de oxigénio para utilização de emergência e era importante conseguir fazer regressar os astronautas à Terra o mais depressa possível. Para isso, cientistas, programadores e operadores de computador do Centro de Naves Espaciais Tripuladas de Houston trabalharam sem parar: forneceram aos computadores as informações necessárias para planear uma nova trajectória de vôo. Em cada nova hipótese o computador fornecia dados importantes como a duração do vôo de regresso, as quantidades de combustível e de oxigénio que seriam usadas, o momento e o local de amaragem. Tinham de ser considerados milhares de factores e não havia tempo suficiente para o *staff* do centro de controlo proceder a todos os cálculos. Pode dizer-se que, graças à utilização de computadores, foi possível planear uma nova rota de regresso da Apolo 13 e os astronautas foram trazidos em segurança até à Terra.

tente responder

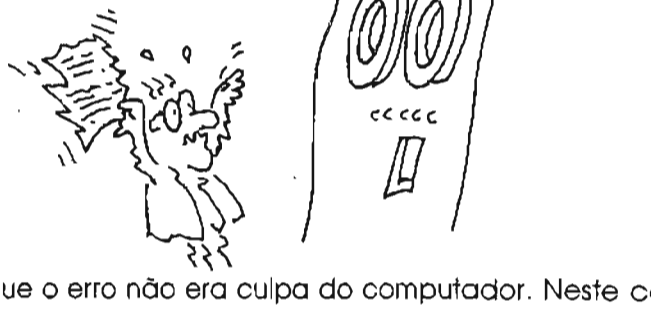
1. Porque é que no planeamento de um vôo espacial é preferível usar computadores em vez de pessoas para fazer determinados trabalhos?
   a. O computador sem instruções do homem conhece as velocidades, a trajectória e o ponto de aterragem.
   b. O computador pode fazer os cálculos mais depressa e com maior precisão.
   c. O computador pode dizer qual o dia em que haverá melhores condições meteorológicas para o lançamento.

Os computadores nos escritórios

Observe uma factura de uma grande empresa comercial. Foi escrita à mão ou impressa por uma máquina? Nas grandes empresas os computadores registam o movimento de vendas e os pagamentos de clientes nas chamadas contas correntes e emitem os respectivos extractos. Seriam necessárias muitas pessoas para fazerem o trabalho de um computador. Mas não é só a questão da rapidez, é também a da precisão: os cálculos dos computadores são mais exactos que os do homem.

Significa isto que as facturas emitidas por computador nunca têm erros? Não! Por vezes causam até alguns problemas desconcertantes.

O senhor Rosas, por exemplo, recebeu de uma grande casa comercial uma factura no valor de 0$00 escudos, que ele, consequentemente, ignorou. No mês seguinte recebeu um aviso «amigável», impresso por computador, lembrando de ainda devia 0$00 escudos. Escreveu então à empresa uma carta pedindo para corrigirem o erro, mas antes do seu pedido ser satisfeito recebeu um novo aviso, também impresso por computador, informando-o de que se não pagasse a factura lhe seria cortado o crédito. Enquanto o seu problema não foi finalmente resolvido o senhor Rosas estava muito furioso... com o computador!



Claro que o erro não era culpa do computador. Neste caso tratava-se de um erro de programação, pois o programador não tinha instruído o computador no sentido de não enviar facturas quando o saldo das contas fosse zero. Mas era fácil para o senhor Rosas responsabilizar o computador.

Os «erros» dos computadores nem sempre são culpa do programador. Alguns produzem-se quando alguém introduz acidentalmente uma informação errada no computador: imagine que um cliente envia um cheque de 1 050$00 mas que o operador digita 105$00: no próximo extracto o saldo estará errado!

Outras vezes são os próprios computadores que provocam erros em consequência de avarias mecânicas ou eléctricas. Mas estas situações são

tão raras que «os erros dos computadores» são quase sempre «erros humanos».

A maioria das grandes empresas usam computadores em múltiplas tarefas: preparam avisos de pagamento e cuidam dos registos das folhas de pagamentos; mantêm actualizados os inventários e as contas correntes; processam dados que são a base das estimativas do dinheiro de que a empresa pode dispor numa data futura. A razão essencial por que as empresas confiam tanto nos computadores é porque alguns trabalhos podem ser feitos com muito maior rapidez e precisão do que se fossem feitos por homens. Se os computadores assumem tarefas maçadoras e tediosas, os homens ficam libertos para ocuparem o seu tempo e a sua experiência em raciocinar e tomar decisões. Os computadores podem executar trabalhos que não requerem pensar, planificar, raciocinar ou ajuizar e podem fazê-lo 24 horas por dia, todos os dias do ano sem interrupções ou dias para férias — e os computadores nunca se cansam!

Com o emprego de computadores as empresas podem poupar dinheiro. Se esta economia resultar em preços mais baixos para os consumidores, então o computador foi-lhes tão útil como o foi para a empresa. Contudo há um problema na utilização de computadores: é que cada novo computador, por vezes, substitui uma série de pessoas, e é um problema sério sempre que uma máquina toma o lugar de uma pessoa. Os computadores fazem coisas maravilhosas mas lembre-se que da sua utilização há também algumas consequências menos felizes.

Inventário é uma relação das existências de mercadorias num armazém, fábrica ou loja. Uma **conta corrente** é um registo dos bens ou serviços comprados e vendidos.

1. Se fosse gerente de uma casa comercial para qual ou quais das seguintes tarefas quereria utilizar um computador?
   a. Imprimir as facturas dos clientes.
   b. Resolver uma reclamação.
   c. Contratar um novo vendedor.
   d. Registar os resultados das vendas diárias.
2. Suponha que poderia substituir três dos seus empregados por um computador. Indique uma vantagem e uma desvantagem que teria de ter em consideração se adoptasse esta solução.

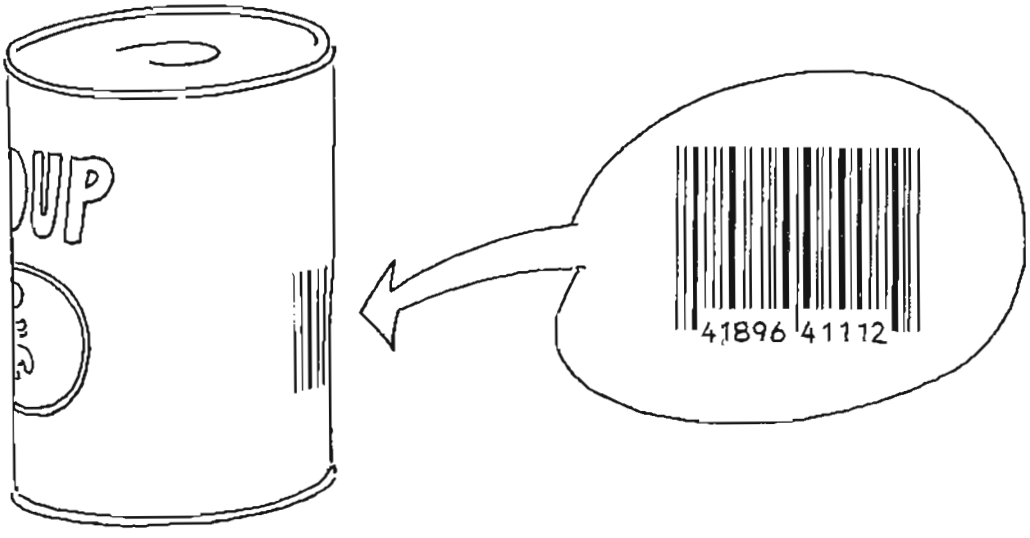
Os computadores nos supermercados

Quando os Costa vão fazer compras ao supermercado não gostam nada de esperar nas bichas da caixa. Muitas vezes as esperas são devidas ao facto de o caixa ter de olhar para o preço marcado na etiqueta de cada unidade adquirida antes de o marcar na caixa registadora. Um dia notaram que as bichas da caixa se moviam mais rapidamente. Quando chegaram à sua vez, perceberam porquê: o caixa já não tinha de procurar

Um **scanner** é um dispositivo de entrada de um sistema de computador.

a etiqueta do preço e registá-lo; passava simplesmente cada unidade por uma «janela» chamada **scanner**, localizada no balcão ao pé de cada caixa e automaticamente aparecia impresso no talão o preço correcto. Como acontecia isto?

O supermercado tinha instalado um sistema de computadores baseado no **Código Universal de Produtos** com que hoje são marcados quase todos os produtos embalados. Este código, o **UPC, CUP** ou **código de barras** é constituído por um conjunto de barras pretas e brancas e de números.



Neste código as primeiras barras e números representam geralmente o nome do fabricante e o produto, e as restantes barras representam as dimensões ou tipo do produto.

Quando um produto marcado com o UPC passa frente ao *scanner* o código é «lido» pelo computador do estabelecimento que, normalmente, nem sequer se encontra próximo da caixa registadora, mas num gabinete noutra parte do estabelecimento, estando ligado às caixas registadoras por cabos colocados sob o pavimento. O computador converte o código no nome do produto, embalagem e preço, enviando estas informações para a caixa registadora que as imprime no talão respectivo; e se algum produto estiver sujeito a impostos, calcula-os. Tudo isto acontece em menos de 1 segundo, pelo que é muito mais rápido do que se o caixa tivesse de digitar cada preço. Quando todos os artigos estão registados, o computador calcula o total, enviando-o também para a caixa registadora em menos de 1 segundo. O sistema UPC torna muito mais rápidas as tarefas relacionadas com o pagamento das mercadorias adquiridas num supermercado.

A memória e o CPU estão no computador.

O talão da caixa registadora é o *output*.

O Código Universal de Produtos poupa muito tempo aos supermercados. Como os produtos embalados já têm um código, os empregados das lojas não têm de colocar etiquetas com o preço em todas as unidades bastando marcá-los em letreiros colocados nas prateleiras. Este sistema permite também melhorias sensíveis nos inventários pois sempre que o computador lê um código UPC altera os dados armazenados na memória, registando uma unidade a menos no produto em causa. Este processo de inventário é muito mais rápido e exacto que o sistema de contagem diária dos artigos deixados nas prateleiras.

Alguns clientes sentem a falta de etiquetas com os preços marcados em cada artigo.

tente responder

1. Como pode o Código Universal de Produtos ajudar um supermercado? Indique duas maneiras.
2. Quais os benefícios que os clientes podem obter da utilização de um UPC num supermercado?
3. Indique alguma coisa que um cliente possa não gostar numa loja com o sistema UPC.

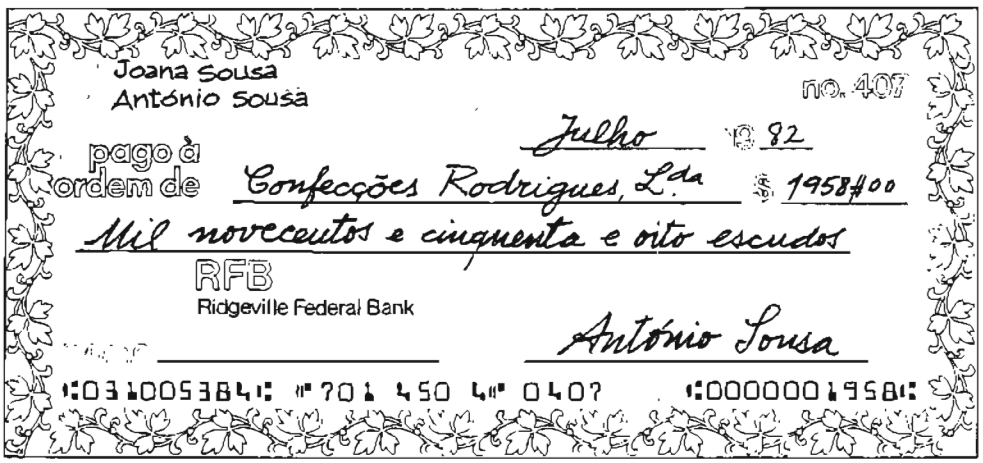
Os computadores nos bancos

A família Sousa decidiu ir ao cinema no domingo. Não tinham dinheiro suficiente em casa e os bancos estavam fechados. Isso não foi problema: no caminho para o cinema pararam num banco, a senhora Sousa introduziu um cartão bancário plastificado na caixa automática de pagamento permanente, marcou o código bancário, carregou num botão e obteve o dinheiro de que precisava. Os bancos estão a instalar por todo o país máquinas de pagamento automático que permitem aos clientes levantar dinheiro, fazer depósitos e pagar contas a qualquer hora do dia ou da noite.

Uma máquina automática de pagamentos dispõe de um teclado especial.

Os bancos empregam ainda computadores para registarem o movimento das contas à ordem ou de poupança dos seus clientes. No capítulo 4 já se viu que os cheques bancários têm impressos os números da respectiva conta em tinta magnética de modo a poderem ser lidos por um dispositivo de entrada chamado leitor de caracteres de tinta magnética.

Um **depósito** é dinheiro que é colocado numa conta.



Quando o banco desconta o cheque, um empregado marca o valor pago também em caracteres magnéticos, no canto inferior direito. No caso representado na figura, o computador do banco «lê» o número da conta dos Sousa e o valor indicado, subtraindo 1958$00 ao saldo disponível. Todos os meses o computador imprime uma lista ou extracto de conta com os cheques pagos, os depósitos efectuados e o saldo, que é enviado pelo correio aos Sousa.

O **saldo** é a quantidade de dinheiro que resta numa conta depois de vários depósitos e levantamentos.

Os computadores usam-se ainda nos bancos para outras tarefas como calcular os juros de empréstimos ou de contas de poupança, conhecer permanentemente o dinheiro disponível para empréstimos e calcular o dinheiro que em cada momento está em poder do banco. São tantas as actividades de um banco que sem computador seria impossível registá-las todas.

Os **juros** são o dinheiro a mais que se tem de pagar quando nos emprestam dinheiro. Por exemplo obtém-se um empréstimo de 100 contos, mas tem de se pagar por ele 115 contos, dos quais 15 contos são os juros. Se você «emprestar» dinheiro ao banco colocando-o numa conta de poupança, o banco pagar-lhe-á a si os juros.

tente responder

1. Suponha que é presidente de um banco. Para quais dos seguintes trabalhos usaria um computador?
   a. Contratar um novo gerente.
   b. Listar todos os empréstimos concedidos a clientes.
   c. Manter os registos de todas as contas à ordem.
   d. Explicar a um cliente o que é uma conta especial de poupança.
2. Por que razão as máquinas automáticas de pagamento são úteis para os clientes dos bancos?

Os computadores nos hospitais

Os computadores que **medem** alterações de condições, tais como as de temperatura ou ritmo cardíaco, são chamados **computadores analógicos.**

Soa um alarme no gabinete da enfermeira. Uma rápida vista de olhos ao *écran* do computador mostra que um doente está com problemas. Estão ligados sensores electrónicos ao tórax do paciente e um computador assinala um batimento irregular do coração fazendo soar um sinal de aviso numa fracção de segundo. A enfermeira chama uma equipa de emergência que, trabalhando rapidamente, consegue restabelecer o ritmo cardíaco normal do doente. Sem o computador passar-se-iam vários minutos antes que a enfermeira soubesse que alguma coisa estava mal. Neste caso foi o sinal de aviso do computador que ajudou a equipa a salvar uma vida.

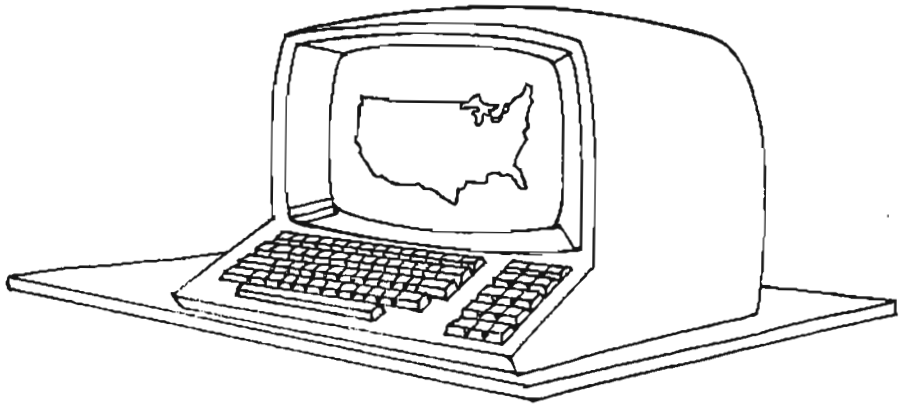
Há muitos locais nos hospitais onde os computadores são usados para acompanharem os doentes: nas salas de operação informam permanentemente sobre a tensão arterial, o ritmo cardíaco e a temperatura e se houver qualquer problema soará um sinal de aviso; as máquinas computorizadas de raios X fornecem imagens de muito pormenor que permitem aos médicos ver coisas que os vulgares raios X não mostravam; além disso os computadores também ajudam na interpretação dos resultados de testes laboratoriais.

Os computadores auxiliam a equipa de um hospital a coligir informações e a interpretar resultados. Será que os computadores alguma vez tomarão o lugar dos médicos, das enfermeiras e do outro pessoal hospitalar? Não. Quando se trata do tratamento de pessoas o julgamento humano será sempre necessário e desejável.

1. Indique uma das aplicações dos computadores nos hospitais.
2. Por que razão os computadores nunca substituirão os médicos e as enfermeiras?

Os computadores nas escolas

Está na hora de Estudos Sociais e a Joana está a estudar a leitura de mapas: vai para o computador da aula e carrega um programa. Aparece no *écran* um mapa com os contornos dos Estados Unidos, representando os seus maiores rios. O computador imprime no *écran* e por debaixo do mapa a seguinte pergunta: QUE RIO ASSINALA A FRONTEIRA ENTRE OS ESTADOS UNIDOS E O MÉXICO? Joana olha para o mapa e escreve: RIO GRANDE. O computador responde: ESTÁ CERTO, JOANA! e em seguida faz outra pergunta. A cadeia de montanhas, os estados e as principais cidades aparecem e desaparecem do *écran* quando necessário.



Este exemplo não é, nem de uma escola especial, nem de uma aula do futuro[2]. Muitas escolas já têm computadores que auxiliam os alunos em muitos temas. À utilização dos computadores no apoio à aprendizagem dos estudantes chama-se **Ensino Assistido por Computador** ou EAC[3].

Ao **Ensino Assistido por Computador** chama-se por vezes educação baseada em computador.

Por que razão se usam computadores no ensino? Os computadores são muito pacientes: não bocejam, não se aborrecem, não se perturbam nem se zangam se se demorar muito tempo a responder à pergunta; e quando se escrever a resposta o computador dirá imediatamente se está correcta ou não, não sendo necessário esperar pelo dia seguinte para saber o resultado.

Suponha que digitou RIO GRAN em vez de RIO GRANDE: o seu professor entenderia logo o que você queria dizer, mas o computador não.

Contudo um computador não pode tomar o papel de um professor. Não explica as coisas de várias maneiras nem se lhe pode fazer *qualquer* pergunta. E também não tem sentimentos: se tivesse de ficar doente em casa uma semana ele não teria pena de si. O computador é uma máquina muito exacta: as suas respostas devem ser extremamente rigorosas ou o computador considerá-las-á erradas. E, como é evidente, um computador não pode realmente sorrir nem dar pancadinhas nas costas.

Os computadores são também usados para ensinar programação. Mas pode haver na escola computadores para outras tarefas: na secretaria, para registar as notas dos alunos, a respectiva assiduidade e também para imprimir os horários e os boletins escolares.

tente responder

1. Que significam as iniciais EAC? O que é?
2. Dos seguintes exercícios indique os que pensa que um aluno poderia fazer no computador da sua sala de aula:
   a. Identificar sujeitos e predicados.
   b. Observar como a luz do sol e a escuridão afectam as plantas.
   c. Praticar multiplicações.
   d. Criar uma escultura de barro.
   e. Memorizar um poema.
   f. Indicar as capitais de vários países.

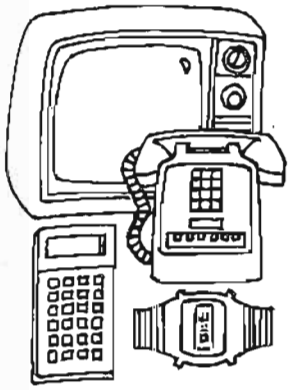
Os computadores em nossa casa

Há quem tenha computadores em casa, usando-os para registar as despesas da família, verificar os extractos das contas bancárias, calcular os impostos e registar datas importantes ou até simples receitas culinárias. Muitas pessoas usam os seus computadores domésticos para jogar ou para aprender a escrever programas.

Um dia os computadores serão tão vulgares nas nossas casas como as televisões. Neles estarão registadas informações como números de telefone, endereços, compromissos e contas de banco. Provavelmente estarão ligados a grandes sistemas localizados fora de casa permitindo a comunicação com estabelecimentos comerciais, bancos e serviços noticiosos. Assim poderá ter no seu *écran* o catálogo dos produtos à venda em determinado estabelecimento e encomendá-los marcando as instruções convenientes no teclado do seu computador; o pagamento será feito por transferência da sua conta bancária para a conta do estabelecimento e tudo isto decidido a partir de sua casa e executado através do computador.

Em sua casa há algum computador? Recorde por um momento que um computador é uma máquina que recebe, armazena, processa e fornece informação; que a memória armazena informação; que a unidade central de processamento a processa; e que a memória e a CPU são *chips* de

Lembre-se que um *chip* de circuito integrado é mais pequeno que a unha de um dedo.



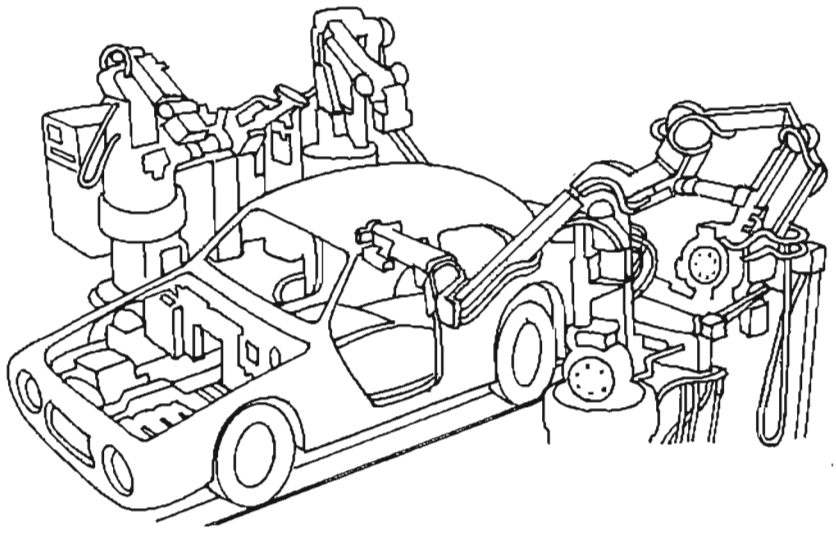
circuitos integrados. Será que tem alguns destes *chips* em sua casa? Provavelmente. Tem um relógio digital? Um jogo electrónico? Um forno de micro-ondas com «memória»? Um novo televisor a cores? Um novo carro? Uma máquina de costura programável? Um telefone de teclas? Uma calculadora? Todos estes artigos dispõem de *chips,* geralmente um *chip* de memória e um *chip* de CPU. Os botões, os mostradores e os interruptores que servem para pôr em funcionamento esses dispositivos são as unidades de entrada; as unidades de saída são os *écrans,* como os que visualizam os números do seu relógio de pulso digital, e também os sinais electrónicos como os da ligação telefónica que faz para um seu amigo noutra cidade.

Provavelmente existem mesmo alguns destes computadores em sua casa. A diferença entre eles e os computadores «pessoais» ou microcomputadores é que fazem parte de outros equipamentos, executando tarefas específicas em vez da vasta gama que os computadores permitem. É então altura de perguntarmos novamente: há algum computador em sua casa?

1. Indique duas finalidades para uma utilização doméstica de um microcomputador.
2. Indique três artigos existentes em sua casa que provavelmente têm *chips* de circuitos integrados.

Os computadores como robots

Numa siderurgia, um braço mecânico entra no interior de um alto forno onde a temperatura é superior a 1000°, e retira um objecto metálico ao rubro. Quem poderia estar junto de um calor tão intenso? Ninguém. Mas o «braço» de aço de um *robot* pode fazê-lo. Nas fábricas, os *robots* executam muitos trabalhos perigosos e podem funcionar em ambientes muito quentes, muito frios ou poluídos.



Um *robot* é um computador que foi preparado para fazer determinadas tarefas. Alguns têm a forma de «braços» com muitos «dedos» conseguindo assim executar tarefas delicadas; outros parecem-se com grandes caixas, ou tartarugas, ou mesmo com homens que se podem mover. Ainda não há muitos anos os únicos *robots* conhecidos eram personagens de ficção dos livros ou dos filmes.

Hoje, os *robots* empregam-se em tarefas aborrecidas — que são maçadoras ou que têm de ser feitas repetidas vezes. Isto permite às pessoas empregarem melhor o seu tempo em tarefas que requerem perícia humana ou inteligência.

Não é só nas fábricas que há *robots*. Também os há em escritórios, escolas e casas. Podem ser programados, por exemplo, para separar correio ou ir buscar jornais. Serão necessários ainda muitos anos até que um *robot* possa fazer tarefas mais complexas como cozinhar, servir uma refeição, lavar os pratos e guardar a loiça. Mas já começaram a assegurar alguns dos nossos trabalhos e certamente o continuarão a fazer ainda mais no futuro.

tente responder

1. Por que razão é uma boa ideia usar *robots* nalguns trabalhos fabris?

2. Quais dos seguintes tipos de trabalho são mais indicados para um *robot* do que para uma pessoa?

   a. mergulhar varas de metal em soluções ácidas.

   b. colocar jarras dentro de uma caixa numa linha de montagem.

   c. escrever um programa de computador.

Capítulo 12

OUTRAS UTILIZAÇÕES — E MÁS UTILIZAÇÕES — DOS COMPUTADORES

Já vimos de que forma os computadores podem ajudar as pessoas na administração pública, nas empresas, nas suas casas e noutros lados. Os computadores armazenam e processam grandes quantidades de informação muito rapidamente. Esta capacidade torna os computadores instrumentos importantes, mas também levanta alguns problemas. Qual é o tipo de informação que se deverá guardar nos ficheiros dos computadores? O que acontecerá se informações secretas ou pessoais cairem em mãos erradas? Neste capítulo iremos ver algumas formas como os computadores podem ser mal utilizados. E verá o que deve ser feito para corrigir tais situações.

Lembre-se que muitos ficheiros de computadores estão em suportes de armazenagem como fitas e discos.

Computadores e privacidade

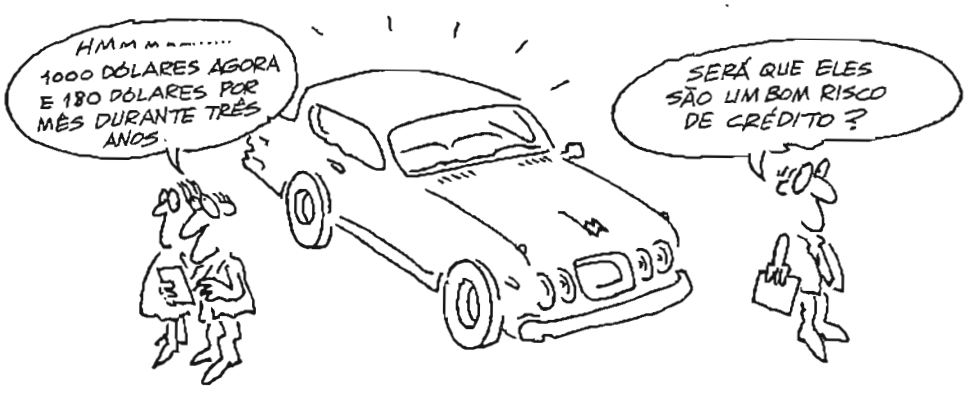
O senhor e a senhora Barros querem comprar um automóvel na Empresa Carros Económicos. Pretendem fazer a compra a crédito, ou seja pagando parte do preço quando levantarem o carro e o restante em pagamentos mensais acrescidos de uma taxa de juro.

Recorde-se que o **juro** é o encargo extra pago por um empréstimo.

A empresa vendedora, no entanto, só pode dar o seu acordo a uma venda deste tipo depois de proceder a uma verificação do crédito dos Barros e de saber se eles podem ou não assegurar os pagamentos. Para responder a uma questão deste tipo, a Empresa Carros Económicos dirige-se a uma **agência de informações** onde se recolhem dados ou informações pessoais que são «vendidos» às empresas que querem verificar o crédito dos seus clientes. Se as informações demonstrarem que os clientes são um bom «risco», então a Empresa Carros Económicos vender-lhes-á o carro a crédito.

Uma **verificação de crédito** ajuda a determinar se uma pessoa está ou não apta a pagar um empréstimo.

Os dados existentes nas agências de informações estão em **bancos de dados**, conjuntos de ficheiros de computador — arquivos, discos ou cartões perfurados — que podem ser pesquisados e lidos em poucos segundos. É



fácil para as agências de informações reunir, armazenar e recuperar enormes quantidades de informação e de dados abrangendo muitos detalhes sobre cada uma das pessoas.

Muita da informação existente nos **bancos de dados** são informações

pessoais. O ficheiro sobre os Barros, por exemplo, pode dar informações sobre o local onde vivem e onde trabalham; indica se têm casa própria ou se é alugada e, no primeiro caso, o montante do empréstimo que obtiveram para a sua aquisição; dele podem ainda constar informações como se ele cumpriu o serviço militar, se algum deles esteve preso ou se já foram processados alguma vez.

Será que os computadores «sabem» demais? Muitas pessoas assim pensam. Mas é evidente que os computadores, em si mesmos, não sabem

nada: apenas armazenam informações. A preocupação é que aquela informação possa ser dada a pessoas que é suposto não a deverem conhecer. Suponha que um vizinho tem acesso à informação sobre os Barros a partir de uma destas agências: neste caso houve uma invasão da privacidade.

Existem leis para proteger a privacidade. As agências de informações apenas podem recolher dados relacionados com a avaliação da sua capacidade de crédito. E devem manter estas informações em segredo, vendendo-as apenas a empresas ou agências que tenham o direito de conhecer a informação; assim, uma agência de informações não pode fornecer informações sobre os Barros a um dos seus vizinhos.

Outra questão que se põe é quando os bancos de dados contêm informações erradas. Por exemplo quando os Fragoso compraram um sofá novo para a sua casa e ele chegou danificado, recusaram-se a pagá-lo. A loja de mobiliário pôs uma acção em tribunal contra os clientes exigindo o dinheiro que lhe era devido; o juiz julgou o caso a favor dos Fragoso e deu-lhes razão para o não pagamento. Mas será que ganharam o caso? No banco de dados da agência de informações consta que eles não pagaram uma conta sendo por isso considerados como pouco merecedores de crédito.

Todos temos o direito de saber aquilo que consta a nosso respeito nos ficheiros. Se algo estiver errado e se a agência de informações estiver de acordo o erro será corrigido; se não houver acordo deve poder exigir-se que a agência inclua no ficheiro também a nossa versão da história. E assim os Fragoso podiam corrigir o seu ficheiro.

Um outro problema é o que se relaciona com a existência de informações muito antigas. Suponha uma família que há muitos anos não pagou algumas contas. Poderá ser considerada como pouco merecedora de crédito durante toda a vida? Isso tornar-lhe-ia impossível para sempre a concessão de um empréstimo ou uma compra a crédito.

Falência significa «ficar impossibilitado de pagar o dinheiro que se deve».

Todas as informações sobre contas não pagas e falências são destruídas sete anos depois. Desta forma alguém que tenha sido considerado pouco merecedor de crédito pode começar de novo com a «ficha limpa».

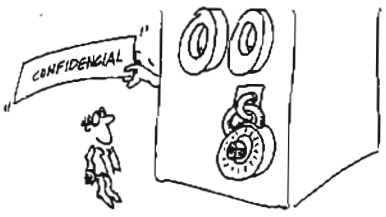
As agências de informações não são os únicos locais que têm bancos de dados: muitas empresas comerciais vão armazenando informações

sobre os seus clientes; as companhias de seguros mantêm registos de informações sobre os seus segurados e os hospitais sobre os seus doentes. Mas a maior colecção de bancos de dados pertence provavelmente ao governo dos Estados Unidos: existem bancos de dados nas Forças Armadas, no FBI, no Serviço de Impostos, no Serviço de Imigração e em centenas de outras agências governamentais. O Governo precisa de informações para proporcionar serviços aos seus cidadãos: assim, por exemplo, o Serviço de Impostos tem que conhecer os rendimentos de cada pessoa para poder colectá-la com um imposto justo; o FBI necessita de saber o mais possível acerca dos criminosos que persegue; o Serviço de Imigração tem que verificar os requerimentos para se assegurar de que não é possível a entrada ilegal de pessoas no país.

Os bancos de dados informatizados ajudam os serviços governamentais a executar os seus trabalhos de uma forma mais eficiente. Mas muitas pessoas preocupam-se que tantas informações sobre a vida privada dos cidadãos estejam em poder desses serviços. Que acontecerá se uma informação for mal utilizada ou usada contra alguém?

As leis de privacidade também se aplicam aos serviços governamentais. Os ficheiros são confidenciais. Sem uma razão forte, não é permitido a qualquer departamento da administração dar informações a alguém, a não ser que o interessado o autorize por escrito. As agências que mantêm ficheiros de dados pessoais devem tornar público o tipo de ficheiros de que dispõem. E há muitos casos em que o interessado pode ter acesso aos ficheiros e corrigi-los, se necessário.

Os computadores tornaram possível coleccionar, armazenar e processar milhões de factos sobre milhões de pessoas. Cabe-lhe a si e a todos os cidadãos assegurar-se de que os bancos de dados são usados em benefício da população, nunca o devendo ser injustamente contra as pessoas.



Os ficheiros **confidenciais** são ficheiros secretos ou privados.

tente responder

1. Quando se fala de um banco pensa-se geralmente num local onde se guarda dinheiro. O que é armazena um banco de dados?

2. Dos factos seguintes quais pensa que a Empresa Carros Económicos tinha o direito de conhecer antes de vender aos Barros um carro a crédito?

   a. quanto ganham o senhor e a senhora Barros

   b. onde vivem

   c. que donativos fazem para obras de caridade

   d. se devem dinheiro a outras empresas

   e. como é que os Barros se dão com os seus vizinhos

3. Que informações pensa que um serviço governamental deve obter de um banco de dados antes de contratar um novo empregado?

   a. se é cidadão americano.

   b. qual é o seu círculo de amigos.

   c. onde é que trabalhou antes.

   d. que dinheiro tem no banco.

Delitos informáticos e ética informática

Cada dia que passa são comprados mais computadores por empresas, pela administração pública e por particulares e há cada vez mais pessoas melhor preparadas para deles se utilizarem. A utilização crescente dos computadores levou ao aparecimento de um outro tipo de problemas, tão novos para a sociedade como os próprios computadores — os delitos informáticos. O que é um delito ou crime informático? Não se trata do roubo de um computador mas sim do seu uso para um acto ilegal. Este tipo de crime pode envolver o roubo de dinheiro, de propriedades ou de informações através da utilização de um computador.

Foi o caso, por exemplo de um banco da Califórnia que há alguns anos sofreu um roubo de mais de 10 milhões de dólares. Como foi roubado tanto dinheiro? Quem eram os ladrões? Traziam armas? Como fugiram?

O ladrão foi uma única pessoa. Praticou o crime sozinho, e nem sequer pôs os pés junto da caixa-forte do banco! Era um especialista de informática contratado para trabalhar nos computadores do banco e que descobriu como, através de um deles, se podia transferir dinheiro para outros bancos. Conhecido o código secreto que podia desencadear essa operação, usou-o para transferir 10 milhões de dólares para uma conta bancária que abrira noutro país.

Os bancos transferem diariamente entre si biliões de dólares, de forma que a transferência não foi logo detectada. E quando o foi, pensou-se que fazia parte dos negócios diários do Banco. Mas cinco dias depois do crime, o ladrão foi apanhado e o dinheiro devolvido.

Noutro exemplo de delito informático, o caixa de um banco usou o seu terminal para transferir pequenas quantidades de dinheiro de várias contas de clientes do banco para a sua própria conta. Também ele acabou por ser apanhado e punido.

Os bancos não são os únicos alvos dos criminosos da informática. Também as empresas que criam novos produtos sofrem roubos. Por vezes os roubos são de fitas ou de discos de computador, mas o mais vulgar é o roubo feito directamente da própria memória do computador de planos armazenados. Os autores de delitos informáticos são geralmente pessoas que têm acesso a um terminal e que encontram maneira de se ligarem a um grande sistema ou ao minicomputador de uma empresa, conseguindo assim, ilegalmente, ter acesso a informações. E por vezes não chega a haver o roubo de informações mas a sua alteração, o que provoca muitos problemas a quem necessita realmente de usar essas informações.

Qualquer lugar que utilize computadores pode ser um alvo.

Como podem ser protegidas as informações existentes nos computadores? Nos grandes sistemas informáticos usam-se dispositivos de segurança como palavras de senha *(passwords)* ou números de código, programados no próprio sistema: se alguém tentar usar o computador sem introduzir a palavra de senha correcta o computador resguarda ou «esconde» as informações e até pode estar programado para, nesse caso, «chamar» um elemento da segurança fazendo accionar uma campainha.

As palavras de senha e os números de código nem sempre são suficientes para conseguir a segurança de informações importantes: as senhas e os códigos podem ser roubados, ou até descobertos. Isto são problemas com que todos se têm de defrontar: empresas e utilizadores, devem proteger o mais possível as suas informações privadas; os construtores devem continuar a procurar as melhores maneiras de proteger os seus sistemas; e todos nós devemos exigir que sejam publicadas ou reforçadas as leis que protegem a informação e que punem os autores de delitos informáticos.

Mas não são as novas leis a única solução para a má utilização da informática. Todas as pessoas deveriam seguir um código de ética na utilização dos computadores e em todos os outros aspectos das suas vidas. A ética tem a ver com o que está certo ou errado e não com o que é determinado por lei. Algumas das utilizações incorrectas dos computadores não envolvem a violação de uma determinada lei, como os exemplos descritos na secção anterior não são mais do que uma utilização sem ética do computador. É o caso de quem usa um computador e um *modem* para penetrar nos ficheiros de uma empresa procurando informações pessoais: poderá não estar a violar uma lei determinada; mas não está a usar o computador de uma maneira ética.

Outro exemplo de abuso é a pirataria do *software*, a cópia ilegal de *software* comercial. Não é ético, nem legal, pois todo o *software* — quer se trate de jogos de computador, de programas de empresas, ou de qualquer outro tipo de programas que foi escrito para venda, está protegido, por *copyright*. Um artigo com *copyright* tem a sua reprodução limitada. Há uma lei publicada pelos governos que torna ilegal fazer cópias de *software* com *copyright*. Quem o faz está a infringir a lei.

Como pessoa instruída em informática é importante que use os computadores sempre eticamente e dentro da lei.

1. Pensa que um computador pode ser responsabilizado por um crime informático? Porquê?
2. Um amigo diz-lhe que pode copiar um jogo que ele comprou. Pensa que é ético fazer isso? Porquê?

Capítulo 13

AS CARREIRAS INFORMÁTICAS

Os computadores desempenham um papel na vida de todas as pessoas, afectando-a diariamente de alguma maneira. Há pessoas que decidiram estar mais envolvidas com eles, escolhendo carreiras que têm a ver com a informática e os computadores: algumas implicam ter de trabalhar-se com o respectivo *software* e outras com o respectivo *hardware*. Vamos ver em seguida algumas das carreiras informáticas existentes.

Recorde que o *software* se refere a programas de computadores e que o *hardware* refere-se à maquinaria.

Programador. Sabe-se que um computador não pode executar nenhum trabalho sem instruções. Um programador escreve as instruções ou o programa que indicam ao computador como deve executar um determinado trabalho. Para tal o programador deve conhecer a linguagem do computador para o qual está a escrever as instruções. Recorde que os computadores são projectados para compreenderem certas linguagens, como o BASIC, o Fortran, o COBOL e outras. A maior parte dos programadores conhece diversas linguagens de programação, o que lhes permite escrever instruções para diversos tipos de computadores. Muitos programadores formaram-se em escolas superiores onde estudaram informática. Há trabalhos que exigem programadores com conhecimentos especiais numa determinada área: por exemplo um programador que tenha necessidade de dar instruções a um computador para projectar uma ponte precisa de conhecimentos de engenharia; quem escrever programas para simular uma experiência de química necessita de preparação nesta matéria; e alguns programadores que trabalham para grandes empresas também têm conhecimentos de gestão.

Os dados podem ser uma série de nomes ou de números.

Operadores de registo de dados. Quer os programas quer o conjunto de dados necessários a um determinado trabalho têm de entrar no computador de alguma forma. Nas grandes empresas são os operadores de registos de dados que se encarregam dessa tarefa. Estes profissionais não precisam de saber muito de informática e alguns deles nem sequer vêem um computador enquanto trabalham. Quando os dados são introduzidos por cartões perfurados, os operadores de registo de dados usam máquinas de perfurar cartões; noutros tipos de computador, os operadores de registo de dados digitam os dados em terminais que dispõem de um teclado e de um *écran* vídeo.

No caso de um grande armazém comercial, cujo movimento contabilístico seja controlado por um grande sistema informático, sempre que um cliente paga uma conta, esse dado é registado no computador por um operador de registo de dados.

Os operadores não precisam de ter conhecimentos especiais sobre computadores; geralmente têm apenas o curso secundário, exigindo-se apenas que sejam bons dactilógrafos. É que se fizerem algum erro, esse erro repercutir-se-á na conta de alguém!

Bibliotecário de *software*. Algumas empresas têm centenas ou mesmo milhares de programas e de ficheiros gravados em fitas, discos ou cartões

perfurados. Quando é necessário recorrer a um determinado programa ou ficheiro ele tem de ser rapidamente encontrado. Por esta razão os discos, as fitas e os cartões estão arquivados em «bibliotecas» de *software,* dirigidas por bibliotecários de *software*. Uma biblioteca de *software* tem um sistema de arquivo para guardar cada peça de *software,* tal como uma biblioteca tem um sistema de arquivo para cada livro. Os bibliotecários de *software* têm como habilitação cursos secundários com boa capacidade de organização e de arquivo.

Analistas de sistemas. Como fará uma empresa que pretende informatizar os seus arquivos ou outras funções? Antes de comprar um computador há necessidade de muito trabalho de planeamento. Cabe a um analista de sistemas desenvolver um plano global para a utilização dos computadores na empresa. O analista de sistemas estuda os trabalhos a ser executados, decide como um computador os pode fazer de uma forma mais eficiente e nalguns casos até pode concluir que a utilização do computador *não é* a melhor maneira para resolver determinada tarefa.

Depois de fazer um estudo completo da empresa o analista de sistemas faz diversas sugestões. Esse estudo determinará o tipo de computador (ou computadores) a adquirir, onde devem ser colocados, as tarefas que executarão e os programas que precisam de ser escritos. O analista de sistemas irá depois dar instruções aos programadores sobre as finalidades e funções que cada um dos seus programas deverá cumprir.

As pós-graduações geralmente envolvem voltar à universidade para além dos anos habituais.

Os analistas de sistemas são formados por escolas superiores (geralmente com estágio ou pós-graduação) e têm de ter uma boa preparação de base em gestão de empresas e em informática.

Engenheiro informático. Como sabe, as partes mais importantes de um computador são os *chips* de circuitos integrados. Embora pequenos, contêm milhares de circuitos. Cada um deles deve estar colocado no local exacto e ser ligado a outros circuitos de uma maneira precisa. A concepção destes circuitos é tarefa para os engenheiros informáticos que tentam melhorar os *chips* e projectar novos modelos de circuitos, muitas vezes com o auxílio de computadores.

Os engenheiros informáticos também planeiam sistemas informáticos completos: projectam as unidades de entrada e de saída; planeiam as ligações entre as memórias e os *chips* da unidade central de processamento e às unidades de entrada e de saída; asseguram-se ainda de que todos os componentes do computador funcionam em conjunto pois uma falha numa pequena parte do computador pode causar a paragem de todo o sistema.

Os engenheiros informáticos devem ter uma boa preparação de base em engenharia electrónica. A sua formação geralmente inclui um curso superior e muitas vezes formação complementar.

Técnico de computadores. Depois de projectadas, as várias partes de um computador devem ser testadas, montadas e novamente testadas por um técnico de computadores. Por vezes estes técnicos fazem os planos dos modelos experimentais a partir do projecto preparado por um engenheiro. São conhecidos também como engenheiros de manutenção ou técnicos de campo, sendo responsáveis pelos trabalhos de reparação dos

Recorde que os *outputs* impressos em papel são também chamados *hardcopy*.

computadores e pela operacionalidade dos mesmos. Geralmente têm uma formação obtida numa escola técnica, num curso médio de dois anos ou nas Forças Armadas.

Operador de computador. Os operadores de computador são os encarregados de fazer funcionar os computadores. Se já operou com um microcomputador, então tem uma ideia, em pequena escala, do que é o trabalho de um operador de computador. Os operadores de computador são contratados para fazerem funcionar os grandes sistemas informáticos: ligam fitas ou discos às unidades de entrada para assim se carregarem os programas; põem cartões perfurados nos leitores de cartões para carregar programas e dados; quando são necessários *outputs* em papel, operam ainda como impressoras.

Os operadores de computador têm geralmente cursos secundários. Alguns recebem o seu treino de computadores numa escola técnica. Muitas empresas ministram também formação no próprio local de trabalho para os seus operadores de computador.

Responsável pelo sector informático. Muitas empresas têm nos seus escritórios um grande número de computadores: podem ter um grande sistema, um minicomputador com terminais ou uma combinação de grandes sistemas, minicomputadores e microcomputadores. Na empresa há funcionários ocupados a tempo inteiro com os computadores (programadores, operadores de registo de dados, operadores de computadores); outros funcionários são utilizadores em tempo parcial do computador (os que usam o tratamento de texto, a gestão de dados, as folhas de cálculo e outros «pacotes» de *software*). Por isso é necessário haver uma pessoa encarregue de todo o *hardware* e *software* da empresa: é o responsável pelo sector informático. Ele assegura que todas as actividades relacionadas com o computador decorrem com normalidade. É ele que chama os técnicos de computadores quando são necessárias reparações, que ensina a utilizar os programas de tratamento de texto, que adquire novo *software* para uma determinada tarefa de um dos funcionários da empresa.

O gestor informático tem geralmente um curso universitário de informática. É importante que se mantenha actualizado em todos os desenvolvimentos da informática, quer em termos de *hardware,* quer em termos de *software,* colaborando assim para o bom funcionamento da sua empresa.

Muitos vendedores têm um grau universitário.

Delegado comercial. Os sistemas informáticos são vendidos por delegados comerciais que servem de ligação entre os fabricantes do computador e os seus utilizadores. Têm de saber como funciona um computador e têm de perceber do negócio ou actividades dos clientes para lhes poderem explicar o que o computador pode fazer para eles.

Os delegados comerciais trabalham intimamente com os engenheiros projectistas e com os programadores para se manterem actualizados sobre as potencialidades dos computadores que estão a vender. Mas não devem ter só uma boa base de conhecimentos técnicos: devem também saber relacionar-se bem com as outras pessoas.

Técnicos de relações públicas e publicitários. Os técnicos de relações

públicas e de publicidade são as pessoas envolvidas na explicação ao público dos produtos de uma empresa. Trata-se de uma área particularmente importante para as empresas de informática. O trabalho dos técnicos de relações públicas e dos publicitários é explicar os computadores e a informática de uma maneira que seja fácil de entender, e fazer propaganda de uma maneira que leve o público a comprar os seus computadores.

Nos departamentos de relações públicas e nas empresas de publicidade trabalham artistas e escritores. Desenhos exactos e precisos e textos bons e claros ajudam a explicar os computadores ao público em geral.

Redactores técnicos. Qualquer novo computador vendido é acompanhado por manuais — livros que explicam como utilizar o computador (e alguns até também como programar).

Estes manuais são escritos por redactores técnicos, geralmente formados em universidades, dispondo de uma forte preparação electrónica e de um bom estilo de escrita.

Monitores de informática. Uma série de empresas usa a informática em muitas aplicações. Os seus empregados têm de ser ensinados a usar computadores. Este treino ou é feito pelos técnicos de vendas e pessoal dos departamentos de relações públicas ou por monitores de informática contratados para desempenhar essa tarefa. Estes têm de ter uma sólida base de conhecimentos de informática e devem estar familiarizados com a actividade da empresa para assim poderem explicar como poderão ser usados os computadores.

Os monitores de informática também são contratados por escolas. Geralmente são pessoas que adquiriram experiência em informática para além da que já tinham noutras áreas de ensino. Ensinam aos seus alunos informática, generalidades sobre computadores, processamento de dados ou programação em diferentes linguagens.

1. Algumas carreiras informáticas exigem trabalhar muito perto de um computador enquanto outras implicam trabalhar mais intimamente com as pessoas. (É evidente que todos os trabalhos envolvem trabalhar com pessoas em alguma extensão.) Das carreiras a seguir indicadas quais as que acentuam o trabalho com computadores e quais as que acentuam o trabalho com pessoas.

   a. programador
   b. operador de registo de dados
   c. bibliotecário de *software*
   d. analista de sistemas
   e. engenheiro informático
   f. técnico de computadores
   g. operador de computadores
   h. delegado comercial
   i. técnico de relações públicas e publicitário
   j. redactor técnico
   k. monitor de informática

2. Quais destes trabalhos são mais atraentes para si? Porquê?

FICHA DE CONTROLO

Escolha a melhor resposta para cada pergunta.

1. Qual o antigo dispositivo chinês auxiliar de cálculos?

   a. Ábaco
   b. Calculador graduado
   c. Máquina de Tabulação

2. Em 1835 Babbage projectou uma máquina que podia aceitar, armazenar, processar e fornecer informação. Indique o nome desta máquina:

   a. Ossos de Napier
   b. «Máquina analítica»
   c. Tear de cartões perfurados

3. Quando foram construídos os primeiros computadores para serem vendidos a empresas?

   a. anos cinquenta
   b. por volta de 1900
   c. cerca de 1840

4. Quais os condutores de electricidade nos computadores actuais de quarta geração?

   a. Tubos de vácuo
   b. Transistores
   c. Pastilhas *(chips)* de circuitos integrados

5. A escrita e revisão de uma carta dirigida a um colega podem ser feitas por um programa que seja de

   a. processamento de texto
   b. gestão de bases de dados
   c. folha de cálculo

6. Dos trabalhos seguintes quais os que não seria desejável que fossem efectuados por um computador, num estabelecimento comercial.

   a. calcular contas de clientes
   b. processar ordens de pagamento
   c. resolver queixas de clientes

7. Se um recibo computorizado tem um erro, tal deve-se provavelmente a

   a. engano do computador
   b. engano de uma pessoa
   c. má ligação eléctrica

8. Os bancos de dados de informações pessoais levantam preocupações sobre

   a. a velocidade a que os dados são processados
   b. as avarias do computador
   c. a privacidade individual

9. Os computadores podem ser responsabilizados por crimes

   a. verdadeiro
   b. falso
   c. às vezes

10. A cópia de *software* comercial é ilegal e não é ética

    a. verdadeiro
    b. falso
    c. às vezes

11. Uma pessoa que escreve programas de computador é

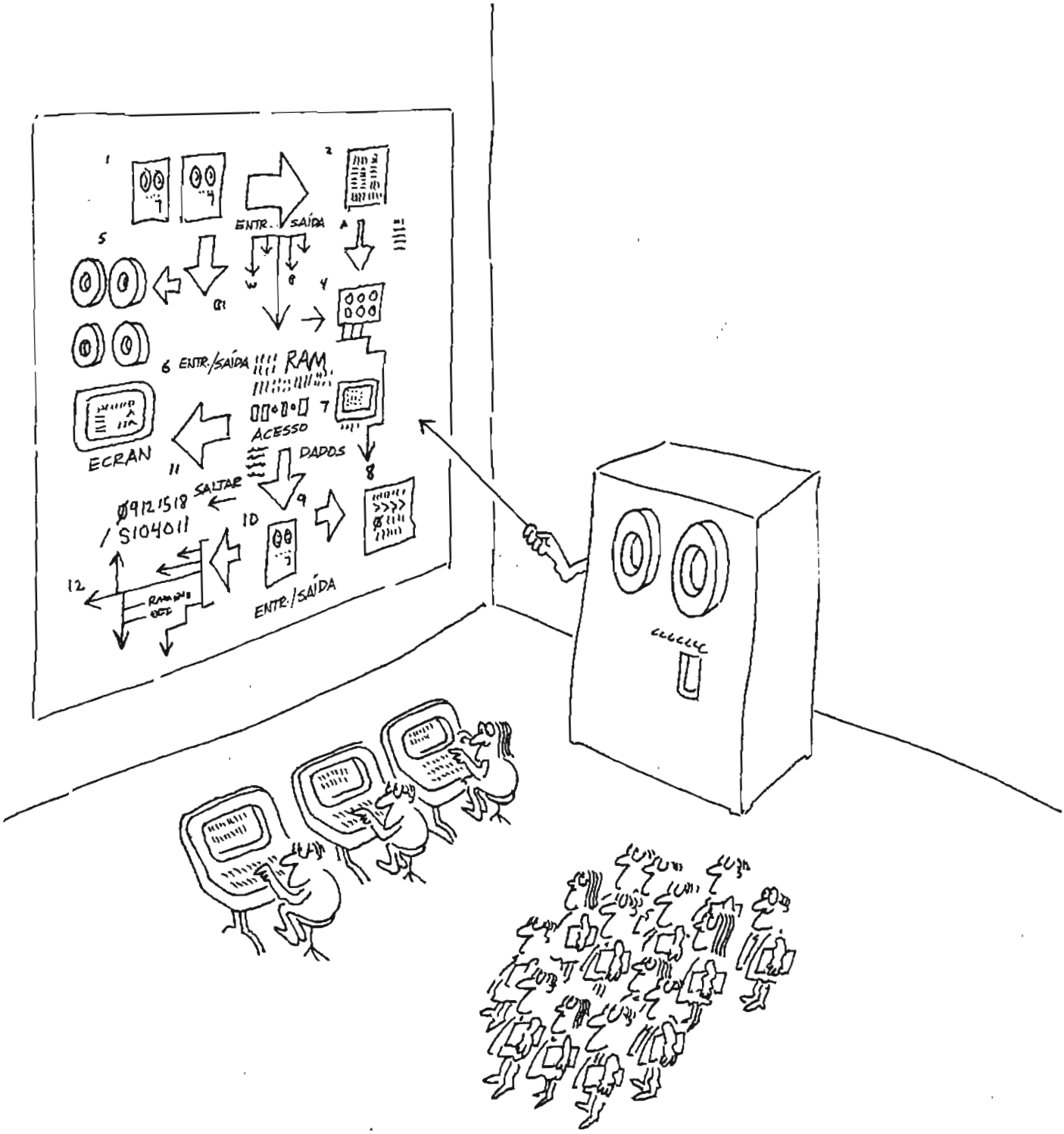
    a. um operador de registo de dados
    b. um operador de computador
    c. um programador

12. Uma pessoa que repara computadores é um

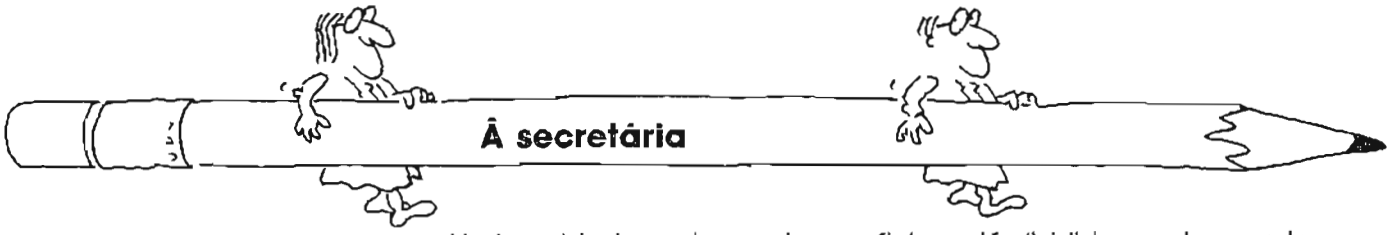
    a. delegado comercial
    b. técnico de computadores
    c. bibliotecário de *software*

UNIDADE 3

PROGRAMAÇÃO BASIC

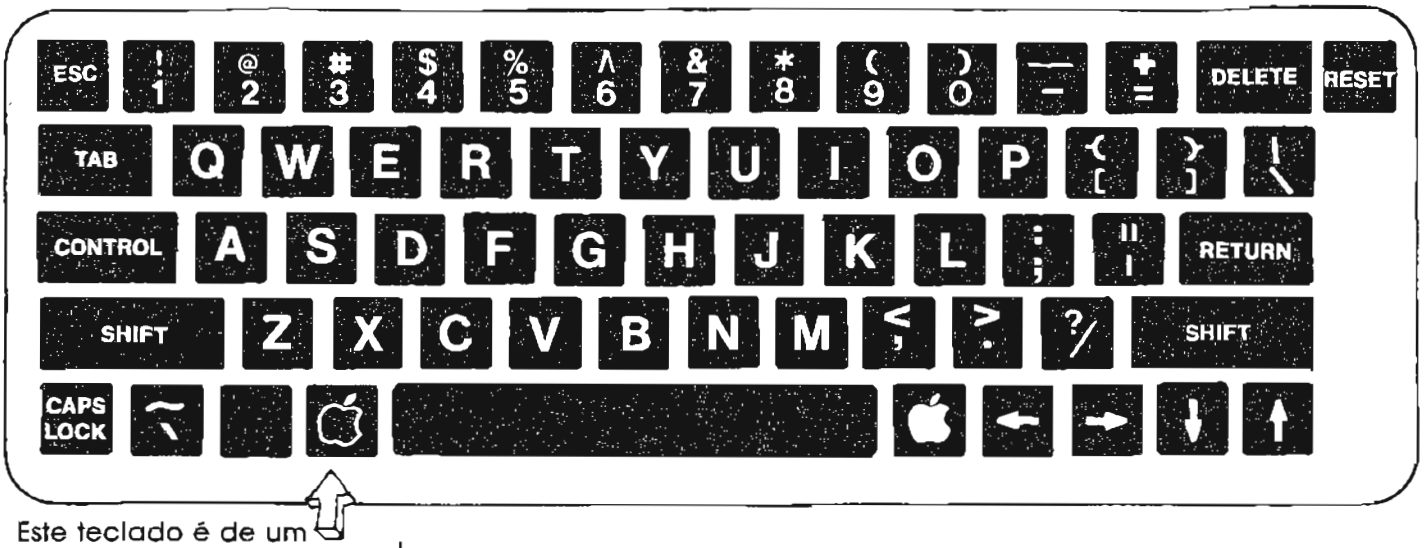


Capítulo 14 CONHEÇA O TECLADO DO SEU COMPUTADOR



Nesta unidade cada um dos capítulos está dividido em duas partes: «À secretária» e «No computador». Na primeira faz-se a introdução de palavras e de símbolos da linguagem BASIC, mostrando como são utilizados na escrita de programas; quando chegar à secção «No computador» deve levar este livro para junto do seu microcomputador e fazer os exercícios indicados.

É agora a altura de começar a comunicar com o seu computador. Em primeiro lugar vamos dar uma olhadela sobre a unidade de entrada que é geralmente usada — o teclado.



Este teclado é de um Apple IIe. Se o seu computador não for deste modelo, o teclado terá algumas diferenças.

Se já alguma vez usou uma máquina de escrever verá que o teclado do computador é similar ao de uma delas. As teclas não estão colocadas por ordem alfabética. Tente procurar, no teclado acima representado, as letras

A P B S T I N

Já reparou que as letras representadas nas teclas são todas maiúsculas? Na maior parte dos computadores assim acontece.

Alguns microcomputadores têm o bloco de teclas de números do lado direito do teclado principal.

Na fila superior do teclado situam-se as teclas de números, existindo também algumas teclas contendo símbolos especiais na parte superior. Por exemplo, a tecla com o número 4 situada na fila de cima tem também na parte superior o cifrão ($). Para escrever um cifrão terá que premir uma das duas teclas SHIFT (de maiúsculas) em simultâneo com a tecla que corresponde ao 4 e ao $. É isto que terá que fazer sempre que quiser

Olhe para o diagrama do teclado representado na página anterior. Procure as teclas SHIFT e a que tem o 4 e o $.

escrever um dos símbolos representados na parte superior de cada tecla. Esta regra tem excepções como é o caso do computador PET[4] em que a maior parte dos símbolos não está na parte superior das teclas mas num bloco ao lado direito do teclado principal ou ainda no caso do Spectrum e Timex em que cada tecla associa uma instrução em BASIC[5].

Abaixo da última fila de teclas existe a **barra de espaços.** Sempre que se carrega na barra de espaços o computador imprime no *écran* um espaço em branco.

EU SOU UM COMPUTADOR.

Os espaços entre palavras são «impressos» premindo a barra de espaços.

Existe em todos os computadores uma tecla que irá ser muito utilizada (nalguns deles é a tecla **RETURN**, noutros a tecla **ENTER**) e que faz «entrar» as instruções do programa na memória de acesso aleatório — a RAM — do seu computador. Sempre que se acaba de digitar uma instrução deve-se carregar nesta tecla.

Os computadores Apple, Atari e Commodore têm a tecla RETURN. Os computadores ZX Spectrum e Timex têm a tecla ENTER. Nos computadores IBM e compatíveis a tecla RETURN ou ENTER está marcada com o sinal ↵.

À medida que for usando o computador verificará como a pouco e pouco se torna mais fácil proceder à localização das teclas e saber como usar cada uma delas. Vejamos o que já sabe sobre o teclado.

tente responder

Escolha a resposta correcta para cada uma das seguintes questões:

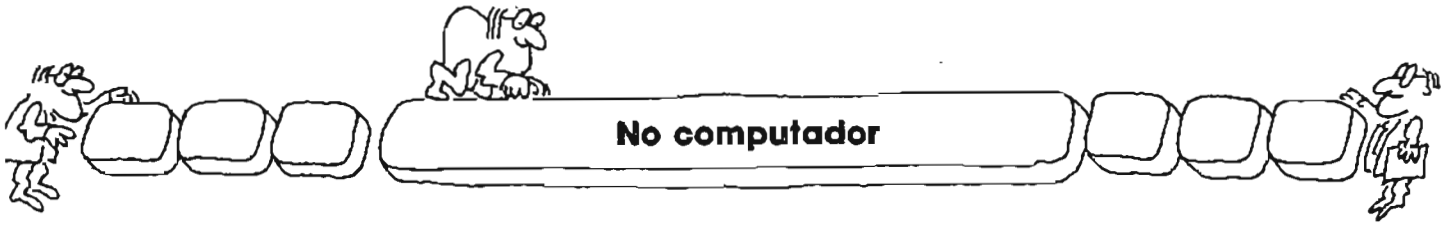
1. Qual dos símbolos seguintes representa o número zero?
   a. O
   b. Ø

2. As letras representadas num teclado são:
   a. letras maiúsculas
   b. letras minúsculas

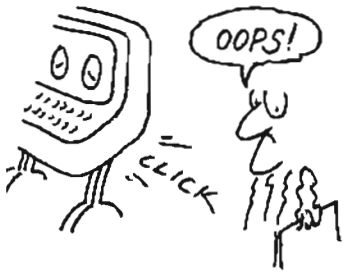
3. Para enviar uma instrução para a memória do computador deve premir a tecla:
   a. RETURN ou ENTER
   b. SHIFT

4. Veja o diagrama de um teclado representado na página anterior. Diga se tem ou não de carregar na tecla SHIFT para escrever os símbolos a seguir indicados. Escreva *sim* ou *não*.

a. !	e. ?
b. =	f. .
c. %	g. ,
d. —	h. +



No computador



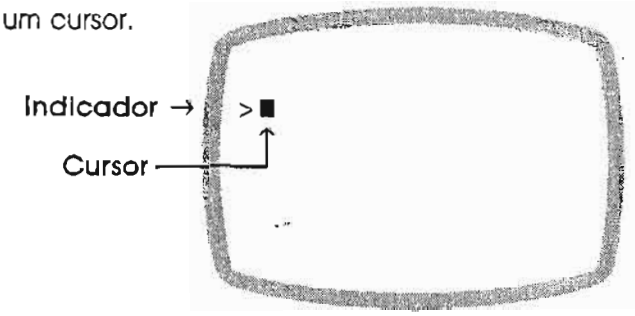
Já é altura de usar o seu microcomputador e de nele escrever algumas das coisas que acaba de aprender sobre os computadores. Assegure-se, que o computador está operacional e ligue-o. Procure na parte inferior, superior ou lateral do teclado o interruptor *on-off* geralmente em local de difícil acesso propositadamente! Se acidentalmente desligasse o computador enquanto escrevia o seu programa, perderia toda a informação que estivesse na RAM, a memória de acesso aleatório onde os programas ficam armazenados.

Assegure-se de que o seu computador está pronto a ser utilizado — isto é, deve haver um **cursor** ou um **indicador** no *écran*. O **indicador** é um sinal que significa que o computador está à espera das suas instruções. O **cursor** indica o local do *écran* onde vai ser impresso o próximo carácter cuja tecla premir. Os símbolos do cursor e do indicador são diferentes conforme os microcomputadores. Mostra-se em seguida um exemplo de um indicador e de um cursor.

Alguns computadores não têm indicador.

Cursor deriva da palavra latina *(cursor)* que significa «corredor». O cursor «corre» através do *écran*.

Alguns computadores emitem as mensagens OK ou READY antes de aparecer o indicador ou o cursor. Outros usam o símbolo] como indicador e o símbolo _ como cursor.



Agora está em condições de escrever no teclado. Escreva com as duas mãos; inicialmente pode parecer difícil, mas com a prática verá que é cada vez mais fácil encontrar as teclas de que precisa. E com as duas mãos escreverá mais depressa.

A utilização do teclado

1. Escreva o seu nome no *écran*. Carregue na barra de espaços para deixar um espaço entre o primeiro e o último nome. Verifique o movimento do cursor à medida que vai escrevendo.

2. Agora apague o seu nome.

   Se tem um computador IBM ou compatível mova o cursor para o começo do seu nome carregando na tecla marcada com uma seta virada para a esquerda (←). Então prima a tecla RETURN.

   Se tem um computador Atari mova o cursor para trás carregando na tecla BACKS. As letras irão desaparecendo à medida que se move o cursor.

Se estiver a usar um computador IBM ou um compatível pode programá-los quer em letras maiúsculas quer em letras minúsculas. No computador IBM, premindo CAPS LOCK, muda-se de maiúsculas para minúsculas e vice-versa. No ZX Spectrum carregando na tecla CAPS SHIFT e premindo 2 (CAPS LOCK) muda-se de maiúsculas para minúsculas e vice-versa.

Se tem um computador Commodore mova o cursor para trás premindo a tecla DEL.

Se tem um computador ZX Spectrum mova o cursor para trás carregando na tecla CAPS SHIFT e no 0 (DELETE). As letras desaparecerão à medida que se move o cursor.

3. Procure os seguintes caracteres no teclado e escreva-os:[6]

   A 2 O Ø , . / ; -

4. Agora carregue na tecla ENTER ou RETURN. Isto indica ao computador para fazer entrar na RAM a informação que se digitou no teclado. O computador emite uma mensagem de erro. O erro é um **erro de sintaxe** e a mensagem será constituída pelas palavras ERROR, SYNTAX ERROR ou uma abreviatura como SN ERROR.[7] Isto significa que o computador não compreendeu a instrução entrada na RAM e por isso emitiu a mensagem. É como se o computador lhe estivesse a dizer: «Não o compreendo!» A razão por que o computador não entende é porque não foi escrita uma instrução em linguagem BASIC. Nestes casos o computador ignorará esses caracteres e esperará por outra instrução.

Sintaxe tem a ver com a estrutura da linguagem.

5. Agora pratique premindo outros caracteres mesmo que o computador não entenda o que você está a escrever. Procure as teclas que têm símbolos na sua parte superior e escreva esses símbolos, carregando em simultâneo na tecla SHIFT.

6. Carregue novamente na tecla ENTER ou RETURN. Obterá uma mensagem de erro de sintaxe! O computador ainda não o compreendeu.

7. Agora limpe o *écran*.

   Se tem um computador Atari, prima CLEAR (será necessário carregar também na tecla SHIFT). Pode também escrever GR.O.

   Se tem um computador Commodore prima CLR (será preciso carregar na tecla SHIFT).

   Se tem um computador IBM ou compatível, escreva CLS (de *clear screen* — limpar *écran*).

   Se tem um computador Spectrum, prima a tecla C (surgirá a instrução CLS no *écran*)[8].

Escrever em BASIC

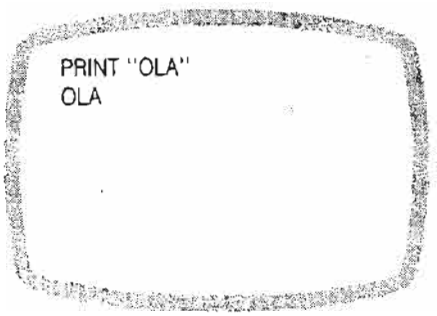
1. Desta vez vamos escrever algo que o computador possa entender — uma mensagem em BASIC. Escreva a linha seguinte:

   PRINT "OLA"

Recorde-se de que o BASIC é uma linguagem de programação de computadores.

Lembre-se que ENTER ou RETURN indicam ao computador que acabou de escrever a instrução.

Agora prima ENTER ou RETURN. No computador aparecerá:



PRINT é um termo da linguagem BASIC que o computador entende e que vai cumprir, «imprimindo» (em inglês *print*) no *écran* o que estiver entre aspas.

2. Se se esqueceu do espaço entre PRINT e OLÁ tal não tem importância. O computador ignora todos os espaços excepto os que estiverem entre aspas. Tente escrever o seguinte, sem deixar espaços:

   PRINT "OLA"

   Terá o mesmo resultado que anteriormente. Os espaços têm a função de tornar mais fácil a leitura às *pessoas.*

Depois de escrever isto prima ENTER ou RETURN.

3. E se escreveu mal a palavra entre aspas? O computador escreverá o que lá estiver, com erros e tudo. Seguirá exactamente as suas instruções tal como foram escritas. Tente escrever:

   PRINT "CMPUTADOR"

   E o computador imprimirá no *écran*:

   CMPUTADOR

Depois de escrever isto prima ENTER ou RETURN.
Nota: Depois de escrever uma linha de instruções prima sempre ENTER ou RETURN. A partir de agora não lhe voltaremos a recordar isto.

4. E se se enganar na palavra PRINT? Experimente escrever:

   PRONT "OLA"9

   Receberá uma mensagem de erro. Quando há um erro num termo de BASIC, o computador não consegue entender a instrução. Qualquer erro numa instrução de um programa é chamado um **bug**. Ao longo desta unidade verá o símbolo de "Bug Alert" ou "Alerta de Erros" a ajudá-lo a evitar os erros mais frequentes.

Escreva PRONT em vez de PRINT.

5. Se introduziu no computador uma instrução com um erro escreva-a novamente. Volte a escrever a instrução anterior:

   PRINT "OLA"

Desta vez escreva novamente tudo, sem o erro. Depois prima ENTER.

6. Agora limpe o écran e escreva esta instrução:

   PRINT "ATE LOGO"

Se não está seguro da maneira de limpar o *écran* veja a pág. 95.

Se detectou um erro antes da instrução ter sido enviada para a RAM (ou seja antes de ter premido RETURN ou ENTER) pode apagar a linha, voltando para trás até ao erro e escrevendo o resto da linha novamente. Se só viu o erro depois de ter premido RETURN ou ENTER já não pode voltar atrás: em vez disso terá de reescrever toda a linha.

Veja a página 94.

Capítulo 15 DIGA OLÁ AO SEU COMPUTADOR

É provável que já tenha visto um computador imprimir mensagens ou respostas no *écran*:

Como sabe, o computador não tem cérebro. A impressão destas mensagens não seria possível se ele não tivesse recebido instruções para tal. Um conjunto de instruções que indicam ao computador o que ele tem de fazer chama-se um programa.

Como se escrevem os programas? Como saber o que se deve dizer ao computador? Em primeiro lugar tem de se decidir o que se quer que o computador faça — resolver um problema, imprimir uma mensagem ou executar qualquer outra tarefa; em seguida deve-se elaborar passo a passo um plano a seguir: o **algoritmo**; finalmente escreve-se cada passo do algoritmo em linguagem de computador (utilizar-se-á aqui a linguagem BASIC). O programa é este conjunto de passos escritos em linguagem de computador.

Neste capítulo aprenderemos como dar instruções, ou seja, como programar um computador por forma a imprimir mensagens ou resolver problemas aritméticos. E veremos a forma de usar os símbolos de pontuação para os resultados poderem ser apresentados de diferentes maneiras.

A utilização da instrução PRINT

Atente no seguinte programa:

```
10 PRINT "EU GOSTO DE COMPUTADORES"
20 PRINT "OS COMPUTADORES GOSTAM DE JOVENS"
30 END
```

Com este programa quais são as instruções dadas ao computador? Isso não é um mistério. Se adivinhou que este programa irá dar instruções ao computador para escrever no *écran* as frases EU GOSTO DE COMPUTADORES e OS COMPUTADORES GOSTAM DE JOVENS então acertou.

Vejamos agora como o programa foi escrito. Cada instrução começa por um **número de linha**, devendo as linhas ser numeradas pela ordem por que se deseja que o computador as siga, começando no número mais pequeno. Examinemos o programa da maneira como o computador o faz, verificando a ordem por que as instruções estão a ser executadas e como; a isto chama-se fazer o **tracing** do programa.

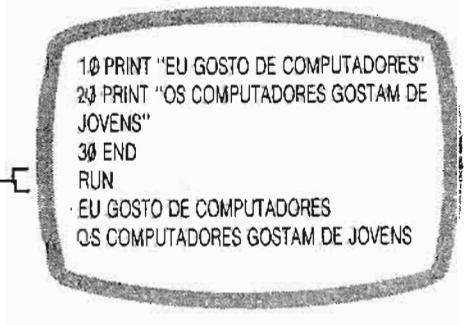
Podem usar-se os números que se desejarem para números de linha, mas os programadores, geralmente, saltam de 10 em 10.

A linha 10 contém a palavra PRINT e uma frase entre aspas. PRINT é uma instrução de BASIC que diz ao computador para *imprimir* (em inglês *print*) o que estiver dentro das aspas — palavras, números, símbolos e espaços. Assim, a linha 10 indica ao computador para imprimir a frase EU GOSTO DE COMPUTADORES.

A linha 20 indica ao computador para imprimir a frase OS COMPUTADORES GOSTAM DE JOVENS.

A linha 30 é a declaração END pela qual o computador sabe que alcançou o *fim* (em inglês *end*) do programa.

Uma vez escrito, o programa é guardado na RAM, a memória de acesso aleatório do computador e, até que se lhe dê a respectiva ordem, ele não vai executar o programa. Para o fazer precisa de uma outra instrução, o comando *RUN*, que indica ao computador que deve seguir as instruções guardadas na RAM. O *écran* agora aparecerá desta forma:



Isto é o *programa* ou *input*.

Este é o *comando* para processar o programa.

Isto é o *output* que o computador imprime.

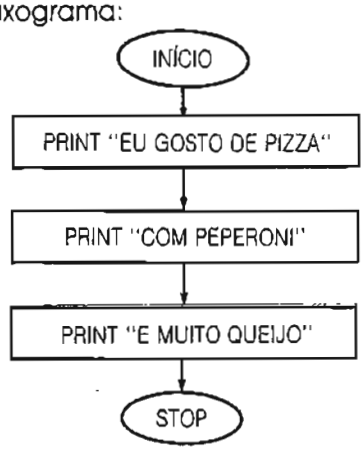
RUN é um **comando.** Indica ao computador para executar imediatamente uma coisa, mas, como não faz parte do programa, não tem consequentemente número de linha. PRINT e END são instruções; têm número de linha e ficam armazenadas na RAM. O computador apenas cumpre as instruções constantes de linhas com números de linha quando for introduzido o comando RUN.

O *output* (a saída) é aquilo que o computador exibe no *écran* nas linhas abaixo de RUN, e é resultante das instruções constantes do programa. Se olhar cuidadosamente para o *output* verá que o computador apenas imprimiu as palavras e os espaços dentro de aspas, não imprimindo nem os números de linhas, nem a palavra PRINT, nem as próprias aspas. Mas imprimiu os resultados em duas linhas separadas — como estava estabelecido no programa. E porque é que não imprimiu END?

Um **fluxograma** é um esquema dos passos utilizados para resolver um problema, usado pelos programadores no planeamento dos seus programas.

Por vezes um fluxograma é um auxiliar útil para compreender como um computador segue as instruções de um programa. Um fluxograma indica o percurso que o computador faz quando segue as instruções. Apresenta-se em seguida um programa e o respectivo fluxograma:

```
10 PRINT "EU GOSTO DE PIZZA"
20 PRINT "COM PEPERONI"
30 PRINT "E MUITO QUEIJO."
40 END
```



Note que o fluxograma tem dois tipos de símbolos: os ovais ⬭ no início e no fim, utilizados para as instruções respectivas, e os rectângulos ▭ para as outras instruções.

De cada símbolo, excepto do símbolo STOP, parte uma seta que indica a instrução que o computador irá executar em seguida.

1. Observe o seguinte programa de computador:

```
10 PRINT "QUE LINGUA SE FALA EM PORTUGAL?"
20 PRINT "EU FALO BASIC!"
30 END
```

   a. Faça a análise do programa, isto é, explique o que cada linha manda executar ao computador.

   b. Qual seria o *output* deste programa?

2. Escreva o *output* do seguinte programa:

```
10 PRINT "BOM-DIA COMPUTADOR"
20 PRINT "**ESTOU BOM**"
30 PRINT "ADEUS!!"
40 END
```

Não esqueça os números de linha, PRINT, as aspas e END.

3. Escreva um programa que forneça o seguinte *output*:

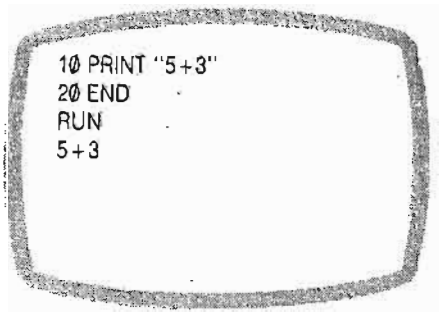
```
HOJE NÃO TENHO ESCOLA... 12
É DOMINGO
```

4. Desenhe um fluxograma para o programa seguinte, não se esquecendo de usar os símbolos correctos:

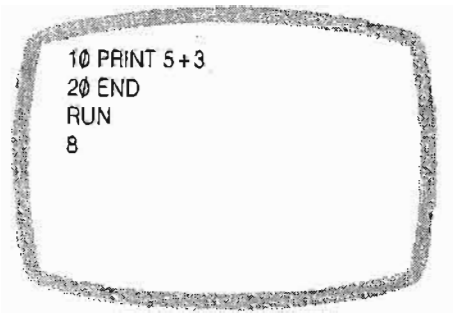
```
10 PRINT "ESTE É"
20 PRINT "O MEU"
30 PRINT "PRIMEIRO"
40 PRINT "FLUXOGRAMA!"
50 END
```

Cálculos

Observe o programa e o *output* representados no *écran* seguinte. O computador imprimiu a «mensagem» entre aspas.



Veja agora o programa seguinte e o respectivo *output*:



Que diferença existe nas linhas 10 dos dois programas? No segundo não há aspas na declaração PRINT, significando a instrução PRINT 5+3 que o computador vai calcular 5+3 e imprimir *apenas* a resposta: 8.

O programa seguinte mostra mais alguns cálculos.

```
10 PRINT 3*4
20 PRINT 16/2
30 END
RUN
12
8
```

Sabe o que significam os símbolos * e /? O asterisco (*) é o símbolo de uma operação de multiplicação: 3*4 significa 3 vezes 4. Não se pode usar o símbolo x pois ele já representa a *letra X*. A barra (/) é o símbolo de uma operação de divisão pelo que 16/2 significa 16 a dividir por 2. O sinal de divisão (÷) não existe no teclado.

Já reparou que a barra faz com que uma operação de divisão se pareça com uma fracção? Não esqueça que as fracções são uma forma de divisão.

Veja agora o programa seguinte onde se combinam duas características que já conhecemos da declaração PRINT.

Uma declaração PRINT tem aspas e a outra não as tem.

```
10 PRINT "7+5="
20 PRINT 7+5
30 END
RUN
7+5=
12
```

Façamos a análise *(tracing)* do programa.

A linha 10 indica ao computador para imprimir o que estiver entre aspas; assim o computador imprime 7+5=.

A linha 20, sem aspas, indica ao computador para calcular e imprimir a resposta de 7+5 que é 12.

A linha 30 finaliza o programa.

tente responder

1. Escreva o *output* do programa seguinte:

```
10 PRINT 12+3
20 PRINT 9-1
30 PRINT 6*2
40 PRINT 8/4
50 END
```

2. Escreva o *output* do programa seguinte:

```
10 PRINT "24/6="
20 PRINT 24/6
30 PRINT "3*3="
40 PRINT 3*3
50 END
```

Ponha o computador a fazer o cálculo.

3. Escreva um programa cujo *output* seja o seguinte:

```
15-4=
11
```

Você decide os números e a operação que deverá usar.

4. Escreva um programa cujo *output* seja o seguinte:

```
A RESPOSTA É
10
```

A utilização de sinais de pontuação[13]

Já conhecemos o *output* deste programa.

```
10 PRINT "26+5="
20 PRINT 26+5
30 END
RUN
26+5=
31
```

Vejamos agora o que a colocação de um **ponto e vírgula (;)** no final da linha 10 faz ao *output*.

```
10 PRINT "26+5=";
20 PRINT 26+5
30 END
RUN
26+5=31
```

O ponto e vírgula indica ao computador para *continuar o output na mesma linha*, não saltando para a linha seguinte.

A combinação de instruções poupa tempo. Não tem de escrever uma linha nova.

Podemos também usar o ponto e vírgula para pôr mais do que uma instrução PRINT numa linha, usando PRINT uma única vez. Os dois programas seguintes produzirão o mesmo *output*.

```
10 PRINT "12*3=";
20 PRINT 12*3
30 END
```

Há computadores que permitem combinar instruções sem usar um ponto e vírgula entre elas.

```
10 PRINT "12*3="; 12*3
20 END
```

O ponto e vírgula não só separa a parte de uma linha entre aspas da parte da linha que *não está* entre aspas, como ainda dá instruções ao computador para continuar o *output* na mesma linha.

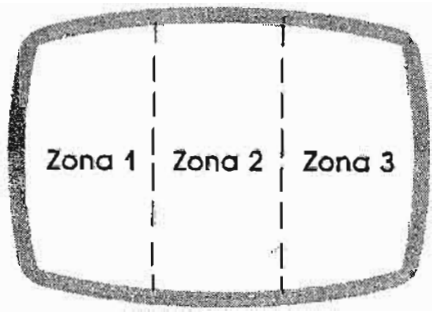
Lembre-se sempre que um ponto e vírgula (;) numa declaração PRINT diz ao computador para *continuar na mesma linha.*

Observe agora o programa abaixo indicado. As três primeiras declarações PRINT têm no seu final uma **vírgula (,)**. Veja o *output*: as quatro respostas aparecem numa única linha, mas separadas.

```
10 PRINT 12+8,,
20 PRINT 16-4,
30 PRINT 8*5,
40 PRINT 20/4
50 END
RUN
20      12      40      5
```

Output das linhas 10, 20, 30 e 40.

Imagine que o *écran* do computador está dividido em várias secções chamadas **zonas**. Uma instrução PRINT com uma vírgula indica ao computador para *continuar o output na zona seguinte.*



Nem todos os computadores têm três zonas. Alguns têm mais e outros menos.

Também se pode usar a vírgula para se proceder a mais do que um PRINT numa linha, com uma única instrução PRINT.

```
10 PRINT 15+5, 15-5, 15*5
20 PRINT 12+3, 12-3, 12*3
30 END
RUN
20      10      75
15      9       36
```

output da linha 10[14]
output da linha 20

Recorde que um *output* impresso pelo computador em papel também se chama *printout*.

Talvez já tenha visto impressões de computador com muitos números listados em colunas. A declaração PRINT com vírgulas, utiliza-se muitas vezes com a finalidade de imprimir *outputs* em colunas.

PROFESSOR	REGISTO ESCOLAR NÚMERO DA SALA DE AULA	NÚMERO DE ALUNOS
R. JACOB	122	22
S. GOMES	123	26
R. RODRIGUES	124	27
J. FREITAS	125	22

No mesmo programa, podem-se usar nas declarações PRINT vírgulas e pontos e vírgulas. O facto de usar uma vírgula, um ponto e vírgula ou não usar qualquer pontuação depende daquilo que se quer que o programa faça e de como se deseja o aspecto gráfico do *output*. Recorde-se que, quer a vírgula, quer o ponto e vírgula são utilizados para combinar instruções; mas enquanto a vírgula imprime os *outputs* em zonas diferentes da mesma linha, o ponto e vírgula imprime-os uns a seguir aos outros na mesma linha. Veja o programa seguinte e o seu *output* e perceberá as diferenças.

```
10 PRINT "BANANAS", "LARANJAS"
20 PRINT "BANANAS"; "LARANJAS"
30 END
RUN
BANANAS       LARANJAS
BANANASLARANJAS
```

Veja onde se colocaram a vírgula e o ponto e vírgula.

Output da linha 10

Output da linha 20

tente responder

Quando escrever as respostas deixe espaços para mostrar as zonas.

1. Escreva o *output* do programa seguinte:

```
10 PRINT "18/3 = ";
20 PRINT 18/3
30 END
```

2. Escreva o *output* do programa seguinte:

```
10 PRINT 1*1, 2*2, 3*3
20 END
```

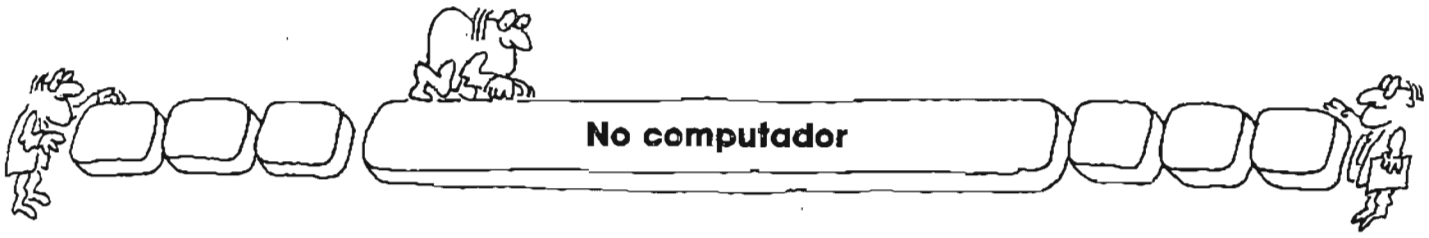
3. Escreva o *output*

```
10 PRINT "OS COMPUTADORES", "SAO", "DIVERTIDOS"
20 PRINT "OS COMPUTADORES", "SAO", "FACEIS"
30 END
```

Há mais do que uma maneira de escrever este programa.

4. Escreva o programa que imprime o *output* seguinte em zonas:

```
HA      HA      HA
```



No computador

Nesta secção iremos praticar a escrita de programas para computador. Poderá usar instruções PRINT e os sinais de pontuação que entender convenientes. Mas primeiro vamos aprender a fazer alterações nos programas, mudando, acrescentando ou eliminando linhas.

Introduzir alterações em programas

1. Digite cuidadosamente este programa:

```
10 PRINT "OLA"15
20 PRINT "ADEUS"
30 END
```

Não esqueça as aspas!

Agora indique ao computador para processar o seu programa premindo RUN. No *écran* poderá ver o *output* seguinte:

```
OLA
ADEUS
```

Lembre-se que deve carregar em ENTER ou RETURN quando acabar de escrever cada linha.

Recorde-se que RUN é um comando, pelo que não deve levar número de linha. Depois de teclar este comando deve premir ENTER ou RETURN.

2. Vamos agora alterar a linha 20 de modo a o computador escrever ATE LOGO em vez de ADEUS. Para alterar uma linha do programa, volte a escrevê-la:

Escreva isto.

```
20 PRINT "ATE LOGO"
```

No *écran* passaremos a ter:

```
10 PRINT "OLA"
20 PRINT "ADEUS"
30 END
RUN
OLA
ADEUS
20 PRINT "ATE LOGO"
```

Como LIST também é um comando não exige número de linha.

3. Agora introduza o comando LIST, que indica ao computador para imprimir o programa armazenado na RAM. E verifique que a «nova» linha 20 substitulu a «antiga».

```
LIST
10 PRINT "OLA"
20 PRINT "ATE LOGO"
30 END
```

4. Faça agora correr o programa accionando o comando RUN. Desta vez o *output* será:

```
OLA
ATE LOGO
```

5. Limpe o *écran* do computador.

   Se tem um computador IBM ou compatível escreva CLS.
   Se tem um computador Atari, prima SHIFT e CLEAR ou escreva GR.O.
   Se tem um computador Commodore prima SHIFT e CLR.
   Se tem um computador ZX Spectrum prima C (surge CLS no *écran*).

   Será que perdeu o programa? Não, ainda está armazenado na RAM. Accionando esta instrução apenas se limpou o *écran*, mas não a memória do computador. Veja novamente o programa escrevendo LIST.

6. Junte outra linha ao seu programa de modo a que o *output* tenha SOU UM COMPUTADOR entre OLA e ATE LOGO. Para acrescentar esta linha ao programa escolha um número de linha entre 10 e 20 e escreva a instrução que deseja. Eis um exemplo:

   Escreva isto. Pode usar qualquer número de linha entre 10 e 20.

   ```
   15 PRINT "SOU UM COMPUTADOR"
   ```

7. Agora liste o programa existente na RAM, para o que deve escrever LIST. Veja como o computador inseriu a linha 15 na ordem correcta. O programa parecer-se-á com o seguinte:

   ```
   LIST
   1Ø PRINT "OLA"
   15 PRINT "SOU UM COMPUTADOR"
   2Ø PRINT "ATE LOGO"
   3Ø END
   ```

   Escreva RUN.

   É altura de fazer correr o programa.

8. Elimine a linha que imprime OLA. Para tal digite o número de linha que quer eliminar e prima RETURN.

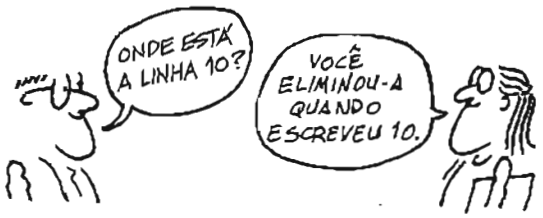
   Escreva isto.

   ```
   1Ø
   ```

   A linha ainda lá está? Está no *écran*, mas não na memória do computador. Escreva LIST para ver o programa na RAM e terá:

   ```
   LIST
   15 PRINT "SOU UM COMPUTADOR"
   2Ø PRINT "ATE LOGO"
   3Ø END
   ```

   Escreva RUN e corra o programa.



9. Antes de passar a um próximo exercício deve apagar este programa da RAM. Para tal introduza o comando NEW. Agora faça RUN. Acontece alguma coisa?
   Marque LIST. O programa ainda está na RAM? Não, NEW apagou o programa da memória do computador e criou espaço para um *novo* (em inglês *new*) programa.

   Escreva isto.

A utilização de declarações PRINT

1. Escreva um programa para imprimir o seu nome na primeira linha do *output*, a sua morada na segunda linha e o número de telefone na terceira linha. A seguir mostra-se parte de um programa de exemplo:

Não esqueça as aspas!

```
10 PRINT "ROSA VIEIRA"                    ← Escreva o seu nome
20 PRINT "....."                          ← Indique aqui a sua morada
30 PRINT "....."                          ← Indique aqui o seu telefone.
```

Faça RUN.

2. Acrescente ao seu programa, abaixo do endereço, uma linha para indicar a cidade. Faça RUN e LIST ao programa.

3. Elimine a linha que imprime o número de telefone. Faça LIST e RUN ao programa.

4. Altere a linha com o seu nome indicando o nome de um seu irmão, irmã ou amigo. Faça LIST e RUN ao programa.

5. Escreva NEW para apagar o programa e escreva e corra o programa seguinte:

Escolha um número de linha entre 20 e 30.

Para eliminar uma linha escreva o respectivo número de linha seguido de ENTER.

Recorde-se que para alterar uma linha deve voltar a escrevê-la.

```
10 PRINT "20+9=";
20 PRINT 20+9
30 END
```

6. Elimine a linha 20 e altere a linha 10 de forma a incluir ambas as instruções de PRINT.

7. Escreva NEW para apagar o antigo programa da RAM. Escreva um programa para calcular 543÷3. (Deixe que seja o computador a fazer o cálculo e não você.)

O sinal de divisão no teclado é a barra (/).

8. Escreva NEW e faça um programa para calcular 64×4.

O sinal de multiplicação é um asterisco (*).

9. Introduza e faça RUN ao seguinte programa:

Escreva NEW antes de começar um novo programa.

```
10 PRINT "ELEFANTES";
20 PRINT "ELEGANTES"
30 PRINT "ENORMES",
40 PRINT "HIPOPOTAMOS"
50 END
```

Assegure-se de que os sinais de pontuação estão fora das aspas.

10. Agora escreva e faça RUN a um programa com quatro declarações PRINT e que forneça o *output* seguinte:

Escreva NEW.

```
GIRAFASVERDES
IAQUE      AMARELO
```

11. Escreva e faça correr um programa que imprima a frase OS COMPUTADORES SÃO DIVERTIDOS em duas zonas da primeira linha do *output* e os COMPUTADORES SÃO FÁCEIS em duas zonas da segunda linha do *output*. O *output* final terá o seguinte aspecto:

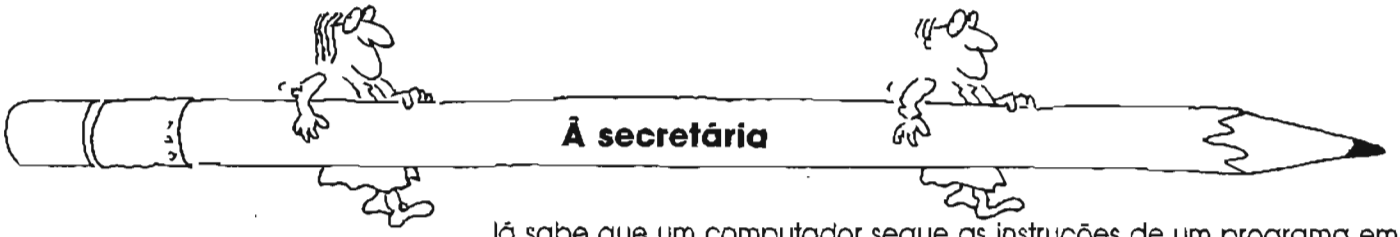
Escreva NEW.

```
OS COMPUTADORES     SAO DIVERTIDOS
OS COMPUTADORES     SAO FACEIS
```

12. Use declarações PRINT para produzir o seu próprio modelo, com referência ao programa que escreveu na pág. 100.

Escreva NEW.

Capítulo 16 SALTITANDO PELOS PROGRAMAS



Já sabe que um computador segue as instruções de um programa em BASIC começando na linha de número mais baixo e continuando pelos números imediatamente a seguir até ao mais alto. Mas há em BASIC uma declaração que dá instruções ao computador para sair da ordem normal da numeração do programa. Vejamos qual é e como funciona.

Observemos o programa seguinte. Na linha 20 há uma instrução que ainda não conhecemos:

GOTO é uma instrução BASIC.

```
10 PRINT "OLA"
20 GOTO 40
30 PRINT "TENHA UM DIA FELIZ"
40 PRINT "ADEUS"
50 END
```

Na linha 20, a nova instrução GOTO significa «ir para» (em inglês *go to*). GOTO 40 significa ir para a linha 40, portanto uma vez «lida» na linha 20 essa instrução, o computador «salta» para a linha 40, por cima da linha 30, que o computador nunca vai «ler». TENHA UM DIA FELIZ *nunca* será impresso. O programa e o respectivo *output* serão:

O computador salta da linha 20 para a linha 40.

```
10 PRINT "OLA"
20 GOTO 40
30 PRINT "TENHA UM DIA FELIZ"
40 PRINT "ADEUS"
50 END
RUN
OLA
ADEUS
```

GOTO pode fazer o computador saltar por todo o programa, tanto «para a frente» como «para trás». Veja se é capaz de dizer qual vai ser o *output* do programa seguinte:

Imagine que é o computador. Siga as instruções.

```
10 PRINT "POSSO"
20 GOTO 50
30 PRINT "EM TODAS AS DIRECCOES!"
40 GOTO 70
50 PRINT "SALTAR"
60 GOTO 30
70 END
```

Você acertou se pensou que o *output* seria:

```
POSSO
SALTAR
EM TODAS AS DIRECCOES!
```

Este programa faz com que o computador leia as respectivas linhas pela ordem seguinte: 10, 20, 50, 60, 30, 40, 70. Verifiquemos o programa fazendo a respectiva análise:

A linha 10 diz ao computador para imprimir POSSO.
A linha 20 manda o computador para a linha 50.
A linha 50 diz ao computador para imprimir SALTAR.
A linha 60 manda o computador para a linha 30.
A linha 30 diz ao computador para imprimir EM TODAS AS DIRECCOES!
A linha 40 manda o computador para a linha 70.
A linha 70 finaliza o programa.

Se não se tiver cuidado quando se usa uma declaração GOTO, poder-se-á chegar a um programa como o seguinte:

Pode dizer já qual é o *output*?

```
10 PRINT "POR VEZES"
20 PRINT "NUNCA CHEGAMOS"
30 GOTO 20
40 PRINT "AO FIM"
50 END
```

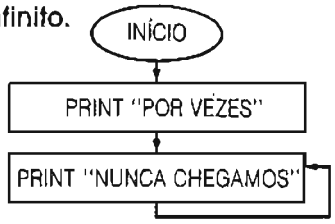
Se analisarmos as instruções cuidadosamente veremos que o computador nunca chega ao fim do programa. Se tentar escrever o *output* verá que não conseguirá parar (e o computador também não!).

```
POR VEZES
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
NUNCA CHEGAMOS
```

Infinito significa «sem fim».

Como pode ver, as linhas 40 e 50 não são realmente necessárias no programa.

Quando o computador chega à linha 30, é enviado para a linha 20. Depois segue para a 30 e daí é novamente enviado para a 20 e assim sucessivamente. Chama-se a isto um **ciclo infinito**.



Olhe cuidadosamente para as setas do fluxograma. Um programa com um ciclo infinito não tem um STOP no fluxograma.

Ora quando planeamos e escrevemos um programa queremos que o computador chegue ao fim do programa; se acontece um ciclo infinito é porque há *bugs* ou incorrecção na programação.

tente responder

1. Escreva o *output* do programa seguinte:

```
1Ø PRINT "SEGUNDA-FEIRA"
2Ø PRINT "*******"
3Ø GOTO 5Ø
4Ø PRINT "TERCA-FEIRA"
5Ø END
```

2. Escreva o *output* deste programa.

```
1Ø PRINT "COMO AS ERVILHAS COM MEL"
2Ø GOTO 7Ø
3Ø PRINT "E NA FACA ELAS FICAM"
4Ø GOTO 9Ø
5Ø PRINT "QUE BEM SABEM AS ERVILHAS"
6Ø GOTO 3Ø
7Ø PRINT "TODA A VIDA ASSIM O FIZ"
8Ø GOTO 5Ø
9Ø END
```

3. Atente no seguinte programa de computador:

```
1Ø PRINT "PRESTE ATENCAO"
2Ø GOTO 1Ø
3Ø PRINT "AO ERRO"
4Ø END
```

   a. Desenhe um fluxograma deste problema.

   b. Como será o *output*?

   c. Diga qual o erro do programa.

No computador

Se fizer um erro pode apagá-lo antes de premir ENTER ou RETURN. De outra forma terá de voltar a escrever a linha.

1. Escreva e faça correr o programa seguinte. O *output* deverá ter quatro linhas.

```
10 PRINT "PATOS PATETAS"
20 PRINT "BALEIAS BONITAS"
30 PRINT "MORCEGOS MALUCOS"
40 PRINT "TARTARUGAS TOLAS"
50 END
```

Recorde que para acrescentar uma linha ao programa tem que a escrever.

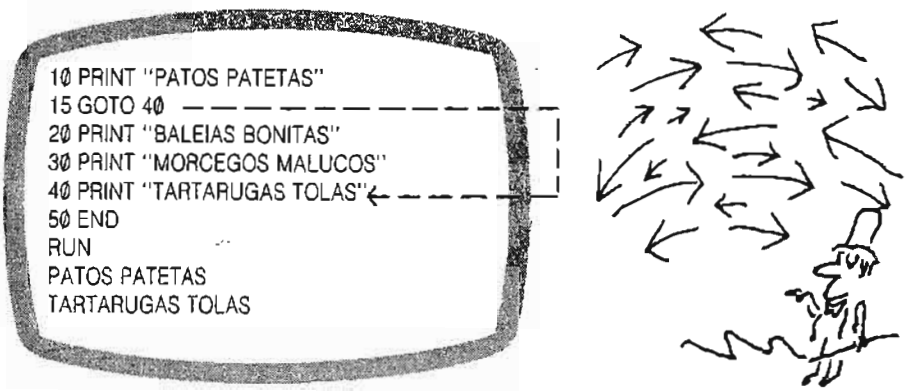
2. Agora junte a seguinte linha 15:

```
15 GOTO 40
```

Marque LIST para listar o programa.

Liste o programa. Verá que a linha 15 se vai colocar entre a 10 e a 20.

Agora corra o programa (RUN). Terá um *output* só com duas linhas, pois o computador salta por cima das linhas 20 e 30 e não imprime nem BALEIAS BONITAS nem MORCEGOS MALUCOS. Os seus programas e o *output* parecem-se com isto?



Recorde que para alterar uma linha, deve reescrevê-la.

3. Agora altere a linha 15 para

```
15 GOTO 50
```

Antes de correr o programa, imagine o que vai acontecer. Quando pensar que sabe qual o *output,* corra então o programa. Obteve o seguinte?

```
PATOS PATETAS
```

Introduza NEW para apagar o programa antigo antes de fazer este exercício.

4. Escreva este programa. Preveja o *output* e só depois corra o programa.

```
10 PRINT "OLA"
20 GOTO 50
30 PRINT "ADEUS"
40 GOTO 70
50 PRINT "VIVA"
60 GOTO 30
70 PRINT "ATE LOGO"
80 END
```

O seu *output* deverá ter as seguintes quatro linhas:

```
OLA
VIVA
ADEUS
ATE LOGO
```

5. Mude a linha 20 de forma ao *output* ser:

```
OLA
ATE LOGO
```

Tome atenção à sugestão.

Veja qual a linha que manda imprimir ATE LOGO. Altere a linha 20 de forma a enviar o computador para essa linha.

Lembre-se que **infinito** significa «sem fim».

6. O programa seguinte tem um ciclo infinito pelo que o computador irá imprimir o *output* ininterruptamente. Antes de escrever e correr o programa deverá saber como proceder para fazer parar o programa enquanto corre.

Se o seu computador não for um destes, pergunte ao seu professor como o pode parar.

Se tem um computador IBM ou compatível carregue na tecla CTRL e na tecla BREAK.

Se tem um computador Atari, prima BREAK.

Se tem um computador Commodore, prima STOP.

Se tem um computador ZX Spectrum, prima BREAK (CAPS SHIFT e SPACE).

Agora escreva e corra o seguinte programa:

```
10 PRINT "CORRER"
20 GOTO 10
30 END
```

Obterá um *output* em coluna. Sempre que o computador chega à linha 20 a declaração GOTO envia-o para a linha 10. Agora *pare* o computador e marque LIST para listar o programa no *écran*.

7. Junte ao programa a linha seguinte e faça-o correr.

```
15 PRINT "PARA SEMPRE"
```

Pode não conseguir ler todo o *output* porque o computador o imprime muito rapidamente — e continuará a imprimi-lo[17].

Pare o computador e introduza NEW para apagar o programa.

8. Pode divertir-se com saltos infinitos e sinais de pontuação. Lembra-se da função do ponto e vírgula e da vírgula numa declaração PRINT?

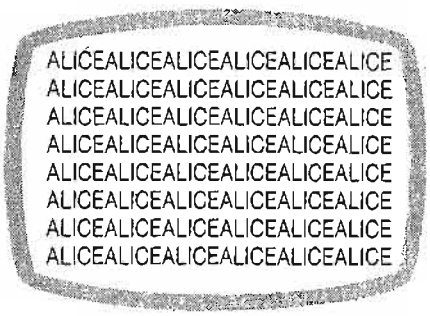
Ponha o ponto e vírgula *depois* das aspas.

Escreva e faça correr o seguinte programa:

```
10 PRINT "ALICE";
20 GOTO 10
30 END
```

← Escreva entre aspas o seu nome

Formidável! Conseguiu encher o *écran* com o seu nome? Pare o programa e faça LIST. O ponto e vírgula tem certamente grande importância. Lembre-se que indica ao computador para *continuar o output na mesma linha*. Sempre que o computador imprime o seu nome fá-lo-á imediatamente à direita do nome «antigo» da mesma linha. E quando chega ao final de uma linha continua na seguinte:



9. Pode agora escrever o seu nome de uma maneira mais fácil de ler. Altere a linha 10 para:

```
10 PRINT "ALICE ";
```

Deixe um espaço aqui.

Lembre-se que o computador escreverá tudo o que esteja entre aspas incluindo o espaço. Faça RUN ao programa. Não ficou um espaço entre cada nome? Isso porque se disse ao computador para imprimir um espaço — pondo-o entre aspas. Pare o computador e faça LIST.

10. Vejamos o efeito de uma vírgula. Altere a linha 10 para

```
10 PRINT "ALICE",
```

Faça RUN ao programa e depois pare-o. Lembre-se de que uma vírgula diz ao computador para *continuar o output na zona seguinte.* No primeiro ciclo do programa o computador escreveu ALICE na primeira zona, no segundo ciclo na segunda zona, no terceiro ciclo na terceira zona e assim sucessivamente. Quando não há mais zonas o computador continua a imprimir na linha seguinte. E o seu nome aparece escrito em várias colunas no *écran*.

Lembre-se que deve escrever o *seu* nome.

Recorde que diferentes computadores têm diferentes números de zonas.

Alguns computadores avançam o início de cada nova linha quando imprimem por zonas. Pode por isso não obter colunas alinhadas.

11. Introduza NEW. Escreva um programa para imprimir uma coluna a toda a altura do *écran* com BUU (use como guia o exercício 6).

12. Introduza NEW. Escreva um programa que faça o computador continuar a imprimir a palavra em zonas. Escolha a palavra que quer imprimir. Use uma declaração PRINT com uma vírgula.

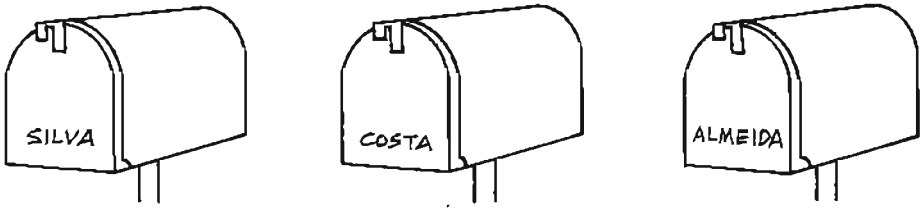
13. Introduza NEW. Escreva um programa para imprimir linhas de estrelas por todo o *écran*, usando apenas uma única estrela no programa.

Que sinal de pontuação deverá usar?

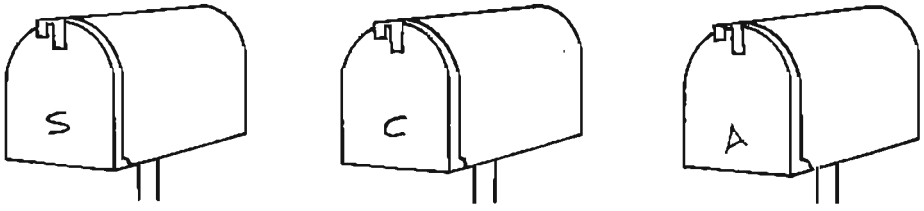
Capítulo 17 ENDEREÇOS

Neste capítulo iremos ver como a memória de acesso aleatório do computador armazena determinados blocos de informação. A RAM contém milhares de unidades de armazenagem chamadas endereços, a que correspondem os circuitos eléctricos de um *chip* de circuito integrado. Sendo circuitos eléctricos, estes endereços não se podem ver, mas ajuda se os imaginarmos como pequenas caixas de correio.

Na rua onde mora as caixas de correio têm nomes tais como:



Os endereços do computador também podem ter nomes, sendo cada um deles, geralmente, uma letra do alfabeto.



A utilização da instrução LET

LET é uma instrução em linguagem BASIC.

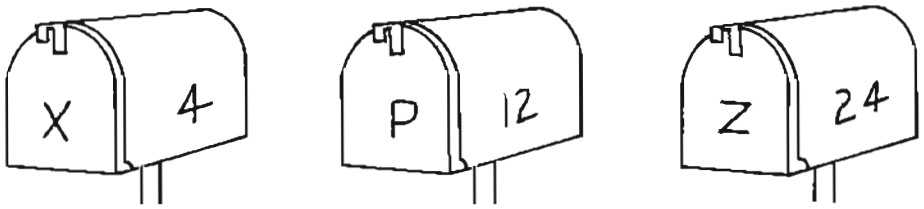
Considere a seguinte instrução:

```
10 LET X=4
```

Isto significa que vai chamar X a um dos endereços do computador e aí armazenar um 4. Assim sendo qual pensa ser o significado das instruções seguintes?

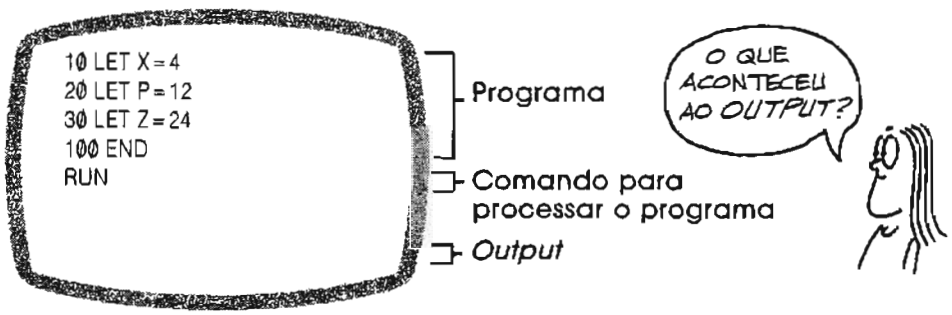
```
20 LET P=12
30LET Z=24
```

A linha 20 significa que um outro endereço será chamado P e aí estará armazenado o valor 12. E a linha 30 significa que no endereço chamado Z está armazenado o valor 24. Para recordar como funciona a instrução LET tente imaginar que os endereços do computador são «caixas de correio», que três delas são chamadas X, P e Z e que em cada um delas estão guardados os números 4, 12 e 24.



Façamos um programa de computador com aquelas três linhas. No *écran* do computador o programa e o seu *output* teriam o seguinte aspecto:

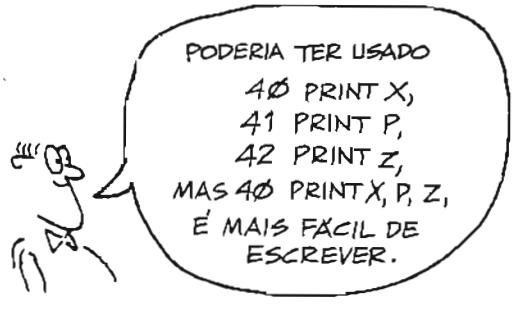
O nome de um endereço fica do lado *esquerdo* do sinal de igual. O *número* aí armazenado indica-se do lado *direito*.



Recorde-se que todos os dados estão armazenados na RAM.

Não há nada no *output*! Porquê? O computador não fez nada? O computador só faz o que nós lhe dizemos para fazer. *Armazenou* 4 no endereço X, 12 no endereço P e 24 no endereço Z. Mas no *écran* não se vê qualquer *output*. A instrução LET significa «armazena» e, portanto, não produz qualquer resultado visível no *écran*. Precisará de uma instrução PRINT se pretender um resultado visível. Se quiser ver o que está armazenado nos endereços terá de juntar uma linha 40 ao programa. Vejamos então como aquele ficava e o que se obtinha.

```
10 LET X=4
20 LET P=12
30 LET Z=24
40 PRINT X, P, Z
100 END
RUN
4          12          24
```



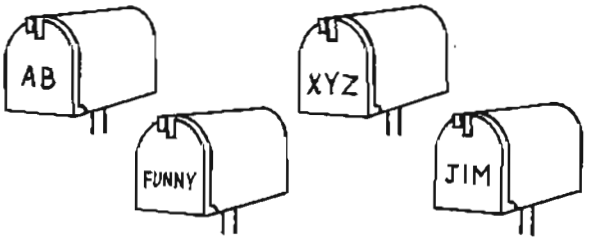
A linha 40 dá instruções ao computador para imprimir o *número* armazenado em X, o *número* armazenado em P e o *número* armazenado em Z.

Como é que o computador sabia que devia imprimir os números armazenados nos endereços X, P e Z e não as letras X, P e Z. Juntemos uma linha 50 ao programa para ver se consegue encontrar a resposta.

```
10 LET X=4
20 LET P=12
30 LET Z=24
40 PRINT X, P, Z
50 PRINT "X", "P", "Z"
100 END
RUN
4          12          24
X          P           Z
```

Como pode ver, na linha 50 há aspas envolvendo as letras e a linha 40 não as tem. Assim, o *output* da linha 50 são as três letras entre aspas — X, Y, Z. Na linha 40 são os valores armazenados em cada um destes endereços — 4, 12 e 24.

Até agora os endereços foram identificados com uma letra do alfabeto. Muitos computadores, no entanto, permitem usar mais do que uma letra.

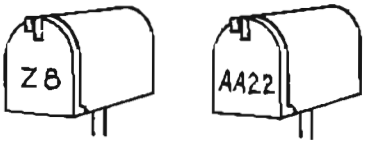


10 LET XYZ=9

O nome de um endereço não deve conter palavras da linguagem BASIC. Por exemplo não deve usar para identificar uma posição de endereço a palavra LETRA porque ela contém a palavra LET.

Aqui está um exemplo.

E também podemos usar letras e números para identificar um endereço.



10 LET K32L=15

Um endereço deve começar sempre por uma letra e nunca por um número. B2 pode ser um endereço mas 2B não.

Eis um exemplo.

Num programa escrito por outras pessoas pode encontrar endereços com mais do que uma letra. Neste livro usaremos apenas letras isoladas para assim se evitarem erros na escrita de programas.

1. Escreva o *output* de cada programa:

   a.
   ```
   10 LET A=12
   20 PRINT A
   30 END
   ```

   b.
   ```
   10 LET A=12
   20 PRINT A
   30 PRINT "A"
   40 END
   ```

2. No programa 1b explique a diferença entre as linhas 20 e 30.

3. Que erro existe no programa seguinte? Diga como o emendar.

   ```
   10 LET 25=S
   20 LET T=33
   30 PRINT S, T
   40 END
   ```

4. a. Escreva o *output* do programa seguinte.

   ```
   10 LET G=2
   20 LET H=4
   30 PRINT "O NUMERO ARMAZENADO EM G E"
   40 PRINT G
   50 END
   ```

   b. Porque é que o número 4 não aparece no *output*?

5. Escreva um programa para armazenar 5 no endereço D e 8 no endereço E. O *output* deverá ser:

   ```
   D       E
   5       8
   ```

A utilização de números armazenados

Vejamos agora como podemos usar números atribuídos a endereços. Este programa mostra-lhe como fazer cálculos com estes números.

Olhe para as linhas 30 e 40.

```
10 LET A=14
20 LET B=7
30 PRINT A+B, A-B
40 PRINT A*B, A/B
50 END
RUN
21          7
98          2
```

Se não houver aspas a rodearem A+B o computador calcula e imprime a resposta.

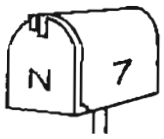
A linha 30 diz ao computador para calcular e imprimir os valores de A+B na zona 1 e de A−B na zona 2. Para calcular A+B o computador «olha» para o endereço A e «vê» 14; depois «olha» para B e «vê» 7; então soma 14 a 7 e imprime a resposta 21. Para calcular A−B o computador segue idêntica via. Na linha 40 o computador calcula A*B e A/B pelo mesmo processo e imprime as respostas.

Aquilo que está atribuído a uma posição de endereço pode ser alterado:

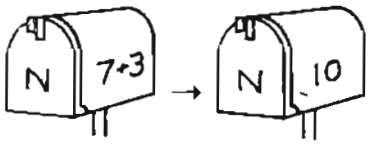
```
10 LET N=7
20 PRINT N
30 LET N=N+3
40 PRINT N
50 END
RUN
 7
 10
```

Você sabe que a linha 10 indica ao computador para armazenar 7 no endereço N; a linha 20 indica a impressão do que estiver registado em N. E quanto à linha 30? Que significado terá? A linha 30 indica ao computador que deve armazenar um novo número em N, partindo do já existente 7 — e adicionando 3; logo, em N fica armazenado 10 ou, por outras palavras, substitui-se o 7 por 10. A linha 40 diz ao computador para imprimir o que estiver em, N, que é 10.

LET N=7



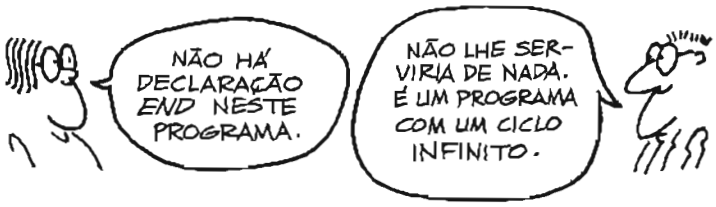
LET N=N+3



A um endereço também se chama *variável*. No programa acima indicado N é uma variável, ou seja alguma coisa que pode mudar: na linha 10 a variável N é 7; na linha 30 passa a ser 10.

O número registado num endereço, ou seja, o valor da variável, pode ser alterado as vezes que se desejar. O programa seguinte é um exemplo de um programa que altera sucessivamente o valor atribuído a uma variável.

```
10 LET S=1
20 PRINT S
30 LET S=S+1
40 GOTO 20
```



Façamos a análise do programa.

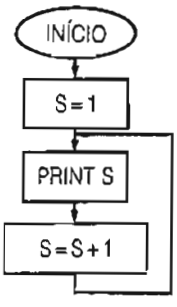
A linha 10 armazena 1 no endereço S.

A linha 20 imprime o que está armazenado em S que é 1.

A linha 30 adiciona 1 ao conteúdo do endereço S que assim fica a ser 1+1=2.

A linha 40 indica ao computador para regressar à linha 20.

A linha 20 imprime o que está armazenado em S que é agora 2, etc.



O *output* será uma coluna de números que «rolam», vindos da margem inferior do *écran*[18]:

1
2
3
.
.
.

Como pode ver o computador continua sempre a adicionar 1 à variável S e a imprimir o novo número. Há um *ciclo infinito* no programa que mantém o computador a contar ininterruptamente. *Mas,* os computadores *têm de ter*

O ciclo infinito é entre a linha 40 e a linha 20

um limite. Apenas funcionam com números até uma certa dimensão e então param. Alguns microcomputadores trabalham com números de comprimento até 38 dígitos.

1. Escreva o *output* do programa seguinte:

```
10 LET X=12
20 LET Y=4
30  PRINT X+Y, X-Y
40 PRINT X*Y, X/Y
50 END
```

2. Escreva o *output*:

```
10 LET S=24
20 PRINT S
30 LET S=S/6
40 PRINT S
50 LET S=S*4
60 PRINT S
70 LET S=S-10
80 PRINT S
90 END
```

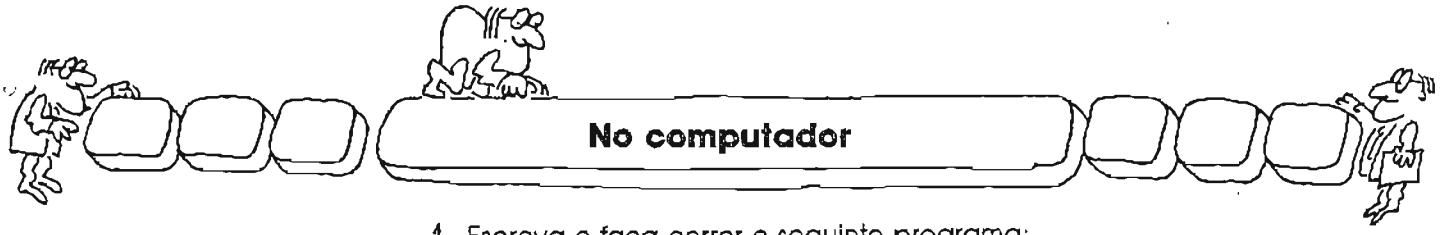
As linhas 30, 50 e 70 modificam o número armazenado em S.

3. Escreva o *output* até descobrir uma sequência. Que tipo de números representa o *output*?

```
10 LET A=0
20 PRINT A
30 LET A=A+2
40 GOTO 20
```

Se há quatro instruções PRINT deverá haver quatro linhas de *output*.

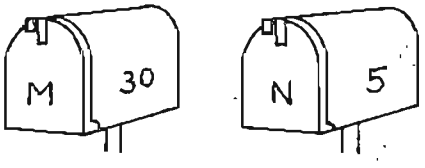
Não há sinal de pontuação na linha 20 pelo que o *output* será uma coluna de números.



No computador

1. Escreva e faça correr o seguinte programa:

```
10 LET M=30
20 LET N=5
30 PRINT "M", "N"
40 PRINT M, N
100 END
```



O *output* será parecido com:

```
M      N
30     5
```

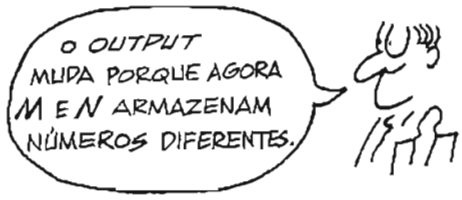
2. Junte ao programa as duas linhas seguintes. Marque LIST para ordenar as linhas.

```
50 PRINT M+N
60 PRINT M*N
```

Agora corra o programa. Obterá o seguinte *output*:

```
M      N
30     5
35
150
```

30+5=35
30*5=150



3. Altere as linhas 10 e 20 para:

```
10 LET M=100
20 LET N=50
```

Pode dizer qual será o novo *output*? Corra o programa.

4. Muitos computadores permitem a omissão da palavra LET numa instrução. Neles A=5 é o mesmo que LET A=5.

Introduza NEW para apagar o programa antigo.

Escreva e faça correr o seguinte programa:

```
10 A=375
20 B=961
30 C=A+B
40 PRINT "A SOMA DE "; A; " COM "; B; " E IGUAL A "; C
50 END
```

Deixe espaços como os aqui indicados entre as palavras do *output*.

O *output* parece-se com o resultado que a seguir se indica? Se não, volte atrás e reveja o programa.

```
A SOMA DE 375 COM 961 É IGUAL A 1336
```

5. Altere as linhas 10 e 20 do programa anterior para:

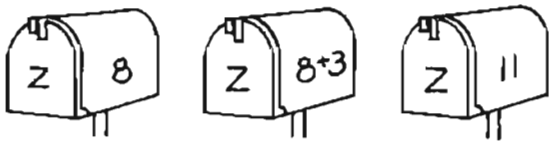
```
10 A=100
20 B=100
```

Corra o programa. Obteve o *output* que esperava?

6. Escolha dois números para armazenar em A e B. Altere as linhas 10 e 20 novamente de forma aos números escolhidos ficarem naqueles endereços. Corra o programa.

Recorde que primeiro deve apagar o programa antigo.

7. Escreva e corra o programa seguinte. O número armazenado em Z vai sendo alterado.




```
10 LET Z=8
20 PRINT Z
30 LET Z=Z+3
40 PRINT Z
100 END
```

A vírgula na linha 20 indica ao computador para imprimir o *output* da linha 40 na zona seguinte.

Obteve 8 na primeira zona e 11 na segunda zona?

8. Junte ao programa as linhas seguintes e faça-o correr.

```
50 LET Z=Z-5
60 PRINT Z
```

A linha 50 dá instruções ao computador para subtrair 5 de Z. O último número em Z, que era 11, passa agora a ser 11−5 ou seja 6. É este valor que a linha 60 indica ao computador para imprimir.

Se se esqueceu da maneira de parar um programa enquanto ele corre, veja a pág. 112.

9. Introduza NEW. Escreva e faça correr o programa seguinte.

```
10 LET X=1
20 PRINT X
30 LET X=X+1
40 GOTO 20
```

Que está a acontecer? Sempre que o computador passa no salto GOTO adiciona 1 ao valor que está na posição do endereço respectivo. Primeiramente X armazena 1 e o computador imprime esse valor. Depois junta mais 1, passa a registar 2 e o computador imprime esse valor e assim sucessivamente.

Escreva novamente a linha 20.

10. Que sinal de pontuação deveria juntar ao seu programa para que o *output* fosse apresentado em linha em vez de o ser em coluna? Junte um ponto e vírgula ao final da instrução PRINT da linha 20. Corra o programa de novo.

Sugestão: faça X igual ao primeiro número que quer imprimir: LET X=0

11. Escreva um programa para imprimir os números pares desde o 0. O princípio do *output* deverá ser: 0, 2, 4, 6, 8, . . .

Sugestão: o que é que é necessário para obter 99 a partir de 100? E 98 de 99? É subtrair uma unidade de cada vez.

12. Agora escreva um programa que imprima os números desde 100 em contagem regressiva de um em um: 100, 99, 98, 97, 96, etc. Verifique o que acontece aos números depois de se chegar ao zero.

Sugestão: como obter 4 a partir de 2? E 8 a partir de 4? E 16 a partir de 8? Multiplicando por 2.

13. Escreva um programa para apresentar números na seguinte sequência: 2, 4, 8, 16, 32, 64, 128. Os números deverão vir em coluna. Deixe o programa correr durante um certo tempo e verá que algo estranho começa a acontecer ao *output*. É que muitos computadores escrevem os números grandes, geralmente maiores que 1 000 000, de uma forma abreviada especial chamada **notação científica**. Um exemplo de um número escrito em notação científica é:

```
3.425E+06
```

A **notação científica** é também usada para escrever números muito pequenos — menores que 1 e muito próximos de 0.

Há uma maneira fácil de ver o que o número é: +6 depois de E significa que se pode mover a vírgula seis espaços para a direita. Assim:

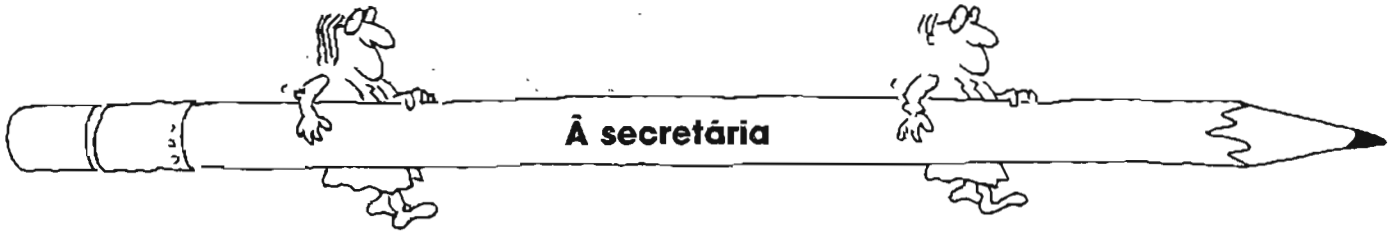
```
3.425E+06=3 425 000
          ↑
          19
```

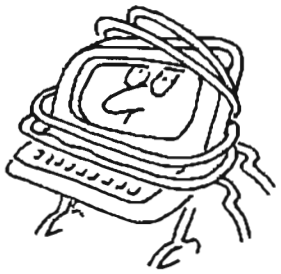
E significa «expoente». O número apresentado representa realmente $3{,}425 \times 10^6$ (10^6 é 1 000 000).

Nos nossos *outputs* não encontraremos muitas vezes notação científica, mas quando tal acontecer lembre-se que é uma maneira abreviada de escrever números muito grandes ou muito pequenos.

Capítulo 18

A FORMAÇÃO DE CADEIAS DE TEXTO





Já conhece a forma como um computador armazena números nos endereços da memória. Mas para além disso ele pode também armazenar palavras.

Chama-se **cadeia** a uma palavra ou qualquer grupo de caracteres que não seja um número. Veja em seguida alguns exemplos de cadeias:

HOJE ARCO-IRIS
COMPUTADOR FORMULA 28Ø/Z

Para se armazenar uma cadeia tem de se dar a conhecer ao computador que se vai seguir uma cadeia (e não um número). Para tal acrescenta-se um cifrão ($) no fim do nome da variável e escreve-se a cadeia entre aspas.

```
LET A$="AZUL"
LET P$="BOM DIA!"
LET C$="!!!***###"
```

Tentar armazenar uma cadeia sem aspas é criar um erro *(bug)* no seu programa.

Esta instrução lê-se: «Seja A cifrão igual a azul».

A maior parte dos computadores permite designar uma variável de cadeia (variável alfanumérica ou variável de texto) por mais do que uma letra ou número (seguido de $).[20]

```
LET D5E6$="GATOS E CAES"
```

Não é possível iniciar com um número um endereço destinado a uma variável de texto (cadeia). Também não é possível usar uma palavra da linguagem BASIC.

As regras existentes para variáveis numéricas também funcionam para variáveis de texto. Por isso neste livro e para evitar erros na identificação das cadeias, todos os endereços respectivos serão designados por uma única letra seguida de cifrão.

As variáveis numéricas e as variáveis de texto ocupam posições inteiramente diferentes na memória do computador. Veja o programa seguinte:

```
10 LET A=25
20 LET A$="RABISCOS"
30 PRINT A, A$
40 END
RUN
25            RABISCOS
```

Lembre-se que a identificação do endereço fica do lado esquerdo do sinal de igual.

Verifiquemos o programa ou seja façamos a respectiva análise.

A linha 10 armazena o valor 25 na posição de endereço A, mas não imprime nada.

A linha 20 armazena "RABISCOS" na posição de endereço A$, mas igualmente não imprime nada.

A linha 30 imprime o conteúdo de A na primeira zona e o de A$ na segunda zona.

Como se viu as variáveis A e A$ não são as mesmas.

Esta secção é destinada aos utilizadores dos computadores ATARI.

Se tiver um microcomputador Atari terá de tomar atenção às instruções particulares que lhe vamos dar. Para usar cadeias deve previamente reservar espaço para elas, o que se consegue iniciando o programa com uma declaração como:

```
5 DIM A$(20)
```

DIM significa «dimensão» e é uma instrução BASIC.

Esta declaração vai reservar na memória do computador o espaço suficiente para que se possa vir a atribuir à cadeia, denominada A$, até 20 caracteres. Isto significa que a cadeia A$ pode ter qualquer número de caracteres até 20 (pode ter menos, não pode é ter mais). Ao escolher o valor da dimensão máxima da cadeia deixe sempre um espaço «extra». Se pensa que A$ pode precisar de uma cadeia maior, ponha entre parêntesis um número maior, por exemplo:

Não exagere e não use um número muito grande. Pode ocupar demasiado espaço da RAM e ficar sem espaço suficiente para o resto do programa.

```
5 DIM A$(50)
```

que lhe vai permitir armazenar até 50 caracteres em A$.

A reserva de espaço para várias cadeias que eventualmente tenha de utilizar no programa pode ser feita com uma única declaração DIM:

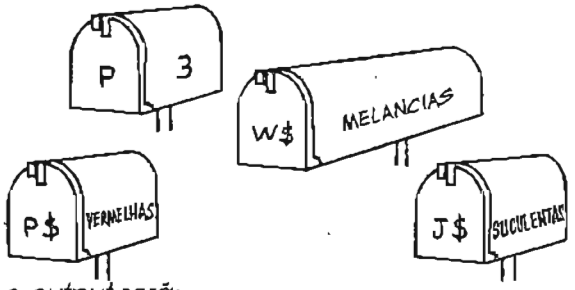
```
5 DIM A$(20), B$(15), C$(30)
```

Esta secção volta a ser para todos os leitores.

Com declarações PRINT podemos imprimir cadeias em qualquer ordem. Vejamos o programa seguinte:

Os utilizadores de computadores Atari devem juntar a seguinte linha:
5 DIM M$(15), S$(15), V$(15)

```
10 LET P=3
20 LET M$=" MELANCIAS"
30 LET S$=" SUCULENTAS"
40 LET V$=" VERMELHAS"
50 PRINT P; S$; M$; V$
60 END
```

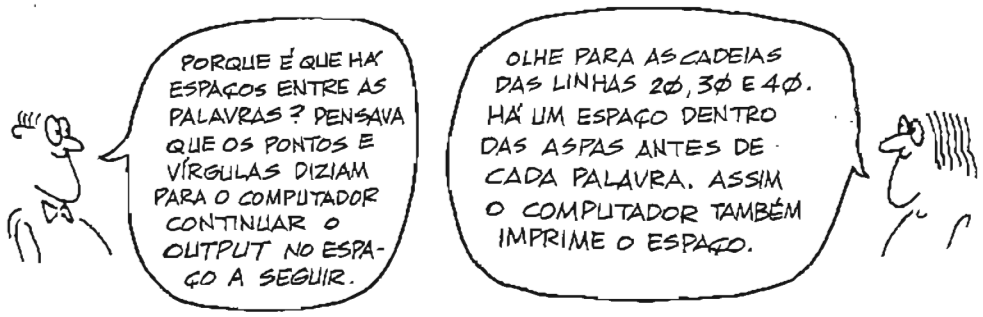


Lembre-se que não tem de usar a instrução LET.

Fazendo correr o programa o *output* será:

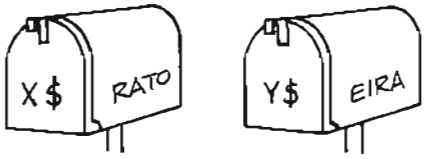
```
3 SUCULENTAS MELANCIAS VERMELHAS
```

correspondendo à ordem com que os diferentes elementos são indicados na linha 50: primeiro o valor atribuído a P, depois a cadeia armazenada em S$, depois a de M$ e finalmente a de V$, tudo na mesma linha.



Também é possível juntar cadeias utilizando o sinal (+). Vejamos:

```
10 LET X$="RATO"
20 LET Y$="EIRA"
30 PRINT X$+Y$
40 PRINT Y$+X$
50 END
RUN
RATOEIRA
EIRARATO
```



Este processo não se aplica no computador Atari. Se tiver um, use um ponto e vírgula (;) em vez de (+) nas linhas 30 e 40.

A linha 30 indica ao computador que deve juntar as cadeias armazenadas em X$ e Y$ e imprimi-las. A linha 40 significa que as cadeias armazenadas em Y$ e X$ se juntam e são impressas por esta ordem. A operação de juntar cadeias com o sinal + chama-se **concatenação.**

Concatenar significa «ligar».

Também é possível armazenar cadeias concantenadas em endereços. Veja o seguinte programa:

```
10 LET X$="RATO"
20 LET Y$="EIRA"
30 LET Z$=X$+Y$
40 PRINT Z$
50 END
```

Se tem um computador Atari omita a linha 30 e altere a linha 40 para PRINT X$; Y$

A linha 30 indica ao computador para armazenar X$+Y$ ou RATOEIRA no endereço Z$. A linha 40 faz o computador imprimir o que estiver em Z$, pelo que o *output* será:

```
RATOEIRA
```

tente responder

1. Cada uma das instruções seguintes tem um erro. Descubra-o e escreva a instrução correctamente:

   a. LET C$=CENOURAS
   b. "COMPUTADOR"=A$
   c. Z="ZEBRA"

2. Escreva o *output*:

```
10 LET A=100
20 LET A$=" MENINOS FRANCESES"
30 PRINT A; A$
40 END
```

Os utilizadores de computadores Atari devem acrescentar:
5 DIM A$(20)

3. Use, na linha 50 do programa seguinte, uma instrução PRINT com os sinais de pontuação correctos por forma ao *output* ser ASPALAVRASESTAOJUNTAS.

```
10 LET P$="PALAVRAS"
20 LET E$="ESTAO"
30 LET A$="AS"
40 LET J$="JUNTAS"
50
60 END
```

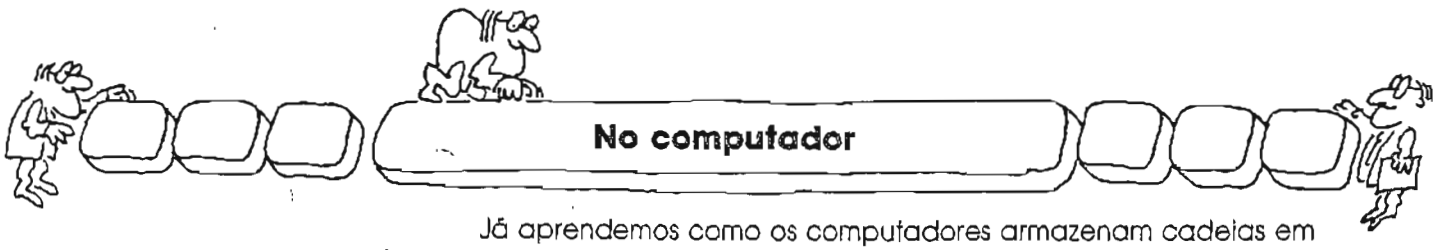
Os utilizadores de computadores Atari devem acrescentar:
5 DIM P$(10), E$(7), A$(4), J$(8)

Os computadores Atari não podem usar o sinal (+) para concatenação.

4. Escreva o *output* do programa seguinte. Os utilizadores de computadores Atari poderão saltar este problema.

```
10 LET X$="AS CADEIAS SÃO"
20 LET Y$=" FACEIS"
30 LET Z$=" ALEGRES"
40 LET A$=X$+Y$
50 LET B$=X$+Z$
60 PRINT A$
70 PRINT B$
80 END
```

Note os espaços nas linhas 20 e 30. No *output* as palavras não vão sair pegadas.



No computador

Já aprendemos como os computadores armazenam cadeias em endereços ou variáveis de cadeias. Agora vamos praticar com algumas cadeias no seu computador!

1. Escreva e faça correr o seguinte programa:

```
10 LET A$="LAPIS"
20 LET B$="CANETAS"
30 LET C$="BORRACHAS"
```

Os utilizadores dos computadores Atari deverão juntar:
5 DIM A$(10), B$(10), C$(10)

Que obteve no *output*? Apenas *armazenou* cadeias em endereços, sem dizer ao computador para imprimir coisa alguma. Agora junte ao programa as seguintes linhas e corra-o:

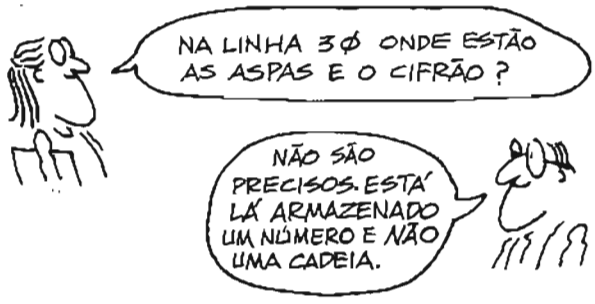
```
40 PRINT A$, B$, C$
50 PRINT C$, A$, B$
60 PRINT B$, C$, A$
```

O *output* será:

```
LAPIS        CANETAS      BORRACHAS   ← output da linha 40
BORRACHAS    LAPIS        CANETAS     ← output da linha 50
CANETAS      BORRACHAS    LAPIS       ← output da linha 60
```

Já notou que o programa não tem declaração END? Quando não há mais instruções a seguir, o programa automaticamente termina.

2. Antes de escrever e fazer correr o programa, introduza NEW para apagar o antigo.




```
1Ø LET Q$=" PEPERONI"
2Ø LET R$=" PIZZAS"
3Ø LET S=13
4Ø PRINT S; R$; Q$
```

Os utilizadores de computadores Atari acrescentarão:
5 DIM Q$(10), R$(10)

Assegure-se que inclui os espaços indicados.

Obteve este *output*?

```
13 PIZZAS PEPERONI
```

Primeiro apague o programa antigo.

3. Também se podem usar símbolos em cadeias. Escreva e faça correr o seguinte programa:

Os utilizadores dos computadores Atari acrescentarão:
5 DIM L$(50); C$(20)

```
1Ø LET L$="**********CADEIA LONGA**********"
2Ø LET C$="*CADEIA CURTA*"
3Ø PRINT C$
4Ø PRINT L$
```

Nota para os utilizadores de computadores Atari: Não verá a partir de agora e até ao fim do livro qualquer nota a lembrar que deve acrescentar declarações DIM no início dos seus programas. Sempre que usar cadeias num programa, lembre-se que deve *dimensioná-las*, ou seja, reservar-lhes espaço. Comece os seus programas com uma declaração DIM.

Introduza NEW.

4. Pode-se alterar o que está registado num endereço de cadeias substituindo-o por uma nova cadeia. Escreva e corra o seguinte programa:

Recorde-se que deve verificar possíveis erros de dactilografia antes de fazer o ENTER de cada linha.

```
1Ø LET S=5
2Ø LET S$=" GIRINOS PEQUENINOS"
3Ø PRINT S; S$
4Ø LET S$=" GRANDES RAS"
5Ø PRINT S; S$
```

O computador na linha 30, ao imprimir S e S$ verifica que S é 5 e S$ é GIRINOS PEQUENINOS. Na linha 50 quando o computador imprime S e S$, S ainda é 5 mas S$ é GRANDES RAS.

Introduza NEW.

5. Lembre-se que se pode **concatenar** ou «adicionar» cadeias. Escreva o programa seguinte e tente adivinhar o *output* antes de o correr.

Os utilizadores de computadores Atari omitirão as linhas 40 e 50 e alterarão as linhas 60 e 70 para:
60 PRINT A$; C$; B$
70 PRINT B$; C$; A$

```
1Ø LET A$=" BAGAS"
2Ø LET B$=" NOZES"
3Ø LET C$=" E"
4Ø LET X$=A$+C$+B$
5Ø LET Y$=B$+C$+A$
6Ø PRINT X$
7Ø PRINT Y$
```

Introduza NEW.

6. Escreva um programa que registe o seu primeiro nome num endereço e o último noutro. Usando endereços dê instruções ao computador para imprimir um *output* como o seguinte:

MIGUEL CAMPOS
CAMPOS MIGUEL

Use os sinais de pontuação adequados para os nomes serem impressos por zonas.

Os utilizadores de computadores Atari deverão juntar uma instrução PRINT.

7. Junte ao programa anterior uma linha de forma a **concatenar** os seus primeiro e último nomes num novo endereço. Depois acrescente uma nova linha para imprimir o conteúdo do novo endereço.

Introduza NEW.

8. Escreva um programa para armazenar cada uma das seguintes palavras em vários endereços:

DIAMANTE LARANJA CANETA

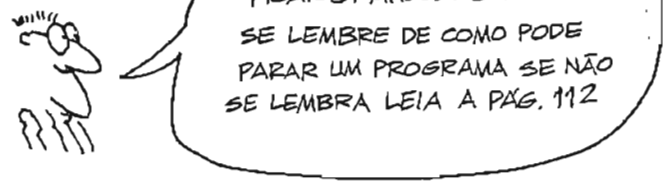
Depois dê instruções ao computador para imprimir estas palavras por ordem alfabética em três zonas.

Que sinal de pontuação fará aparecer o *output* em zonas?

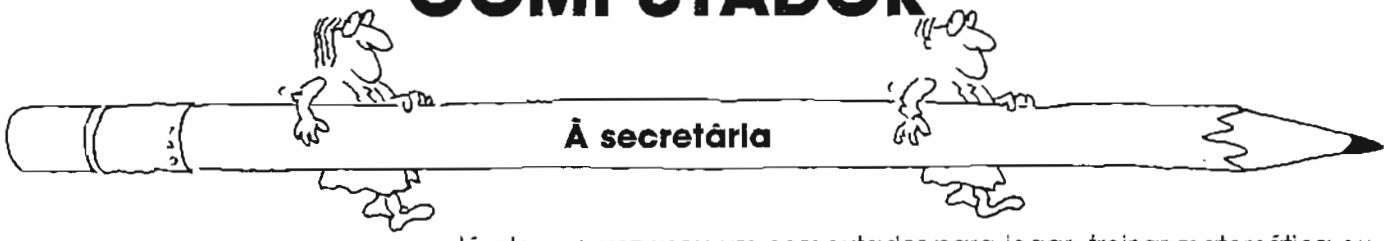
9. Neste programa vai utilizar instruções que estudou atrás. Escreva NEW. Escreva e corra o seguinte programa:

```
10 LET X=1
20 LET X$=" AS CADEIAS SAO DIVERTIDAS"
30 PRINT X; X$
40 LET X=X+1
50 GOTO 30
```

Enquanto corre, verifique o programa (faça a respectiva análise) e perceba o que está a acontecer. O papel da linha 50 é fazer o computador regressar à linha 30. Cada vez que faz este salto o computador adiciona uma unidade a X e imprime o número registado no endereço X e a cadeia armazenada em X$.
Pare o programa.



Capítulo 19 CONVERSANDO COM O COMPUTADOR



Já alguma vez usou um computador para jogar, treinar matemática ou resolver um teste? Se sim, sabe que o computador por vezes faz perguntas e aguarda as suas respostas. Neste capítulo iremos aprender como o computador aceita as suas respostas, embora elas façam parte do próprio programa.

Veja o programa seguinte e repare na nova instrução constante da linha 20.

INPUT é uma instrução na linguagem BASIC

```
10 PRINT "COMO SE CHAMA?"
20 INPUT N$
40 END
RUN
COMO SE CHAMA?
?
```

Já sabe que a linha 10 imprime a mensagem COMO SE CHAMA? no *écran*. Na linha 20 aparece uma nova instrução chamada INPUT que indica ao computador que deve *parar, imprimir um ponto de interrogação no écran* e *esperar* que o **utilizador** escreva a resposta. Esta vai ser registada na variável N$. Vamos supor que o Nuno ao chegar junto ao seu computador vê no *écran* o *output* acima representado, pelo que escreve o seu nome — NUNO. O *écran* do computador passou a ter o seguinte aspecto:

O **utilizador** é a pessoa que usa o programa. Pode ser quem escreve o programa ou outra pessoa.

```
10 PRINT "COMO SE CHAMA?"
20 INPUT N$
40 END
RUN
COMO SE CHAMA?
?NUNO
```

O computador imprimiu o ponto de interrogação.

O Nuno escreveu o seu nome.

Depois do Nuno carregar na tecla ENTER (ou RETURN), o computador segue para a linha seguinte que finaliza o programa.

Os utilizadores dos computadores Atari devem recordar-se que precisam de incluir no seu programa uma instrução DIM (de DIMensão) para N$.

Tornemos agora o programa mais interessante acrescentando uma linha 30 de tal forma que, quando corre, fica com este aspecto:

```
10 PRINT "COMO SE CHAMA?"
20 INPUT N$
30 PRINT "OLA, "; N$
40 END
RUN
COMO SE CHAMA?
?NUNO
OLA, NUNO
```

Depois do Nuno escrever o seu nome e de carregar em ENTER o computador, que registou o nome introduzido em N$, segue para a linha 30; aí imprime OLA, uma vírgula e um espaço. O ponto e vírgula da linha 30 faz com que o computador continue o *output* na mesma linha imprimindo o que tem armazenado em N$ — NUNO; como resultado temos o *output* final OLA, NUNO.

Suponha agora que o Rui faz correr o mesmo programa. Ao ler no *écran* a pergunta COMO SE CHAMA?, escreve o seu nome — Rui que é registado na variável N$. E o *output* será:

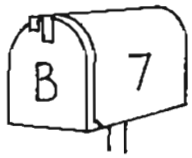
```
COMO SE CHAMA?
?RUI
OLA, RUI
```

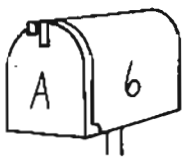
O Nuno, o Rui e todos os que fizerem correr o programa são os utilizadores. A instrução INPUT permite a um utilizador armazenar informação em endereços (variáveis), quando o programa corre. Quando se escreve o programa nada é guardado em N$ e nada aí é colocado até o programa correr e o utilizador fazer esses registos.

Eis um outro programa que usa instruções INPUT. Veja o programa e o *output*.

As linhas 20 e 40 não precisam de $ pois a resposta que se pede é um **número** e não uma cadeia de caracteres.

```
10 PRINT "ESCREVA UM NUMERO"
20 INPUT A
30 PRINT "ESCREVA OUTRO NUMERO"
40 INPUT B
50 PRINT "A SOMA DOS NUMEROS É "; A+B
RUN
ESCREVA UM NUMERO
?6
ESCREVA OUTRO NUMERO
?7
A SOMA DOS NUMEROS E 13
```





Outro *output* poderia ser:

```
ESCREVA UM NUMERO
?123
ESCREVA OUTRO NUMERO
?234
A SOMA DOS NUMEROS E 357
```

Analisemos o programa:

A linha 10 indica ao computador que imprima no *écran* ESCREVA UM NUMERO.

A linha 20 indica ao computador para parar e esperar que o utilizador escreva a resposta. Esta será registada em A.

A linha 30 indica ao computador para imprimir no *écran* ESCREVA OUTRO NUMERO.

A linha 40 indica ao computador para parar e esperar que o utilizador escreva outra resposta, que irá ser armazenada em B.

A linha 50 indica ao computador para imprimir A SOMA DOS NUMEROS E. O ponto e vírgula significa que o *output* continua na mesma linha. O computador imprime a resposta A+B.

Podemos escrever um programa para um jogo de improviso usando declarações INPUT e PRINT. O jogo funciona da seguinte forma: o computador faz-lhe diversas perguntas e as suas respostas vão servir para compor uma pequena história. Suponha que as perguntas são as seguintes:

Como se chama?
Indique um verbo.
Que local gostaria de visitar?
Indique o nome de um animal.

Imagine que o seu nome é "Josefina", que tinha escolhido como verbo "saltar", como local a visitar o "Alasca" e como animal o "camelo". O computador imprimiria a seguinte mensagem:

Josefina decidiu-se a saltar todo o caminho para o Alasca num camelo!

A história muda sempre que uma pessoa diferente responde às perguntas.

Eis o programa:

Os utilizadores de computadores Atari necessitam de uma instrução DIM para N$, V$, L$ e A$.

```
10 PRINT "COMO SE CHAMA?"
20 INPUT N$
30 PRINT "INDIQUE UM VERBO"
40 INPUT V$
50 PRINT "QUE LOCAL GOSTARIA DE VISITAR?"
60 INPUT L$
70 PRINT «INDIQUE O NOME DE UM ANIMAL»
80 INPUT A$
90 PRINT
100 PRINT N$; «DECIDIU-SE A»; V$; «TODO O CAMINHO PARA O»; L$; «NUM»; A$;«!»
```

A linha 90 «imprime» uma linha de espaço.

Nalguns computadores pode-se omitir o ponto e virgula.

As linhas 10, 30, 50 e 70 imprimem as perguntas no *écran*. As linhas 20, 40, 60 e 80 dão instruções ao computador para parar e esperar pela resposta do utilizador. Cada resposta é atribuída a uma variável. A linha 90 deixa uma linha de espaço em branco. A linha 100 é uma longa declaração PRINT que iremos examinar cuidadosamente:

```
100 PRINT N$; «DECIDIU-SE A»; V$; «TODO O CAMINHO PARA O»; L$; «NUM»; A$; «!»
```

O computador imprime todas as palavras entre aspas exactamente como foram escritas por si — incluindo os espaços. Imprime o ponto de exclamação no final da frase e sempre que «lê» o nome de uma variável imprime o que estiver atribuído a essa variável.

Se fosse o Filipe a correr o programa e a responder FILIPE, NADAR, POLO NORTE e HIPOPOTAMO a resposta seria:

JOSEFINA à variável N$
SALTAR à variável V$
ALASCA à variável L$
CAMELO à variável A$

E o computador imprimiria:

JOSEFINA DECIDIU-SE A SALTAR TODO O CAMINHO PARA O ALASCA NUM CAMELO!

Se fosse o Filipe a correr o programa e a responder FILIPE, NADAR, POLO NORTE e HIPOPOTAMO a resposta seria:

FILIPE DECIDIU-SE A NADAR TODO O CAMINHO PARA O POLO NORTE NUM HIPOPOTAMO!

tente responder

1. Encontre os erros existentes no seguinte programa e diga como se pode emendar.

```
1Ø PRINT «QUAL E A SUA COR FAVORITA?»
2Ø INPUT C
3Ø PRINT «INDIQUE O SEU NUMERO DE SORTE»
4Ø INPUT N$
```

2. Escreva o *output* do programa seguinte. Como é você o utilizador considere no *output* o seu próprio nome.

```
1Ø PRINT «COMO SE CHAMA?»
2Ø INPUT N$
3Ø PRINT N$; «SABE BASIC.»
```

3. Escreva o *output* do programa seguinte. Empregue os números que entender para respostas dos utilizadores.

```
1Ø PRINT «INDIQUE UM NUMERO.»
2Ø INPUT A
3Ø PRINT «INDIQUE OUTRO NUMERO.»
4Ø INPUT B
5Ø PRINT «A SOMA DOS SEUS NUMEROS E»; A+B
6Ø PRINT «A DIFERENCA ENTRE OS SEUS NUMEROS E»; A—B.
```

4. Eis um programa de um jogo de improviso para computador. Responda às perguntas das linhas 10 a 160 e escreva o *output* para as linhas 170 a 250. Terá assim um computador «improvisador».

```
1Ø PRINT «COMO SE CHAMA?»
2Ø INPUT N$
3Ø PRINT «QUE IDADE TEM?»
```

Assegure-se que no seu *output* estão o ponto de interrogação e a resposta do utilizador.

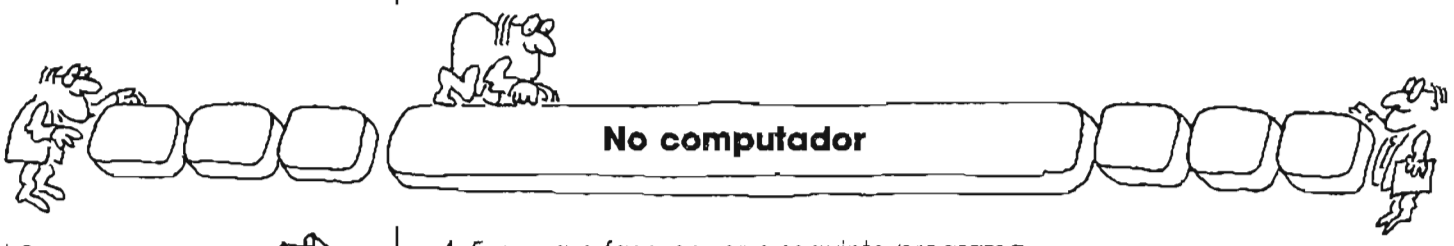
Para tornar as coisas mais fáceis escolha um número maior como primeira resposta.

Para ajudar a recordar as suas respostas, vá escrevendo o nome de cada variável e o que atribuiu a cada uma.

```
40 INPUT I
50 PRINT «DIGA O NOME DE UM GRANDE ANIMAL DE QUATRO PATAS.»
60 INPUT A$
70 PRINT «DIGA O NOME DE UMA PARTE DO SEU CORPO.»
80 INPUT C$
90 PRINT «INDIQUE UM ADJECTIVO.»
100 INPUT J$
110 PRINT «DIGA OUTRO ADJECTIVO, SE FAZ FAVOR.»
120 INPUT K$
130 PRINT «AGORA DIGA O NOME DE UM VEGETAL (NO PLURAL).»
140 INPUT V$
150 PRINT «FINALMENTE, INDIQUE O APELIDO DE UMA PESSOA FAMOSA.»
160 INPUT F$
170 PRINT «AGORA PARA O SEU COMPUTADOR FALANTE...»
180 PRINT
190 PRINT «CARO DR.»; F$
200 PRINT «O MEU MASCOTE E UM»; A$; «E PARECE ESTAR DOENTE.»
210 PRINT «TEM»; V$; «A CRESCER EM»; C$
220 PRINT «GERALMENTE E UM MASCOTE MUITO»; J$; «, MAS ULTIMAMENTE TEM-SE PORTADO
COMO UM»; K$
230 PRINT «POR FAVOR DIGA-ME O QUE FAZER, PORQUE TENHO»; I; «OUTROS ANIMAIS EM CASA
E A MINHA MAE ESTA A FICAR LOUCA.»
240 PRINT «                    COM OS MELHORES CUMPRIMENTOS.»
280 PRINT «                    »; N$
```

Uma instrução PRINT sem nada à frente «imprime» uma linha em branco.

No seu *output* não se esqueça de incluir os espaços em branco dentro das aspas.



No computador

Os utilizadores de computadores Atari necessitarão de escrever 5 DIM Y$(30)

Assegure-se que na linha 30, deixa um espaço depois da vírgula.

1. Escreva e faça correr o seguinte programa:

```
10 PRINT «OLA, SOU O TEU AMIGO COMPUTADOR. QUEM ES TU?»
20 INPUT Y$
30 PRINT «PRAZER EM CONHECER-TE, »; Y$
```

Depois do computador imprimir OLA, SOU O TEU AMIGO COMPUTADOR. QUEM ES TU?, aparece no *écran* um ponto de interrogação e o computador fica a aguardar a sua resposta, que irá ser armazenada em Y$. Então o computador imprime PRAZER EM CONHECER-TE, juntando o seu nome ao final da mensagem. Corra o programa diversas vezes, utilizando em cada uma delas um nome diferente.

2. Acrescente ao programa as seguintes linhas:

```
40 PRINT «QUANTOS ANOS TENS.»; Y$
50 INPUT X
60 PRINT X; «ANOS DE IDADE... QUE BOM!»
```

Corra o programa. Depois de indicar a sua idade, o computador segue para a linha 60 e imprime a mensagem correspondente, substituindo X pela sua idade.

Lembre-se de «escrever» os espaços que estão *dentro* das aspas. Não se preocupe com os que estejam fora.

Se quiser incluir anos bissextos use 356.25 em vez de 365.

Antes de mais, introduza NEW.

Introduza NEW.

Os utilizadores dos computadores Atari deverão escrever:
5 DIM N$(20); O$(30), A$(30).

Lembre-se de que se fez um erro pode voltar atrás para o apagar ou reescrever toda a linha.

Introduza NEW.

Pode usar como guia o exercício 5.

3. Agora acrescente as seguintes linhas:

Não use aspas em volta de cálculos. Você quer que o computador imprima as *respostas.*

```
70 PRINT «OU SEJAM »; X * 12; « MESES!»
80 PRINT «OU SEJAM »; X * 365; « DIAS!»
90 PRINT «OU SEJAM »; X * 365 * 24; « HORAS!»
100 PRINT «JA ESTAS BASTANTE VELHO,»; Y$
```

Na linha 70 a idade registada em X é multiplicada por 12, pois num ano há 12 meses. Na linha 80 a idade é multiplicada por 365, o número de dias que há num ano. Na linha 90 multiplica-se a idade por 365 x 24, donde resulta o número de horas que já viveu. Faça correr o programa com a sua idade correcta e ensaie depois com outras idades.

4. Escreva o seguinte programa:

```
10 PRINT «POSSO ADICIONAR QUAISQUER DOIS NUMEROS.»
20 PRINT «DE-ME UM NUMERO.»
30 INPUT X
40 PRINT «DE-ME OUTRO NUMERO.»
50 INPUT Y
60 PRINT «A SOMA DE »; X; «E »; Y; « SAO »; X + Y
```

Corra o programa diversas vezes usando de cada vez números diferentes.

5. Escreva e corra o seguinte programa para um jogo de improviso:

```
10 PRINT «COMO SE CHAMA?»
20 INPUT N$
30 PRINT «INDIQUE UM OBJECTO QUE TENHA NA ESCOLA»
40 INPUT O$
50 PRINT «O QUE E QUE QUER PARA ALMOCO?»
60 INPUT A$
70 PRINT «O MEU PROFESSOR DISSE,»
80 PRINT N$; «! TIRE ESSE »; O$ « DO SEU »; A$
90 END
```

Corra o programa diversas vezes. De cada uma das vezes pergunte a si mesmo: «O que é que o computador tem armazenado em N$, O$ e A$?». Quando chega à linha 80 o computador imprime as cadeias armazenadas em N$, O$ e A$.

6. Escreva um programa que pergunte a idade do utilizador. Depois o computador deve dizer que idade tinha o utilizador no ano passado e a idade que terá no próximo ano. Eis o exemplo para um *output* de um utilizador de 9 anos de idade.

```
QUE IDADE TEM?
?9
NO ANO PASSADO VOCE TINHA 8 ANOS.
NO PRÓXIMO ANO TERA 10 ANOS.
```

7. Escreva um programa para um jogo de improviso. Primeiro pense numa curta história ou frase com três ou quatro palavras-chave que se possam armazenar em endereços variáveis. Depois faça três ou quatro perguntas com instruções PRINT, cada uma delas seguida de uma declaração INPUT para armazenar a resposta. Depois de três ou quatro PRINTs e INPUTs, precisará de algumas instruções PRINT para escrever a «história», usando as variáveis em vez das palavras-chave.

Capítulo 20 A TOMADA DE DECISÕES PELO COMPUTADOR

Imagine que no *écran* do seu computador aparecia o seguinte *output*:

Escrito pelo utilizador.

Escrito pelo utilizador.

```
QUAL É A CAPITAL DO TEXAS?
? DALLAS
NÃO, TENTE OUTRA VEZ
? AUSTIN
MUITO BEM!
```

Como é que perante a resposta do utilizador o computador decidiu que ela era correcta? Já sabe que o computador, por si próprio, não pode tomar esse tipo de decisões, que têm de estar programadas. Neste capítulo aprenderemos a programar o computador de forma a ele responder a uma pergunta.

Veja as duas linhas de programa adiante apresentadas.

```
1Ø PRINT «GOSTA MAIS DE MARGARIDAS OU DE DENTES-DE-LEAO?»
2Ø INPUT C$
```

Já sabe que o computador imprimirá no *écran* a pergunta GOSTA MAIS DE MARGARIDAS OU DE DENTES-DE-LEAO?. Depois pára e espera que o utilizador escreva a resposta; esta vai ser atribuída à variável C$.

Recorde-se que o *utilizador* é a pessoa que faz correr o programa.

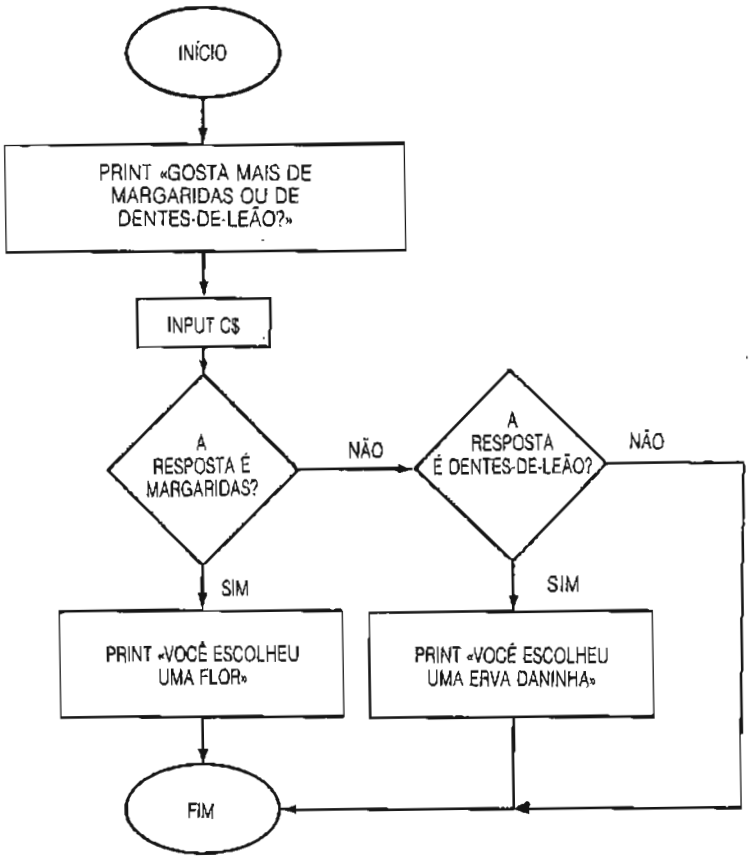
Vamos juntar mais linhas ao programa. Nas linhas 30 e 40 há uma nova instrução.

IF-THEN é uma instrução de BASIC.

```
1Ø PRINT «GOSTA MAIS DE MARGARIDAS OU DE DENTES-DE-LEAO?»
2Ø INPUT C$
3Ø IF C$ = «MARGARIDAS» THEN PRINT «VOCE ESCOLHEU UMA FLOR.»
4Ø IF C$ = «DENTES-DE-LEAO» THEN PRINT «VOCE ESCOLHEU UMA ERVA DANINHA.»
```

As linhas 30 e 40 contêm instruções IF-THEN. Este tipo de instrução é usada para indicar ao computador que tem que tomar uma decisão. Assim, a linha 30 dá instruções ao computador no sentido de que *se* (IF) o utilizador armazenar MARGARIDAS em C$, *então* (THEN) deverá ser impressa no *écran* a mensagem VOCE ESCOLHEU UMA FLOR. E que acontece se o utilizador não armazenar MARGARIDAS em C$? Então o computador ignora a linha 30, passa sobre ela e segue para a linha 40. Esta indica-lhe que se (IF) o utilizador armazenar DENTES-DE-LEAO em C$ então (THEN) será impressa a mensagem VOCE ESCOLHEU UMA ERVA DANINHA.

O fluxograma seguinte auxiliá-lo-á a seguir o programa:



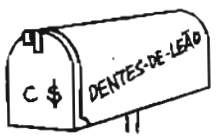
Note no fluxograma um novo símbolo, o losango, usado quando se tem de tomar uma decisão. Do losango saem duas setas, uma representando o caminho que o computador seguiria se a decisão fosse «sim» e outra mostrando o caminho que o computador seguiria se a resposta fosse «não».

Se a resposta fosse MARGARIDA, o *output* seria:

```
GOSTA MAIS DE MARGARIDAS OU DE DENTES-DE-LEAO?
? MARGARIDAS
VOCE ESCOLHEU UMA FLOR
```

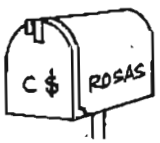
Se a resposta fosse DENTES-DE-LEÃO, o *output* seria:

```
? DENTES-DE-LEÃO
VOCE ESCOLHEU UMA ERVA DANINHA
```



Suponha agora que se decidiu a escrever ROSAS. O *output* seria:

```
GOSTA MAIS DE MARGARIDAS OU DE DENTES-DE-LEAO?
? ROSAS
```



O computador não imprimiu *nada* no *output*. Porquê? A linha 30 diz IF C$ = «MARGARIDAS» THEN PRINT «VOCE ESCOLHEU UMA FLOR». Mas a resposta é ROSAS e assim em C$ está ROSAS. O computador *ignora* a linha 30 e move-se para a seguinte, a linha 40. Esta diz IF C$ = «DENTES-DE-LEAO» THEN PRINT «VOCE ESCOLHEU UMA ERVA DANINHA». Mas em C$ também não está DENTES-DE-LEAO, está ROSAS e por isso o computador vai ignorar a linha

40 e move-se para a linha 50 onde está o fim (END) do programa. Em resumo o computador chegou à linha 50 sem imprimir qualquer outra mensagem.

Eis agora um programa que emprega 2 símbolos que provavelmente já usou em matemática:

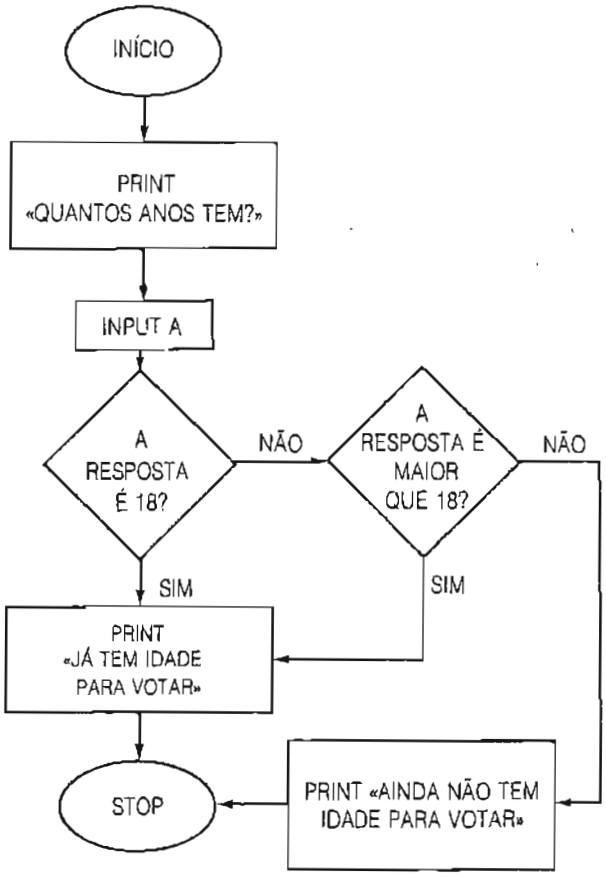
> é maior que
< é menor que

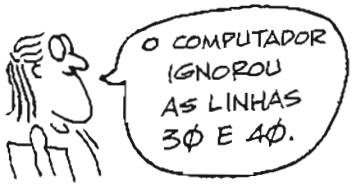
Com estes símbolos não use as palavras *é* e *que*.

Vamos agora ver outro programa e a primeira parte do seu *output*:

```
1Ø PRINT «QUANTOS ANOS TEM?»
2Ø INPUT A
3Ø IF A = 18 THEN PRINT «JÁ TEM IDADE PARA VOTAR.»
4Ø IF A > 18 THEN PRINT «JA TEM IDADE PARA VOTAR.»
5Ø IF A < 18 THEN PRINT «AINDA NÃO TEM IDADE PARA VOTAR.»
RUN
QUANTOS ANOS TEM?
?
```

O fluxograma ajudá-lo-á a seguir o programa.



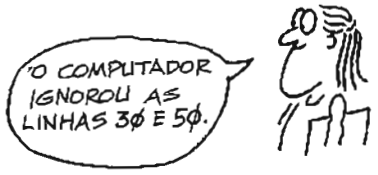


Suponha que o utilizador escreve 17 na resposta. Este valor é atribuído à variável A. Então o computador ao ler o programa salta sobre a linha 30 (A *não é* igual a 18), e sobre a linha 40 (A *não é* maior que 18) e «lê» a linha 50 porque A *é* menor que 18. E o *output* será.

```
QUANTOS ANOS TEM?
? 17
AINDA NAO TEM IDADE PARA VOTAR.
```

E se responder 18, será:

```
QUANTOS ANOS TEM?
? 18
JÁ TEM IDADE PARA VOTAR.
```



Há em BASIC uma forma abreviada que permite juntar as linhas 30 e 40 numa única linha, combinando os símbolos > e =.

```
IF A > = 18 THEN PRINT «JA TEM IDADE PARA VOTAR.»
```

Pode também alterar a ordem dos símbolos, assim: = >

Esta linha diz ao computador que se *A for maior ou igual a 18* então ele deve imprimir JÁ TEM IDADE PARA VOTAR.

As declarações IF-THEN podem-se usar para escrever um teste de computador. Mas cuidado, o computador é uma máquina «burra» que não sabe as respostas às perguntas. Assim se quiser escrever um teste deste tipo você deve, para além de escolher a pergunta, conhecer também a resposta. E deve-se escolher uma pergunta que tenha *uma única* resposta correcta.

Eis um exemplo:

```
10 PRINT «QUANTOS ESTADOS CONSTITUEM OS ESTADOS UNIDOS DA AMERICA?»
20 INPUT E
30 IF E = 50 THEN GOTO 60
40 IF E > 50 THEN PRINT «NAO, TENTE OUTRA VEZ.»
45 IF E < 50 THEN PRINT «NAO, TENTE OUTRA VEZ.»
50 GOTO 20
60 PRINT «ESTA CERTO!»
```

Verifiquemos o programa.

A linha 10 imprime a pergunta no *écran*.

A linha 20 permite ao utilizador escrever a resposta que vai ser armazenada em E.

A linha 30 diz ao computador que se E for 50 deve saltar para a linha 60, imprimir ESTA CERTO! e o teste foi resolvido. Mas se E *não for* igual a 50 então ele vai ignorar a linha 30 e segue para a linha 40.

A linha 40 indica ao computador que se E for maior que 50 deve imprimir NAO, TENTE OUTRA VEZ. Se E *não for* maior que 50 então o computador ignorará a linha 40 e seguirá para a linha seguinte.

A linha 45 diz ao computador que se E for menor que 50 deve imprimir NAO, TENTE OUTRA VEZ.

A linha 50 diz ao computador para voltar à linha 20 e assim o utilizador pode escrever outra resposta que ficará armazenada em E substituindo a anterior.

Como pode ver o computador apenas pode chegar à linha 60 se E for igual a 50.

Os sinais *menor que* e *maior que* estão combinados.

As linhas 40 e 45 podem-se combinar na seguinte forma abreviada:

```
40 IF E < > 50 THEN PRINT «NAO, TENTE OUTRA VEZ.»
```

Há duas formas de ler esta linha:

Se E *é menor ou maior* que 50, ...
Se E *não é igual* a 50, ...

Com esta alteração o programa registado na RAM parecer-se-á com:

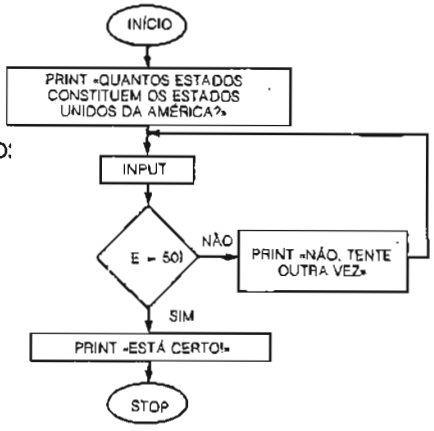
```
10 PRINT «QUANTOS ESTADOS CONSTITUEM OS ESTADOS UNIDOS DA AMÉRICA?»
20 INPUT E
30 IF E = 50 THEN GOTO 60
40 IF E < > 50 THEN PRINT «NAO, TENTE OUTRA VEZ.»
50 GOTO 20
60 PRINT «ESTA CERTO!»
```

Neste programa pode-se ainda usar uma outra forma abreviada na qual talvez já tenha pensado. Examine cuidadosamente as linhas 30 e 40. Se E é igual a 50 então o computador salta para a linha 60; e se E *não for* igual a 50, o computador ignora a linha 30 e segue para a linha 40. Como isto só acontece se E *não for* igual a 50, já não é necessário indicar no programa IF E < > 50 e pode alterar a linha 40 para:

```
40 PRINT «NAO, TENTE OUTRA VEZ.»
```

Com todas estas formas abreviadas que lhe introduzimos, o teste de computador tem agora o seguinte aspecto:

```
10 PRINT «QUANTOS ESTADOS CONSTITUEM
OS ESTADOS UNIDOS DA AMÉRICA?»
20 INPUT E
30 IF E = 50 THEN GOTO 60
40 PRINT «NAO, TENTE OUTRA VEZ.»
50 GOTO 20
60 PRINT «ESTA CERTO!»
```



Suponha que este programa está a correr e que as suas respostas à pergunta são 44, depois 52 e finalmente 50. O *output* completo seria:

```
? 44
NAO, TENTE OUTRA VEZ.
? 52
NAO, TENTE OUTRA VEZ.
? 50
ESTA CERTO!
```

Este programa pode tornar-se ainda mais pequeno, pois é possível colocar numa mesma linha duas declarações separadas por dois pontos (:). Veja cuidadosamente as linhas 30 e 40.

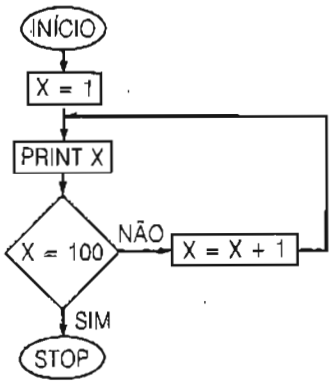
Por cada IF apenas pode usar um THEN. Não se pode dizer: IF E = 50 THEN PRINT «ESTA CERTO!» THEN END

```
10 PRINT «QUANTOS ESTADOS CONSTITUEM OS ESTADOS UNIDOS DA AMERICA?»
20 INPUT E
30 IF E = 50 THEN PRINT «ESTA CERTO!»: END
40 PRINT «NAO, TENTE OUTRA VEZ.»: GOTO 20
```

Se E = 50 na linha 30 o computador imprimirá ESTA CERTO! e o programa acaba. Mas se E não for igual a 50 o computador ignorará toda a linha incluindo a declaração END; chegado à linha 40 ele cumpre ambas as instruções: PRINT «NAO, TENTE OUTRA VEZ.» e GOTO 20.

Eis um outro tipo de programa que usa diversos tipos de instruções já conhecidas e do qual vai resultar no *écran* a impressão dos números de 1 a 100.

```
10 LET X = 1
20 PRINT X;
25 IF X = 100 THEN END
30 LET X = X + 1
40 GOTO 20
RUN
1 2 3 4 5 6 7 8 ...
```



Nalguns computadores os números aparecem juntos.

A linha 10 armazena 1 em X.

A linha 20 imprime o conteúdo de X.

Sem a linha 25 teríamos um ciclo infinito.

A linha 25 diz ao computador para finalizar se X for igual a 100; se X não for igual a 100 o computador segue para a linha seguinte.

A linha 30 junta 1 ao número armazenado em X.

A linha 40 faz o computador regressar à linha 20. O programa continua a correr até ser armazenado e impresso o número 100.

tente responder

1. Relativamente ao programa seguinte escreva o respectivo fluxograma e o *output:*

```
10 LET P = 2
20 PRINT P
30 IF P = 20 THEN END
40 LET P = P + 2
50 GOTO 20
```

2. Complete no programa seguinte a linha 30 de forma a que o *output* seja o indicado:

```
10 LET Z = 5
20 PRINT Z;
30 IF
40 LET Z = Z + 5
50 GOTO 20
RUN
5 10 15 20 25 30 35 40 45 50
```

Tome atenção que no seu computador os números poderão aparecer impressos juntos.

3. Escreva a linha 40 do programa seguinte de forma a que o *output* seja o indicado:

```
10 LET F = 100
20 PRINT F;
30 IF F = 0 THEN END
40
50 GOTO 20
RUN
100 90 80 70 60 50 40 30 20 10 00
```

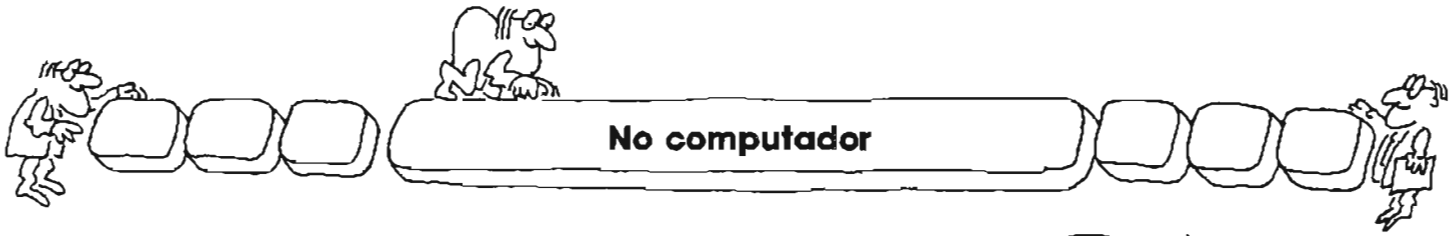
4. Desenhe o fluxograma do programa seguinte e indique depois qual o *output* deste jogo. Considere que a sua primeira resposta é 4, a segunda é 35 e a terceira é 20.

```
10 PRINT «QUANTAS MOEDAS DE 5 ESCUDOS HA EM CEM ESCUDOS?»
20 IF D = 20 THEN PRINT «CERTO!»: END
40 IF D < 20 THEN PRINT «SAO POUCAS — TENTE OUTRA VEZ.» GOTO 20
50 IF D > 20 THEN PRINT «SAO DEMAIS — TENTE OUTRA VEZ.»: GOTO 20
```

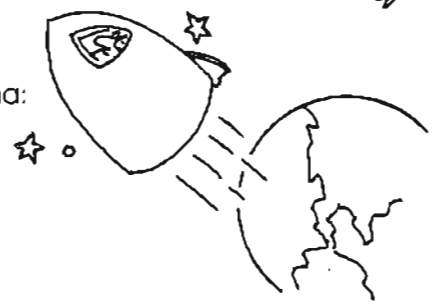
5. Escreva o programa para um jogo de computador que pergunte «Qual é a capital de Portugal?». Antes escreva um fluxograma que o auxilie a planear o programa.

Lembre-se que a resposta à pergunta é uma *cadeia* pelo que deverá usar uma variável tipo cadeia alfanumérica e aspas envolvendo-a.

6. Escreva o *output* do programa que escreveu no problema 5. Suponha que o utilizador respondeu Coimbra na primeira tentativa e Lisboa na segunda.



No computador



1. Escreva e faça correr o seguinte programa:

```
10 LET W = 10
20 PRINT W
30 IF W = 1 THEN PRINT «LARGADA!»: END
40 LET W = W — 1
50 GOTO 20
```

A linha 40 subtrai 1 ao número em W de cada vez que o computador salta da linha 50 para a linha 20.

O *output* deverá ser uma coluna de números desde 10 a 1 e a palavra LARGADA! Agora junte ao programa a seguinte linha e corra-o novamente:

```
25 IF W = 6 THEN PRINT "ESTOU A MEIO CAMINHO."
```

Introduza NEW para apagar o programa anterior.

2. Antes de escrever e de fazer correr o programa seguinte tente imaginar o que será o *output*.

```
10 LET X = 10
20 PRINT X
30 IF X = 100 THEN END
40 LET X = X + 10
50 GOTO 20
```

O *output* é uma coluna de números iniciada em 10, primeiro valor atribuído à variável X; o último número do *output* é 100, pois a linha 30 dá instruções ao computador para fazer END se X for 100; os números do *output* aumentam de 10 em 10, como pode ver pela linha 40 que indica ao computador para adicionar 10 ao número atribuído a X sempre que o computador «lê» aquela linha.

3. Altere as linhas 10, 30 e 40 para:

```
10 LET X = 100
30 IF X = 200 THEN END
40 LET X = X + 50
```

Para listar o programa introduza LIST.

Liste o programa para que possa ver as novas linhas na sua sequência normal. Corra o programa. Qual é agora o primeiro número do *output*? E o último?

4. Altere a linha 40 para

```
40 LET X = X + 3
```

Corra o programa. Eh! O que aconteceu? Porque é que o programa não pára? A linha 30 indica ao computador que deve terminar quando X for igual a 200. Mas se reparar X nunca terá o valor 200! Começa em 100, depois é 103, 106, 109 ... até que chega a 199, 202, 205... Portanto se X nunca é 200 o computador vai ignorar a linha 30 e o programa nunca irá acabar. Pare o programa.

Se se esqueceu de como parar um programa que está a correr, veja na pág. 112

5. Indique ao computador para parar o programa alterando a linha 30 para:

Recorde-se que o símbolo > significa «é maior que».

```
30 IF X > 200 THEN END
```

Corra o programa. Assim que o computador chega a um número maior que 200 e o imprime, o programa termina.

Introduza NEW.

6. Use uma declaração IF — THEN num ciclo para o computador imprimir numa coluna os números pares desde 0 a 40.

Introduza NEW antes de começar este programa.

7. Escreva este programa:

Não coloque quaisquer espaços extra dentro das aspas das linhas 30 e 40.

```
10 PRINT «DE QUAL GOSTA MAIS: CAMELO OU CROCODILO?»
20 INPUT G$
30 IF G$ = «CROCODILO» THEN GOTO 70
50 PRINT «ESCOLHEU UM MAMIFERO!»
60 GOTO 80
70 PRINT «ESCOLHEU UM REPTIL!»
80 END
```

Os utilizadores de computadores Atari deverão escrever:
5 DIM G$(15)

Corra o programa duas vezes. Numa das vezes responda CAMELO e na outra CROCODILO. Vê como a declaração IF — THEN nas linhas 30 e 40 determina a resposta ao computador?

8. Corra o programa novamente e responda TORDO. Qual é a resposta do computador? O quê!? ESCOLHEU UM MAMÍFERO. Um pássaro não é um mamífero. Por que razão o computador responde isto? Ele está apenas a seguir as suas instruções. Ignorou as linhas 30 e 40 porque G$ não armazenava nem CAMELO nem CROCODILO: a seguir passou para a linha seguinte, a linha 50, que indicava para imprimir ESCOLHEU UM MAMÍFERO; depois seguiu para a linha 60 que ordenava o salto para a linha 80 onde o programa finaliza. Pode corrigir este erro do programa juntando ao programa a seguinte linha 45:

```
45 GOTO 10
```

Agora corra o programa e responda novamente TORDO.

Introduza NEW.

9. Escreva o seguinte programa:

```
10 PRINT «ESCOLHA UM NUMERO ENTRE 1 E 99.»
20 INPUT N
30 IF N < 10 THEN PRINT «O SEU NUMERO TEM 1 DIGITO.»
40 IF N = > 10 THEN PRINT «O SEU NUMERO TEM 2 DIGITOS.»
```

Corra o programa diversas vezes usando números diferentes na sua resposta. Se a resposta for 500, o computador responde O SEU NUMERO TEM 2 DIGITOS! Mas 500 tem 3 digitos; olhe cuidadosamente para a linha 40 e para a mensagem que o computador deve imprimir se N for maior ou igual a 10. Como 500 é maior que 10, o computador imprimiu-a: O SEU NÚMERO TEM 2 DIGITOS.

10. Podemos melhorar o programa introduzindo uma linha 25:

```
25 IF N > 99 THEN PRINT «O SEU NUMERO E GRANDE DEMAIS!»: GOTO 10
```

Liste o programa e leia-o. Agora corra-o e tente introduzir a resposta 500.

Primeiro introduza NEW.

11. Escreva o seguinte teste de computador, parecido com o que já escreveu:

Os utilizadores de computadores Atari deverão escrever:
5 DIM C$(20).

```
10 PRINT «QUAL E A CAPITAL DE PORTUGAL?»
20 INPUT C$
30 IF C$ = «LISBOA» THEN PRINT «VOCE E ESPERTO!»: END
40 PRINT «NAO, TENTE NOVAMENTE.»
50 GOTO 20
```

Corra o programa e comece por escrever uma resposta errada. Depois escreva a resposta correcta.

12. Altere a linha 50 para:

```
50 GOTO 10
```

Corra o programa. Qual a diferença que esta alteração provoca?

Se vive em Lisboa use os distritos e suas capitais.

13. Altere as linhas 10 e 30 de forma a que o teste pergunte o nome da capital do seu distrito. Depois corra o programa.

14. Introduza NEW. Escreva um programa que produza o *output* seguinte. Para o ajudar a fazer o programa escreva antes um fluxograma.

No *écran* do seu computador, poderá não ter espaços entre os números.

```
1 2 3 4 5 6 7 8 9 10 DEZ
11 12 13 14 15 16 17 18 19 20 VINTE
21 22 23 24 25 26 27 28 29 30 TRINTA
31 32 33 34 35 36 37 38 39 40 QUARENTA
```

Sugestão — primeiro escreva um programa para imprimir os números de 1 a 40 no *écran*. Depois da instrução PRINT junte mais as seguintes quatro instruções IF-THEN:

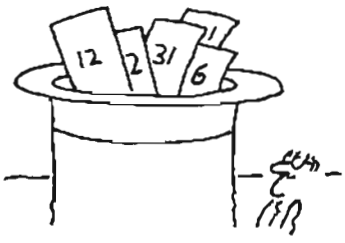
Corra o programa para se assegurar de que funciona.

```
IF X = 1Ø THEN PRINT «DEZ»
IF X = 2Ø THEN PRINT «VINTE»
IF X = 3Ø THEN PRINT «TRINTA»
IF X = 4Ø THEN PRINT «QUARENTA»
```

Introduza NEW.

15. Escreva um teste de computador. Antes de começar pense na pergunta e na resposta. Depois escreva um fluxograma do programa e só então o deve começar a escrever *no teclado*. Recorde que quando faz uma pergunta, se a resposta for uma cadeia, deve prever uma variável alfanumérica e se for um número deve prever uma variável numérica.

Capítulo 21 LANÇAR OS DADOS



Imagine um chapéu cheio de papéis numerados de 1 a 100. Retire um deles e leia o respectivo número: pode ser 6, 66, 32 ou qualquer outro número de 1 a 100. Volte a pôr o papel numerado no chapéu e faça outra tiragem: pode ser 45, 99, 2 ou até o mesmo número que tirou da última vez. Neste capítulo veremos como o computador pode fazer escolha de números ao acaso, como se estivesse a tirar à sorte papelinhos numerados de dentro de um chapéu.

Veja a primeira linha deste programa:

```
1Ø LET N = RND (25)
```

RANDOM significa "sem qualquer ordem ou modelo previsível".

Já sabe que a declaração LET significa que o computador vai atribuir qualquer coisa à variável N. Mas o quê? Qual o significado de RND (25)? RND deriva de *random*, palavra inglesa que significa "ao acaso", "casual", "aleatório". RND (25) significa que o computador vai escolher um número ao acaso entre 1 e 25. O número mais baixo que se pode escolher é sempre 1; o número entre parênteses é o número mais alto. Assim, no exemplo dado, pode ser escolhido qualquer número entre 1 e 25[21]. Para conhecer o número que o computador escolhe, junte ao programa a seguinte linha 20:

```
1Ø LET N = RND (25)
2Ø PRINT N
```

Apresentam-se a seguir alguns *outputs* deste programa. Como não sabemos previamente qual o número escolhido pelo computador e em seguida armazenado em N, vai ser pela linha 20 que o saberemos, pois ela indica ao computador para imprimir o que estiver atribuído à variável N.

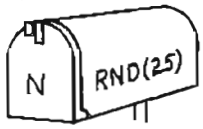
Quando você correr o programa pode obter um *output* diferente.

```
1Ø LET N = RND (25)
2Ø PRINT N
RUN
8
RUN
17
RUN
2
```

Este é o número mais alto que o computador pode escolher. O número mais baixo que pode ser escolhido é o 1.

Vamos agora juntar um ponto e vírgula à linha 20 e acrescentar uma linha 30 de forma a que o programa na RAM seja como o seguinte:

```
10 LET N = RND (25)
20 PRINT N;
30 GOTO 10
```



Recorde-se que um **ciclo infinito** continua ininterruptamente.

O programa tem um ciclo infinito: dá instruções ao computador para escolher um número entre 1 e 25 e imprimi-lo; depois regressa à linha 10 e de novo é escolhido e impresso um número entre 1 e 25 — e assim sucessivamente. Um exemplo de um *output* deste programa será:

No seu computador pode não haver espaços entre os números.

```
3 6 15 24 1 6 13 25 16 ...
```

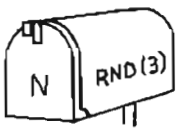
Se correr o programa várias vezes, de cada uma delas o *output* parecerá diferente[22].

Suponha agora que altera a linha 10 do programa da seguinte forma:

```
10 LET N = RND (3)
20 PRINT N;
30 GOTO 10
```

Agora o computador apenas pode escolher números entre o 1 e o 3. E o *output* poderia ser:

```
1 2 1 3 3 1 2 2 1 1 3 3 3 1 2 ...
```



Verifique com o seu professor se precisa ou não de ler esta secção.

Em muitos computadores a geração de números aleatórios não é tão fácil como a que vimos até agora. É o caso dos computadores Apple, Atari, Commodore, IBM e ZX Spectrum em que a escolha dos números aleatórios é feita de uma maneira diferente.

Numa instrução como:

```
LET X = RND (8)
```

O número entre parênteses é um número "morto", sem significado especial[23]. Também se poderia utilizar:

```
LET X = RND (92)
```

Ambas as instruções dão origem a um número decimal entre 0 e 1 que vai ser armazenado em X. Veja o programa e o *output*:

```
10 X = RND (10)
20 PRINT X
30 GOTO 10
RUN
 .2368392
 .1479472
 .3859285
```

Os números obtidos no seu computador podem ser mais pequenos ou mais compridos do que estes.

Geralmente os números décimais como estes não têm grande utilidade. Necessitamos de os transformar em números inteiros. Eis uma maneira de o fazer:

Quando se multiplica por 10, a vírgula ou ponto decimal move-se de um espaço para a direita.

INT deriva de INTEGER (parte inteira). É uma instrução da linguagem BASIC.

10 N = RND (1)	Armazena um número decimal em N, por exemplo .234967.
20 N = N * 10	Multiplica N por 10; agora N = 2,34967.
30 N = INT (N)	N Torna-se na parte inteira do novo valor de N. É como se se desprezassem os números à direita da vírgula. Agora N = 2.
40 PRINT N	Imprime 2
50 GOTO 10	O programa recomeça

```
RUN
2
9
3
6
...
```

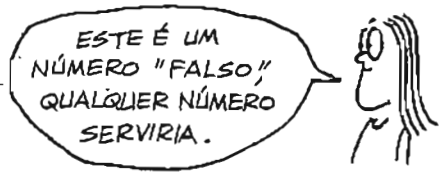
Este programa produzirá números dígitos de 0 a 9. Se se quisessem números de 1 a 10, neste caso teríamos de adicionar 1 a N:

```
35 N = N + 1
```

Mas há uma maneira abreviada de combinar as linhas 10, 20, 30 e 35:

Nos computadores Apple e IBM não utilize o 0 como número "morto".

```
N = INT(RND (1)*10) + 1
```



Quando desejar que o computador gere números aleatórios use esta instrução. A seguir ao sinal de multiplicação indica-se o número *maior* que se deseja que saia; o número mais baixo será sempre o número 1. Assim, por exemplo para obter números aleatórios entre 1 e 25 escrever-se-ia:

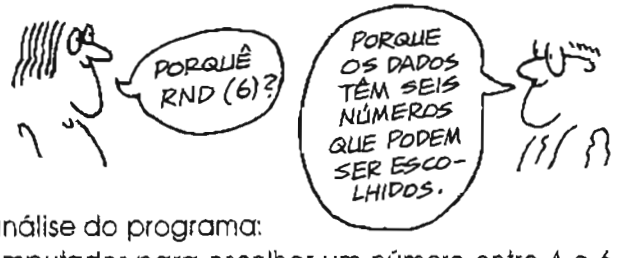
```
N = INT(RND(1)*25) + 1
```

Esta secção volta a ser para todos.

Com a instrução RND podemos fazer o computador "jogar dados":

Para o computador ZX Spectrum as linhas 10 e 20 são:
10 LET D = INT(RND*6) + 1
20 LET E = INT(RND*6) + 1

```
10 D = INT(RND(1)*6) + 1
20 E = INT(RND(1)*6) + 1

30 PRINT "O DADO # 1 É"; D
40 PRINT "O DADO # 2 É"; E
```



Vamos agora fazer a análise do programa:
A linha 10 indica ao computador para escolher um número entre 1 e 6 e armazená-lo em D.
A linha 20 indica ao computador para escolher um número entre 1 e 6 e armazená-lo em E.
A linha 30 indica ao computador para imprimir O DADO # 1 É e o número armazenado em D.
A linha 40 indica ao computador para imprimir O DADO # 2 É e o número armazenado em E.

Se correr o programa duas vezes poderá obter o seguinte *output*:

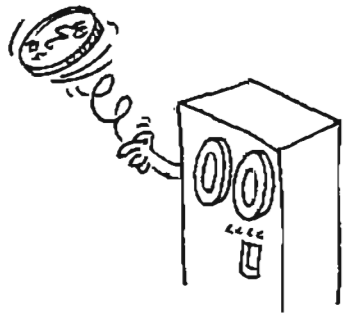
```
RUN
O DADO # 1 É 5
O DADO # 2 É 2

RUN
O DADO # 1 É 4
O DADO # 2 É 4
```

Recorde-se que a simulação do computador imita uma situação da vida real.

Também podemos usar a instrução RND para fazer o computador simular o lançamento de uma moeda ao ar. Vejamos esse programa:

Para o computador ZX Spectrum, a linha 10 é:
LET C = INT(RND*2) + 1

```
10 C = INT(RND(1)*2) + 1
20 IF C = 1 THEN PRINT "CARAS"
30 IF C = 2 THEN PRINT "COROAS"
RUN
COROAS
RUN
CARAS
```



Neste programa o computador apenas pode escolher os números 1 e 2. Com as linhas 20 e 30 o computador faz corresponder a cada um daqueles números as palavras CARAS (se sair 1) ou COROAS (se sair 2) e não os números tirados ao acaso.

Muitos programas de jogos de computador usam números aleatórios. Eis como escrever um jogo para adivinhar um número. O computador escolhe um número entre 1 e 10 e o utilizador tenta adivinhá-lo. As três primeiras linhas serão:

Para o computador ZX Spectrum, a linha 10 é:
LET C = INT(RND*10) + 1

```
10 N = INT (RND(1)*10) + 1
20 PRINT "ADIVINHE UM NÚMERO ENTRE 1 e 10."
30 INPUT G
```

Em N está armazenado um número escolhido pelo computador entre 1 e 10; no *écran* está impressa uma mensagem que convida o utilizador a adivinhar esse número. O utilizador fá-lo e o seu palpite é armazenado em G, ficando-se agora com *duas* variáveis — N, o número do computador e G, o palpite do utilizador.

Iremos ver agora como se compara o palpite do utilizador com o número escolhido pelo computador:

```
40 IF G = N THEN GOTO 70
50
60
70 PRINT "VOCE ACERTOU!"
```

Já iremos preencher estas linhas.

Se o utilizador adivinhou o número é impressa uma mensagem. Mas e se o palpite *não for* o número escolhido pelo computador? Nesse caso ele vai saltar a linha 40 e segue para a linha 50:

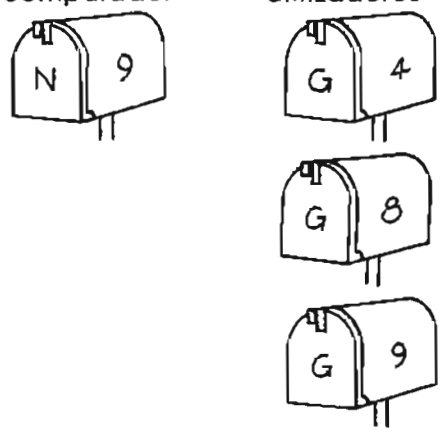
```
50 PRINT "NÃO É ESSE. TENTE NOVAMENTE."
60 GOTO 30
```

Esta mensagem é impressa e o utilizador deve adivinhar outro número. Vejamos agora o programa completo:

```
10 N = INT(RND(1)*10) + 1
20 PRINT "ADIVINHE UM NÚMERO ENTRE 1 E 10."
30 INPUT G
40 IF G = N THEN GOTO 70
50 PRINT "NAO E ESSE. TENTE NOVAMENTE."
60 GOTO 30
70 PRINT "VOCE ACERTOU!"
80 END
```

Eis um *output* possível deste programa supondo que o número armazenado é 9.

```
ADIVINHE UM NÚMERO ENTRE 1 E 10.
?4
NAO E ESSE. TENTE NOVAMENTE.
?8
NAO E ESSE. TENTE NOVAMENTE.
?9
VOCE ACERTOU!
```



Recorde-se que os utilizadores do ZX Spectrum precisarão de escrever:
LET N = INT(RND*10)+1

1. Observe o seguinte programa:

```
10 N = INT(RND(1)*20) + 1
20 PRINT N
30 GOTO 10
```

← Para o computador ZX Spectrum a linha 10 será:
LET N = INT(RND*20) + 1

   a. Qual o maior número que o computador pode escolher?
   b. Qual o menor número que o computador pode escolher?

2. Escreva um exemplo de *outputs* do programa do exercício 1.

3. Escreva dois *outputs* possíveis do programa seguinte:

```
10 R= INT(RND(1)*2) + 1
20 IF R = 1 THEN PRINT "VERMELHO"
30 IF R = 2 THEN PRINT "AZUL"
40 END
```

← Para o computador ZX Spectrum a linha 10 será:
LET R = INT(RND*2) + 1

No computador

1. Escreva e corra o seguinte programa:

```
10 B = INT(RND(1)*12) + 1
20 PRINT B
```

Corra o programa diversas vezes: obtem sempre um número entre 1 e 12.

Os utilizadores do ZX Spectrum escreverão
10 LET B = INT(RND*12)+1

2. Junte a seguinte linha 30:

```
30 GOTO 10
```

Corra o programa: obtem, um após outro, todos os números entre 1 e 12? Páre o programa.

3. Altere a linha 30 para:

```
30 GOTO 20
```

Adivinhe o que vai acontecer antes de correr o programa. Note que depois de escolher um número (linha 10) e de o imprimir (linha 20), o computador regressa *à* linha 20, imprimindo novamente esse número; nunca regressa à linha 10 para escolher um novo número. Corra o programa algumas vezes: não sabe previamente o número que o computador vai escolher, mas qualquer que ele seja, o computador continua a imprimi-lo.

Para o computador ZX Spectrum escreva:
10 LET F = INT(RND*25)+1

4. Comece por introduzir NEW. Depois escreva o seguinte programa:

```
10 F = INT(RND(1)*25) + 1
20 PRINT F,
30 IF F = 10 THEN END
40 GOTO 10
```

O *output* será impresso em zonas.

Corra o programa várias vezes: nalguns casos obterá muitos números e noutros poucos números. Porquê? A chave do enigma está na linha 30: *se* (IF) o computador armazenar e imprimir o número 10 *então* (THEN) o programa acaba. Para isto acontecer tanto pode demorar muito como pouco tempo.

5. Altere a linha 30 para:

```
30 IF F = 50 THEN END
```

Corra o programa. Vai parar? Não porque F nunca pode ter o valor 50; com este programa o maior número que o computador pode escolher é 25. Páre o programa.

Para o computador ZX Spectrum, escreva:
10 LET D = INT(RND*6) +
10 LET E = INT(RND*6) +

6. Introduza NEW e depois o programa seguinte. Corra-o várias vezes.

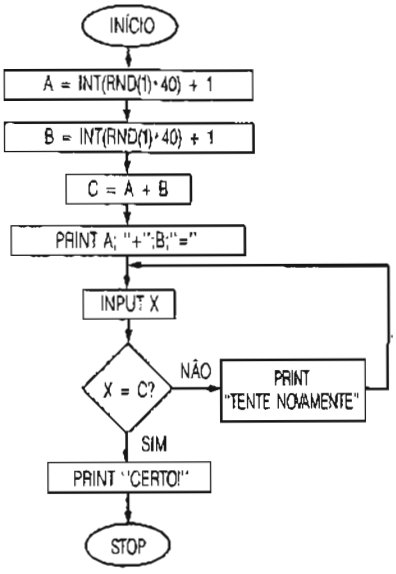
```
1Ø D = INT(RND(1)*6) + 1
2Ø E = INT(RND(1)*6) + 1
3Ø PRINT "O RESULTADO DA SUA JOGADA E:"
4Ø PRINT "DADO 1"; D
5Ø PRINT "DADO 2"; E
```

7. Junte ao seu programa algumas linhas de forma a fazer rolar quatro dados. Para tal faça com que o computador atribua números ao acaso a mais duas variáveis, F e G; precisará de utilizar números de linha entre o 20 e o 30, assim como outras duas linhas, depois da linha 50, para imprimir os resultados completos de cada jogada.

8. Introduza NEW e depois o seguinte teste de matemática. Corra o programa várias vezes.

Para o computador ZX Spectrum, escreva:
10 LET A = INT(RND*40) + 1
10 LET B = INT(RND*40) + 1

```
1Ø A = INT(RND(1)*40) + 1
2Ø B = INT(RND(1)*40) + 1
3Ø C = A + B
4Ø PRINT A; "+";B;"=";
5Ø INPUT X
6Ø IF X = C THEN PRINT "CERTO!": END
7Ø PRINT "TENTE NOVAMENTE.": GOTO 4Ø
```



9. Junte a seguinte linha 80:

```
8Ø GOTO 1Ø
```

Agora o computador em vez de terminar o programa continuará a imprimir novos problemas. Corra o programa. Se achar os problemas fáceis demais, altere os números aleatórios das linhas 10 e 20 de forma a jogar com números entre 1 a 100. Então corra o programa novamente.

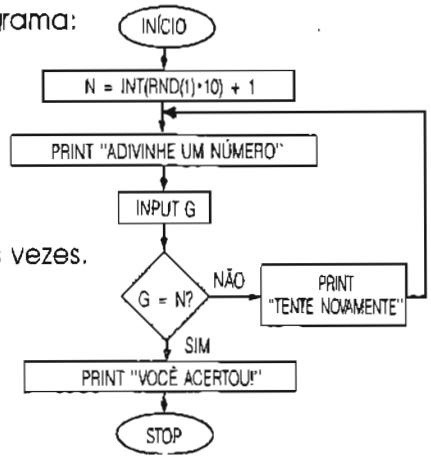
Para alterar as linhas 20 e 30 deverá parar o computador e listar o programa.

10. Tente alterar este programa para um teste de multiplicação. Terá de alterar o sinal de adição das linhas 30 e 40 para o sinal de multiplicação (*). Se quando correr o programa os problemas lhe parecerem muito difíceis altere os números aleatórios das linhas 10 e 20.

11. É agora altura de tentar um jogo de adivinhar números. Depois de introduzir NEW, escreva o seguinte programa:

Para o computador ZX Spectrum escreva:
10 LET B = INT(RND*10) + 1

```
1Ø N = INT(RND(1)*1Ø) 1
2Ø PRINT "ADIVINHE UM NUMERO."
3Ø INPUT G
4Ø IF G = N THEN PRINT "VOCE ACERTOU!"; END
5Ø PRINT "TENTE NOVAMENTE.": GOTO 3Ø
```



Jogue o jogo (corra o programa) várias vezes.

Os utilizadores do computador ZX Spectrum escreverão:
10 LET B = INT(RND*100)+1

12. Altere a linha 10 para:

```
10 N = INT(RND(1)*100) + 1
```

Como poderia demorar muito tempo até adivinhar o número certo, junte algumas sugestões auxiliares. Assim mude a linha 50 para:

```
50 IF G > N THEN PRINT "E MUITO ALTO. TENTE NOVAMENTE.": GOTO 30
```

E acrescente a seguinte linha:

```
55 IF G < N THEN PRINT "E MUITO BAIXO. TENTE NOVAMENTE.": GOTO 30
```

Agora corra o programa várias vezes.

Introduza NEW.

13. Escreva um programa para imprimir nas zonas do *écran* números aleatórios entre 1 e 99.

14. Altere o programa de forma a que o computador imprima números aleatórios entre 1 e 9.

15. Junte ao seu programa uma linha, de forma a que o computador pâre depois de imprimir um 5.

Introduza NEW.

16. Programe um teste de adição. Faça o computador escolher dois números entre 1 e 30 e adicioná-los. Se o utilizador der uma resposta errada assegure-se que o computador indica qual a resposta certa. Desenhe primeiro um fluxograma que o auxilie a escrever o programa.

Introduza NEW.

17. Programe um jogo para adivinhar números em que o computador escolhe um número de 1 a 500 e de cada vez que o utilizador der um palpite o computador emite uma mensagem do tipo É MUITO ALTO ou É MUITO BAIXO. Antes de começar, desenhe um fluxograma.

Capítulo 22 ANDANDO ÀS VOLTAS

Já atrás se utilizou a instrução GOTO com a finalidade de voltar atrás, efectuando um ciclo no programa. O programa seguinte é um exemplo de um ciclo infinito, que se repete ininterruptamente:

Este programa é um exemplo de um ciclo infinito.

```
10 PRINT "ESTOU-ME A REPETIR"
20 GOTO 10
```

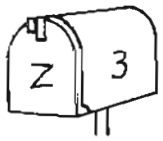
Neste capítulo iremos aprender a forma de dizer ao computador exactamente quantas vezes desejamos que ele repita o ciclo.

Vejamos o programa seguinte:

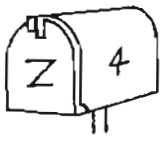
FOR e NEXT são instruções da linguagem BASIC.

```
10 FOR Z = 3 TO 9
20    PRINT Z
30 NEXT Z
```

Existem aqui duas novas instruções — FOR e NEXT — que aparecem sempre como um par nos programas: se existe uma instrução FOR, deve haver uma instrução NEXT. Estas instruções indicam ao computador quantas repetições ele tem de fazer. Analisemos o programa e verifiquemo-lo:



A linha 10 diz ao computador para armazenar um número em Z. Mas que número? O computador apenas atribui a cada variável um número de cada vez; começa armazenando em Z o primeiro dos números indicados, o 3 e depois segue para a linha seguinte.



A linha 20 indica ao computador para imprimir o que estiver em Z: e assim ele imprime 3.

NEXT Z na linha 30 significa que o computador deve regressar à linha com a instrução FOR e coloca em Z o número seguinte (em inglês *next*). Assim o computador volta à linha 10 e coloca em Z o número seguinte, o 4. Depois move-se para a linha 20 e imprime o que estiver em Z ou seja 4.

A linha 30 faz o computador regressar à linha com a instrução FOR, a linha 10, e colocar em Z o número seguinte *(next)*, 5, e assim sucessivamente.

O computador continua a repetir-se da linha 30 para a linha 10, até terem sido armazenados em Z todos os números de 3 a 9, um de cada vez.

Depois de ter sido armazenado e impresso o número 9, não há mais números para armazenar em Z e o programa pára.

Eis o *output* deste programa:

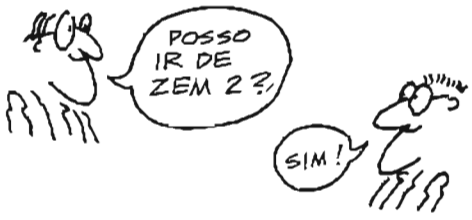
```
RUN
3
4
5
6
7
8
9
```

O programa seguinte faz imprimir no *écran* todos os números entre 0 e 14.

```
10 FOR H = 0 TO 14
20    PRINT H;
30 NEXT H
```

Vamos agora supor que se quer que o computador falhe alguns números em vez de continuar com o número seguinte do ciclo. Veja como se alterou a linha 10 e também como é o *output*.

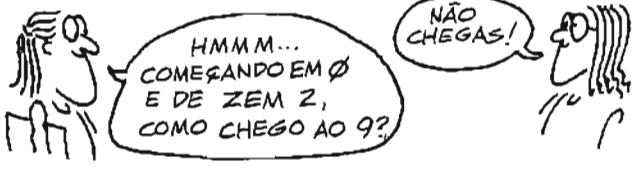
```
10 FOR H = 0 TO 14 STEP 2
20    PRINT H;
30 NEXT H
RUN
0  2  4  6  8  10  12  14
```



Como pode verificar pelo *output*, STEP 2 indica ao computador para, de cada vez que regressa à linha 10, somar 2 unidades ao valor de H, atribuindo a esta variável um valor superior em duas unidades ao anterior. Se a declaração FOR não for completada no fim da linha com STEP, o computador automaticamente soma apenas 1 unidade.

Qual pensa que poderá ser o *output* do programa seguinte?

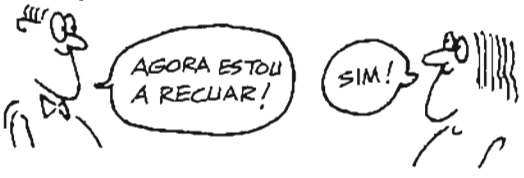
```
10 FOR S = 0 TO 9 STEP 2
20    PRINT S;
30 NEXT S
40 END
```



O computador imprimirá 0, 2, 4, 6 e 8. Depois pára. Numa declaração FOR o segundo número nunca é ultrapassado.

Eis um outro programa com a declaração FOR-NEXT e STEP.

```
10 FOR S = 90 TO 80 STEP -1
20    PRINT V;
30 NEXT V
40 RUN
90  89  88  87  86  85  84  83  82  81  80
```



Os números negativos são números menores que zero

A diferença nesta declaração STEP é que é seguida de um número negativo. Veja com cuidado a linha 10. Na declaração FOR indicam-se os números de 90 a 80. Primeiro o computador imprime 90; depois STEP −1 provoca que o número seguinte seja 89, depois 88 e assim sucessivamente. Num ciclo FOR — NEXT, para ir de um número maior a um número menor, deve sempre usar STEP −1 (ou −2, ou −3, etc. conforme se quiser obter números de 2 em 2, de 3 em 3 e assim sucessivamente).

Eis agora um outro programa com um ciclo FOR—NEXT:

```
10 FOR J = 1 TO 3
20    PRINT "SALTO"
30 NEXT J
```

Como será o *output*? Sabe que a variável J vai armazenando os números de 1 a 3. Mas o programa não tem qualquer instrução para imprimir J. Analizemos o programa:

A linha 10 começa o ciclo armazenando 1 em J.
A linha 20 dá instruções ao computador para imprimir SALTO.
A linha 30 manda o computador regressar à linha 10.
Agora a linha 10 atribui 2 a J.
A linha 20 imprime novamente SALTO.
A linha 30 manda o computador regressar à linha 10.
A linha 10 atribui 3 a J.
A linha 20 imprime novamente SALTO.
A linha 30 manda o computador regressar à linha 10. Como não há mais números para armazenar, o programa termina.

O programa e o seu *output* serão:

```
10 FOR J = 1 TO 3
20    PRINT "SALTO"
30 NEXT J
RUN
SALTO
SALTO
SALTO
```

O computador utilizou um ciclo FOR-NEXT para contar de 1 a 3 imprimindo SALTO tantas vezes quantos os números contados.

Veja agora o programa seguinte e o respectivo *output*:

A linha 20 imprime um número e uma cadeia.

```
10 FOR P = 1 TO 5
20    PRINT P;"CONTINUA "
30 NEXT P
RUN
1 CONTINUA
2 CONTINUA
3 CONTINUA
4 CONTINUA
5 CONTINUA
```

Se olhar atentamente para a linha 20, verifica que, de cada vez que o computador faz o ciclo, imprime o número nesse momento armazenado em P e, na mesma linha, a palavra CONTINUA. Da primeira vez que faz o ciclo, P é 1 e o computador imprime 1 CONTINUA. Da segunda vez que faz o ciclo P é 2, o computador imprime 2 CONTINUA e assim sucessivamente.

1. Escreva o *output* do programa seguinte:

```
10 FOR H = 13 TO 19
20    PRINT H
30 NEXT H
```

2. Escreva o *output* do programa seguinte:

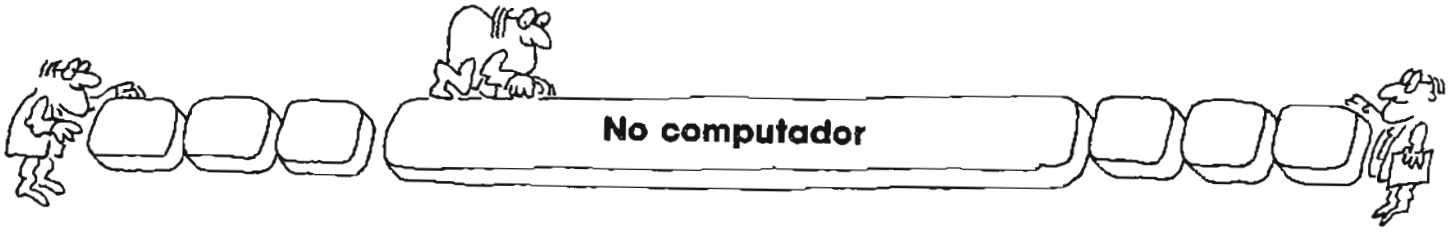
```
10 FOR S = 3 TO 21 STEP 3
20    PRINT S
30 NEXT S
```

3. Detecte o erro do programa seguinte e emende-o; depois escreva o *output*:

```
10 FOR C = 40 TO 35 STEP -1
20    PRINT C
30 END
```

4. Complete a linha 10 de forma ao computador imprimir o *output* representado:

```
10
20    PRINT A;
30 NEXT A
RUN
7  9  11  13  15  17
```



A utilização de ciclos FOR-NEXT

Quando listar o seu programa o computador Commodore "roubará" os espaços que você usou para recolher o início de algumas linhas.

1. Escreva e corra o seguinte programa. O *output* será os números de 2 a 12, apresentados em coluna.

```
10 FOR B = 2 TO 12
20    PRINT B
30 NEXT B
```

2. Junte STEP 2 na linha 10. A linha 10 ficará:

```
10 FOR B = 2 TO 12 STEP 2
```

Corra o programa. O *output* ainda começa em 2 e termina em 12, mas salta dois números de cada vez.

Introduza NEW.

3. Escreva e corra o seguinte programa:

Não esqueça o sinal de menos em STEP −1

```
10 FOR Y = 100 TO 0 STEP −1
20    PRINT Y;
30 NEXT Y
```

Obteve no *écran* os números de 100 até 1?

A linha 20 diz ao computador para "imprimir" um espaço depois de cada número. Nalguns tipos de computadores poderá ter necessidade de escrever
20 PRINT H; " " ;
de forma a ficar um espaço entre cada número.

4. Altere a linha 10 para:

```
10 FOR Y = 100 TO 0 STEP −2
```

Corra o programa. Depois faça alterações sucessivas na linha 10 como a seguir se indica, correndo o programa após cada alteração.

```
10 FOR Y = 100 TO 0 STEP−3
10 FOR Y = 100 TO 0 STEP−10
10 FOR Y = 100 TO 0 STEP−50
10 FOR Y = 100 TO 0 STEP−100
```

Introduza NEW.

Use o seu nome na linha 20.

5. Escreva e corra o seguinte programa:

```
10 FOR U = 1 TO 10
20    PRINT "PAULA"
30 NEXT U
```

Obteve 10 vezes o nome Paula (ou o seu nome)?

6. Que acontece se juntar STEP 2 à linha 10?

```
10 FOR U = 1 TO 10 STEP 2
```

Tente imaginar o *output* verificando o programa antes de o correr.

Introduza NEW

7. Escreva e corra o seguinte programa:

```
10 FOR L = 1 TO 5
20     PRINT "CICLOS"
30 NEXT L
```

Deverá obter cinco CICLOS.

Os utilizadores do computador Atari escreverão:
2 DIM P$(20)

8. Altere o programa anterior juntando-lhe as seguintes linhas 3 e 5:

```
3 PRINT "ESCREVA UMA PALAVRA QUALQUER."
5 INPUT W$
```

A palavra do utilizador é armazenada em W$.

Agora altere a linha 20 para:

```
20 PRINT W$
```

Liste o programa. Quando o correr e escrever uma palavra, ela fica registada em W$; então o computador faz o ciclo FOR-NEXT e imprime W$ (na linha 20), substituindo W$ pela sua palavra. Corra várias vezes o programa escrevendo de cada vez uma palavra diferente.

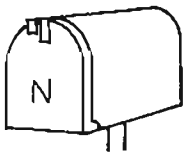
9. Altere a linha 20 para:

```
20 PRINT L; W$
```

Consegue prever o *output* antes de correr o programa? Lembre-se do que está armazenado em L. Da primeira vez que o ciclo é corrido, na posição de endereço L armazena-se 1, da segunda vez, 2, etc.

10. Juntemos mais duas linhas ao programa:

```
7 PRINT "ESCREVA UM NÚMERO."
9 INPUT N
```



O número escrito pelo utilizador é armazenado aqui.

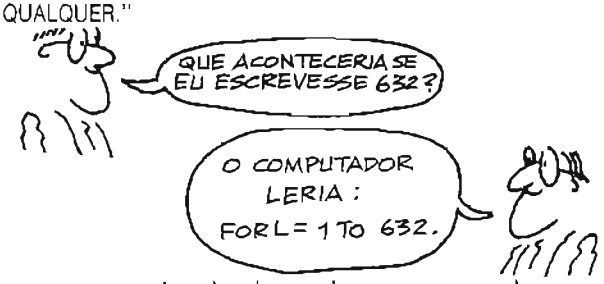
Agora altere a linha 10:

```
10 FOR L = 1 TO N
```

Liste o seu programa. Deverá ficar assim:

```
3 PRINT "ESCREVA UMA PALAVRA QUALQUER."
5 INPUT W$
7 PRINT "ESCREVA UM NUMERO."
9 INPUT N
10 FOR L = 1 TO N
20     PRINT L, W$
30 NEXT L
40 END
```

Recorde que os utilizadores dos computadores Atari necessitam de uma declaração DIM na linha 2.



Corra o programa várias vezes respondendo de cada vez com palavras e números diferentes.

Introduza NEW.

11. Usando um ciclo FOR-NEXT escreva um programa para imprimir em coluna os números de 3 a 33.

12. Altere uma linha no programa para que o computador conte de três em três. O seu *output* será 3 6 9 12 ... 33 mas em coluna.

Introduza NEW.

13. Escreva um programa para imprimir os números desde 99 até 0. Decida a forma como quer que o *output* seja apresentado no *écran*.

14. Altere uma linha no programa de forma a imprimir só os números ímpares.

Introduza NEW.

15. Escreva um programa que imprima no *écran* 10 vezes SOU UM PROGRAMADOR BESTIAL.

16. Altere uma linha do programa de forma a imprimir a mesma frase 100 vezes.

Pausas

Introduza NEW antes de começar.

1. Escreva e corra o seguinte programa:


```
10 PRINT "NUNCA PARO."
20 GOTO 10
```

Se se esqueceu de como parar o computador, veja a pág. 112.

Páre o computador. Depois, acrescente as linhas seguintes e liste o programa:

```
13 FOR P = 1 TO 500
15 NEXT P
```

Antes de correr o programa tente imaginar o que irá acontecer. As linhas 13 e 15 constituem um ciclo FOR-NEXT. O computador armazena 1 na posição de endereço P, depois um 2, depois um 3, e assim sucessivamente e até 500. E que faz o computador depois de armazenar um número em P? Nada! Segue para o NEXT P e armazena o próximo número em P, não imprimindo nada entre o armazenamento de um número em P e o registo do próximo. Mas demora algum tempo ao computador armazenar estes números em P, um de cada vez.
Corra o programa. Verá o tempo que demora ao computador armazenar 500 números, um de cada vez, no endereço (ou variável) P, antes de seguir para a linha 20. Esta manda-o regressar à linha 10. Há assim uma pausa entre a impressão de cada linha do *output* pelo computador. Páre o computador.

2. Altere a linha 13 para:

```
13 FOR P = 1 TO 1000
```

Corra o programa. Poderá ver que o computador demora cerca do dobro do período de tempo anterior a armazenar no endereço P mil números, um de cada vez. Pare o computador.

3. Altere a linha 13 para:

```
13 FOR P = 1 TO 50
```

Corra o programa. Agora o período de tempo para armazenar em P 50 números, um de cada vez, é muito menor. Pare o computador.

4. Acrescente as seguintes linhas ao seu programa:

```
5 PRINT "ESCOLHA UM NÚMERO QUALQUER. SE FOR ALTO O PROGRAMA CORRERÁ LENTAMENTE. SE FOR BAIXO CORRERÁ RAPIDAMENTE."
7 INPUT N
```

Lembre-se que N é um número qualquer escolhido pelo utilizador.

Agora altere a linha 13 para:

```
13 FOR P = 1 TO N
```

Liste o programa. Corra-o várias vezes. Já percebeu por que razão um número elevado torna o *output* mais lento?

Introduza NEW.

5. Escreva um programa para imprimir várias vezes FICAMOS AQUI POR AGORA! com uma pausa após cada linha do *output*.

Capítulo 23 DESCOBRINDO OS FACTOS DADOS

Já vimos que há várias maneiras de armazenar informações em endereços: uma é atribuir os dados usando instruções LET (LET A = 2 ou LET C$ = "COMPUTADOR"); outra é pedir a introdução da informação com a instrução INPUT (INPUT X ou INPUT Z$). Neste capítulo iremos aprender uma terceira maneira de o fazer.

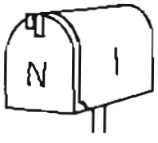
Vejamos o programa seguinte:

READ e DATA são instruções da linguagem BASIC.

```
10 READ N
20 IF N = 0 THEN END
30 PRINT N
40 GOTO 10
50 DATA 1, 3, 5, 0
```

Neste programa há duas novas instruções — READ (ler) e DATA (dados) — que aparecem sempre juntas num programa: se tem uma instrução READ deve vir a ter uma instrução DATA e vice-versa.

Quando temos uma série de informações ou dados para ser usada num programa, podemos pôr todos esses dados numa instrução DATA. A instrução DATA da linha 50 por exemplo contém quatro números, cada um deles separado do seguinte por uma vírgula. Analisemos o programa:

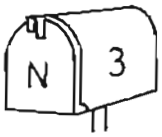


A linha 10 (READ N) indica ao computador para saltar para a declaração DATA onde quer que ela esteja (no princípio, meio ou fim do programa); o computador encontrá-lo-á neste caso na linha 50 e lê (em Inglês READ) o primeiro dado, o número 1; atribui este valor à variável N e continua para a linha seguinte à instrução READ ou seja a linha 20.

A linha 20 diz ao computador para terminar se N = 0; mas N *não é* 0 e portanto o computador segue para a linha seguinte.

A linha 30 diz ao computador para imprimir o número armazenado em N e que é 1.

A linha 40 indica ao computador o regresso à linha 10 que é a instrução READ.



A linha 10 diz ao computador para saltar para a instrução DATA; os dados "usados" nunca são lidos, por isso o computador passa sobre o 1 e lê o 3 que é armazenado em N.

Na linha 20, verificando que o número armazenado em N não é 0, o computador segue para a linha 30 e imprime 3.

O computador regressa à linha 10 e lê o dado seguinte, o 5, e armazena-o em N. N não é 0 e o computador imprime o 5.

O computador regressa à linha 10 e lê o 0. Quando chega à linha 20, o 0 armazenado em N faz o programa terminar.

Eis portanto o programa e o seu *output*:

```
1Ø READ N
2Ø IF N = Ø THEN END
3Ø PRINT N
4Ø GOTO 1Ø
5Ø DATA 1, 3, 5, Ø
RUN
1
3
5
```

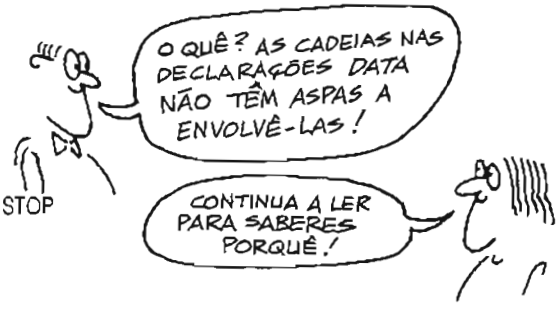
Na linha 50 os vários dados são separados por vírgulas. Estas vírgulas não têm nada a ver com zonas do *écran*.

O programa pára após ler o 0 e antes de o imprimir, pois zero neste caso é um dado "falso".

O computador também pode ler cadeias alfanuméricas. Neste programa cada dado é armazenado em G$. STOP é um dado "falso".

Os utilizadores de computadores Atari terão de usar uma instrução DIM para G$.

```
1Ø READ G$
2Ø IF G$ = "STOP" THEN END
3Ø PRINT G$
4Ø GOTO1Ø
5Ø DATA BICICLETAS, CHAVES, PATINS, STOP
RUN
BICICLETAS
CHAVES
PATINS
```

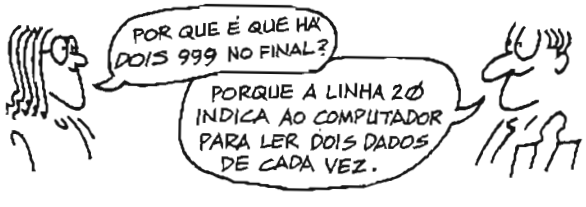


Na linha 50 as cadeias não têm aspas. Na declaração DATA não temos de pôr aspas à volta da cadeia[24] a menos que a própria cadeia tenha uma vírgula: por exemplo a cadeia ALIADOS, PORTO deve ser rodeada de aspas.

O computador pode ser instruído para ler (em inglês READ) mais do que um dado (em inglês DATA) de cada vez. Olhe para a linha 10 do programa. As duas variáveis da declaração READ estão separadas por uma vírgula.

A linha 20 poderia também ser: IF D=999 THEN END. Os dois 999 são dados "falsos".

```
1Ø READ C,D
2Ø IF C = 999 THEN END
3Ø PRINT C,D
4Ø GOTO 1Ø
5Ø DATA 5 2 8 12 6 10 999 999
```



A linha 10 dá instruções ao computador para ler *dois* números e armazenar o primeiro em C e o segundo em D. Primeiro o computador lê 5

e 2; no próximo ciclo lê 8 e 12; depois 6 e 10; e da última vez 999 e 999 e terminará. O *output* do programa será:

Na linha 30 a vírgula separa o *output* em duas zonas.

```
 5    2
 8    12
 6    10
```

O computador pode ler números e cadeias alfanuméricas, devendo apenas assegurarmo-nos que estas estão atribuídas a variáveis alfanuméricas e os números a variáveis numéricas. Eis um exemplo:

Os utilizadores de computadores Atari precisarão de uma declaração DIM para Z$.

Os dados "falsos" serão STOP,0,0,0,

```
10 READ Z$, A,B,C
20 IF Z$ = "STOP" THEN END
30 PRINT Z$,A,B,C,
40 GOTO 10
50 DATA NORTE, 2,7,3, SUL, 2,8,4, ESTE,5,2,9, OESTE,6,9,1, STOP,0,0,0
RUN
NORTE    2      7      3
SUL      2      8      4
ESTE     5      2      9
OESTE    6      9      1
```

Assegure-se de que as cadeias e os números da instrução DATA emparelham com as variáveis da instrução READ.

Veja como as variáveis numéricas e alfanuméricas da instrução READ estão organizadas de acordo com os números e as cadeias da declaração DATA:

```
10 READ Z$,A,B,C
20 DATA NORTE, 2, 7, 3
```

O computador começa por ler (em inglês READ) NORTE e atribui-o a Z$; 2 e atribui-o a A; 7 a B; e 3 a C; quando chega à linha 30 imprime NORTE 2 7 3 em zonas. Regressa à linha 10, lê mais dados e imprime-os nas quatro zonas da linha seguinte.

Mudemos a linha 30, de forma a efectuarmos um cálculo:

```
10 READ Z$, A,B,C
20 IF Z$ = "STOP" THEN END
30 PRINT Z$,A+B+C
4040 GOTO 10
50 DATA NORTE,2,7,3, SUL,2,8,4, ESTE,5,2,9, OESTE,6,9,1, STOP,0,0,0
RUN
NORTE      12
SUL        14
ESTE       16
OESTE      16
```

Desta vez quando o computador chega à linha 30 imprime na primeira zona a palavra armazenada em Z$ e na segunda zona a soma dos números armazenados em A, B e C.

Se soubermos quantas vezes deseja que o computador leia informações a partir da declaração DATA, não precisará de uma instrução IF-THEN para determinar a paragem do computador. Em vez disso podemos utilizar um

A instrução DATA pode colocar-se em qualquer lugar do programa.

ciclo FOR-NEXT. No programa seguinte o computador lê três pares de números:

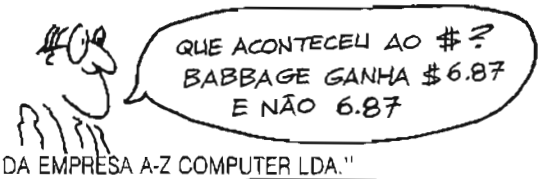

```
10 DATA 12,5,16,4,23,7
20 FOR X = 1 TO 3
30   READ A,B
40   PRINT A;"+";B;"=";A+B
50 NEXT X
RUN
12 + 5 = 17
16 + 4 = 20
23 + 7 = 30
```

De cada vez que o computador completa um ciclo, lê dois números e imprime-os juntamente com a resolução da operação de adição respectiva.

Muitos programas de computador que calculam os ordenados do pessoal de empresas usam instruções READ-DATA: o nome de cada empregado, o salário por hora e o número de horas trabalhadas são os dados (DATA) do programa. No programa seguinte há instruções DATA com informações relativas a quatro empregados, calculando e imprimindo o programa quanto recebe cada um. Leia o programa cuidadosamente: em cada instrução DATA o primeiro número é o vencimento à hora e o segundo o número de horas trabalhadas por cada pessoa; o salário total é obtido multiplicando o vencimento à hora pelo número de horas trabalhadas.

Pode pôr todos os dados numa única instrução, ou reparti-los por várias instruções. Veja as linhas 10-50 do programa.

Não há qualquer vencimento à hora que seja —$1. Por isso se usa —1 como dado "falso".




```
10 DATA BABBAGE, 6.87,38
20 DATA ECKERT, 8.45,41
30 DATA HOLLERITH, 12.06,42
40 DATA JACQUARD, 9,22,42
50 DATA END, —1, —1
80 PRINT "FOLHA DE PAGAMENTOS DA EMPRESA A-Z COMPUTER LDA."
90 PRINT "NOME", "SALÁRIO TOTAL"
100 READ N$,S,H
110 IF S = —1 THEN END
120 PRINT N$,S*H
130 GOTO 100
RUN
FOLHA DE PAGAMENTOS
DA EMPRESA A-Z COMPUTER LDA.
NOME          SALARIO TOTAL
BABBAGE       261,06
ECKERT        346,45
HOLLERITH     506,52
JACQUARD      387,24
```


tente responder

1. Escreva o *output* do programa seguinte:

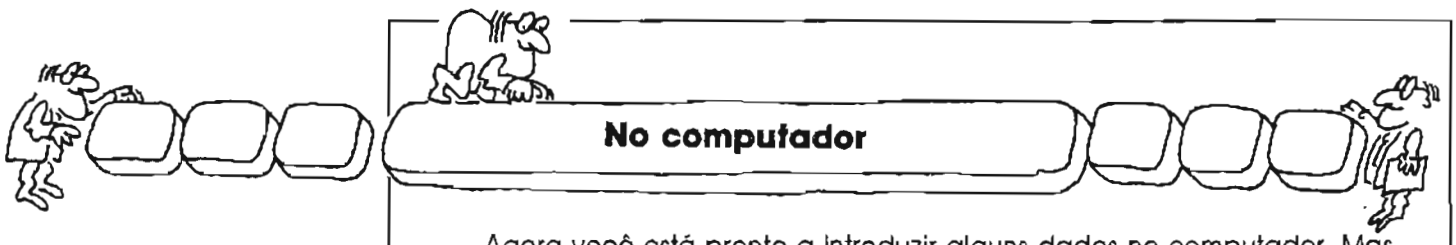
```
10 READ X
20 IF X = 99 THEN END
30 PRINT X
40 GOTO 10
50 DATA 2, 4, 6, 8, 10, 12, 99
```

2. Escreva o *output* do programa seguinte:

```
1Ø DATA FELIZ, TRISTE, PEQUENO, ALTO, X, X
2Ø READ A$, B$
3Ø IF A$ = "X" THEN END
4Ø PRINT A$, B$
5Ø GOTO 2Ø
```

3. Escreva o *output* do programa seguinte:

```
1Ø DATA ZE, 6, 2, EVA, 1Ø, 5, QUIM, 9, 4
2Ø FOR X = 1 TO 3
3Ø     READ A$,M,N
4Ø     PRINT A$,M,N
5Ø NEXT X
```



No computador

Agora você está pronto a introduzir alguns dados no computador. Mas primeiro atente no seguinte ALERTA DE ERROS relativo a toda a secção:

Uma instrução READ toda escrita correctamente pode originar um ERRO DE SINTAXE nessa linha: basta por exemplo ter posto na linha DATA uma vírgula a mais ou ter esquecido uma vírgula para o computador não ler (READ) a informação correctamente.

A situação oposta também pode ser verdadeira. Um ERRO DE SINTAXE numa linha com a instrução DATA pode significar que o erro é afinal na linha com instrução READ.

1. Escreva e corra o seguinte programa:

```
1Ø READ A$
2Ø IF A$="FIM" THEN END
3Ø PRINT A$
4Ø GOTO 1Ø
5Ø DATA MORCEGOS, LOMBRIGAS, RAS, LACRAUS, FIM
```

O seu *output* deverá ser constituído por quatro palavras em coluna.

Os utilizadores de computadores Atari escreverão:
5 DIM A$(15)

2. Escreva e corra o seguinte programa:

```
1Ø FOR X = 1 TO 4
2Ø     READ A,B,C,
3Ø     PRINT A+B+C
4Ø NEXT X
1ØØ DATA 6,3,2
11Ø DATA 5,5,5
12Ø DATA, 2,1,7
13Ø DATA 9,4,8
```

Introduza NEW.

Se desejar, combine estas quatro instruções numa única linha com uma só instrução DATA.

3. Altere a linha 30 para:

```
3Ø PRINT A;"+";B;"+";C;"=";A+B+C
```

Corra novamente o programa.

Introduza NEW.

4. Escreva o programa seguinte. Complete a instrução DATA com nomes de cinco dos seus alimentos favoritos. Corra o programa.

```
10 FOR X = 1 TO 5
20     READ N$
30     PRINT "EU GOSTO DE"; N$
40 NEXT N
100 DATA
```

Os utilizadores de computadores Atari escreverão:
5 DIM N$(25)

5. Junte outra instrução DATA ao seu programa. Coloque nela os nomes de dois programas de televisão. Corra o programa. Porque não aparecem no *output* os programas de televisão? Olhe para a linha 10: quantas repetições são agora necessárias? Mude a linha 10 em conformidade e corra novamente o programa.

Se tiver sete conjuntos de dados precisará de repetir o ciclo sete vezes.

6. Junte outra instrução DATA com o nome do seu desporto favorito. Lembre-se de mudar a linha 10. Corra o programa.

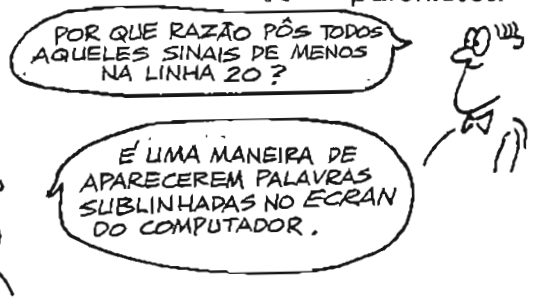
Introduza NEW.

7. O programa seguinte imprime os nomes de cinco estudantes e a sua classificação média relativamente a três testes. Escreva o programa, completando cada instrução DATA com um nome e três resultados de testes. Na linha 100 tem um exemplo. Corra o programa.

```
10 PRINT "NOME"; "PONTUACAO MEDIA"
20 PRINT "------","------------------"
30 FOR X = 1 TO 5
40     READ N$, A, B, C
50     PRINT N$,(A+B+C)/3
60 NEXT X
100 DATA SERGIO, 94, 86, 89
110 DATA
120 DATA
130 DATA
140 DATA
```

Os utilizadores de computadores Atari escreverão:
5 DIM N$(20)

Na linha 50 não se esqueça dos parênteses.



Você está a dar instruções ao computador para calcular e imprimir a média de três números, adicionando-os e dividindo a soma por três.

Introduza NEW.

8. Indicam-se em seguida os nomes de cinco estudantes que fazem parte de uma equipa de atletismo e os seus tempos em três corridas. Coloque estas informações em instrução DATA e imprima uma lista com o nome de cada estudante e o seu tempo médio:

Vasco	5.89	5.72	5.74
Lina	5.21	5.31	5.29
Jorge	6.42	6.21	6.98
Rui	5.25	5.21	4.01
Chica	6.31	5.98	5.52

Introduza NEW.

9. Use instruções DATA para escrever os nomes de quatro dos seus amigos, de qualquer coisa que eles gostem e de qualquer coisa de que eles não gostem. Por exemplo se o Paulo gostar de futebol e não gostar de matemática a sua declaração DATA será:

DATA PAULO, FUTEBOL, MATEMÁTICA

Escreva um programa que imprima uma linha para cada um dos seus amigos semelhante à seguinte:

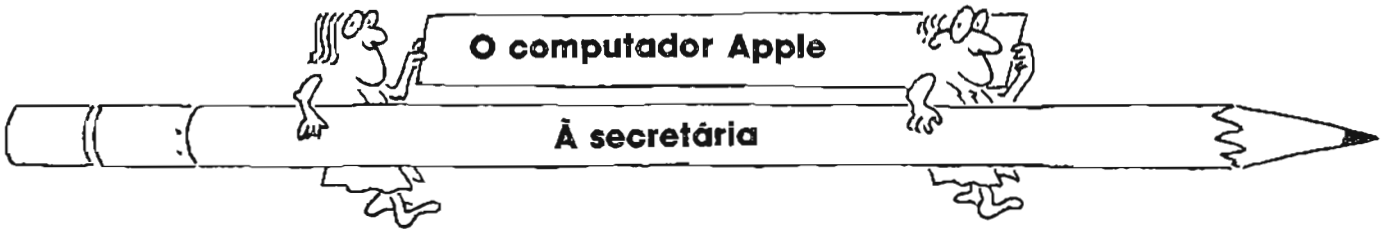
O PAULO GOSTA DE FUTEBOL MAS NÃO GOSTA DE MATEMÁTICA

Capítulo 24

ARTE COM O COMPUTADOR

Os microcomputadores podem ser programados para desenhar esquemas e figuras, os quais vão ser constituídos por blocos de luz e, nalguns computadores, por côres e formas especiais. A capacidade de um microcomputador mostrar gráficos e desenhos designa-se Gráficos por Computador (Computer Graphics).

Cada marca de computador tem uma forma específica de ser programada para gráficos. Este capítulo é dividido em quatro secções, cada uma delas dirigida a um computador diferente. Por isso e conforme o seu computador, consulte a secção respectiva: se tem um Apple, na pág. 165, se tem um Atari na pág. 169; se tem um Commodore na pág. 177 e se tem um ZX Spectrum na pág. 182.



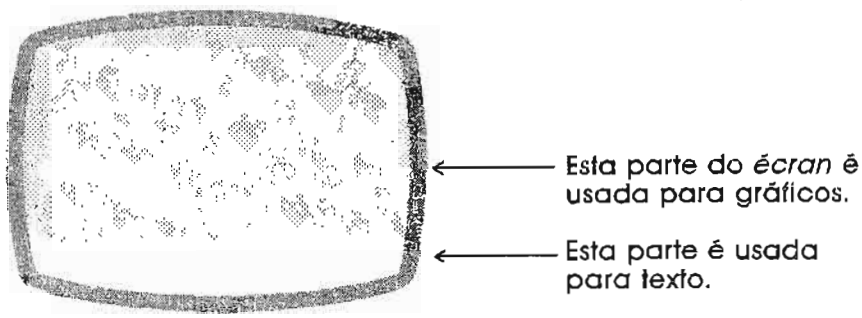
GR deriva de **gráficos.**

Texto significa letras, números e símbolos.

Para fazer desenhos no computador Apple precisa de usar nos seus programas uma declaração especial:

```
10 GR
```

A declaração GR altera o *écran* preparando-o para gráficos; para texto apenas podem ser usadas as quatro linhas do fundo do *écran* ficando o resto reservado para a figura que está a ser programada.

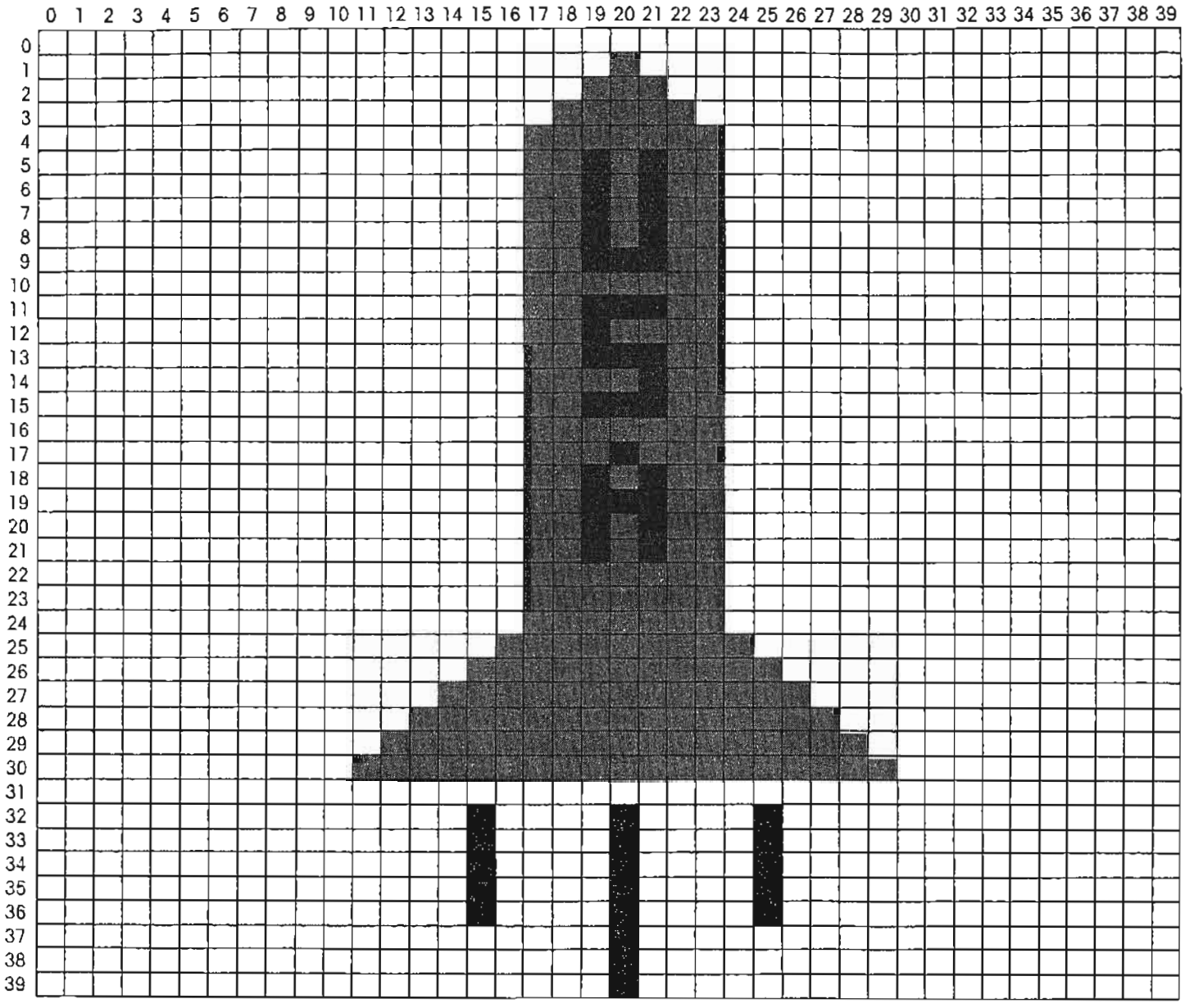


Quando os *pixels* são muito pequenos, os gráficos resultantes chamam-se de «alta resolução»; os gráficos com *pixels* maiores são gráficos de baixa resolução. Este capítulo diz respeito a gráficos de baixa resolução.

Imaginemos agora que a zona de gráficos de um *écran* de computador está coberta por uma quadrícula como o desenho da página seguinte e que se chama *pixel* a cada um dos seus muitos quadrados pequenos. A programação do computador para desenhar a nave espacial no *écran* baseia-se na coloração de certos *pixels* que assim constituem uma forma.

Horizontais

Folha de gráficos do computador Apple



A cor negra (0) é a cor de fundo do *écran*; ela só aparece quando usada sobre uma outra cor de fundo.

Para colorir cada pequeno quadrado do *écran* necessita de escolher uma cor. As possibilidades de escolha são as seguintes:

0 preto	6 azul-médio	11 cor-de-rosa
1 vermelho	7 azul-claro	12 verde
2 azul-escuro	8 castanho	13 amarelo
3 púrpura	9 cor-de-laranja	14 verde-mar
4 verde-escuro	10 cinzento	15 branco
5 cinzento		

Se desejasse colorir o "nariz" da nave espacial a verde, teria de usar as seguintes declarações:

```
COLOR = 12
PLOT 20,1
```

O número da cor verde é 12.

Para localizar um dos pequenos quadrados constituintes do *écran* apenas pode usar números de 0 a 39.

A declaração PLOT significa a iluminação de um *pixel*, localizado em 20,1. Para encontrar esta posição considere como origem o canto superior esquerdo do *écran*; mova-se horizontalmente até à coluna 20 e verticalmente até à linha 1: será aí a ponta do "nariz" da nave espacial.

É possível colorir qualquer *pixel* com uma declaração PLOT seguida de dois números separados por uma vírgula, representando o primeiro número a distância à margem esquerda e o segundo a distância à margem superior do *écran*.

Agora escolha a cor vermelha para colorir a nave espacial.

```
20 COLOR = 1
```

O Apple tem duas declarações que lhe permitem desenhar rectas horizontais e verticais: para desenhar a recta que constitui o lado esquerdo da nave espacial use uma declaração VLIN (VLIN deriva de *vertical line* em português "linha vertical").

Encontre esta linha no gráfico.

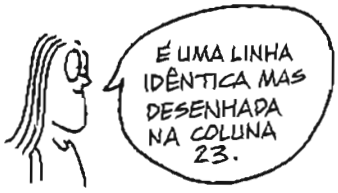
```
30 VLIN 4,24 AT 17
```

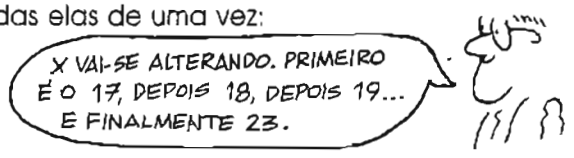
Esta declaração indica ao computador para desenhar na coluna 17 uma recta vertical entre as linhas 4 e 24. Para desenhar a recta do lado direito da nave escrever-se-ia:

```
40 VLIN 4,24 AT 23
```

Como pode verificar, ambas as rectas vão de 4 a 24, tal como todas as que constituem o corpo rectangular da nave espacial. Com um ciclo FOR--NEXT é possível desenhar todas elas de uma vez:

```
30 FOR X = 17 TO 23
40 VLIN 4,24 AT X
50 NEXT X
```





Ainda precisam de ser desenhados o topo e o fundo da nave espacial. Para desenhar linhas horizontais existe a declaração HLIN:

```
60 HLIN 18,22 AT 3
70 HLIN 19,21 AT 2
```

Encontre estas linhas no gráfico.

A linha 60 indica ao computador para desenhar uma recta na linha 3 entre as colunas 18 e 22; e a linha 70 manda desenhar uma horizontal na linha 2 entre as colunas 19 e 21.

Lembre-se que esta localização significa 20 no sentido horizontal e 1 na vertical, desde a origem no canto superior esquerdo.

O nariz da nave espacial é um *pixel* localizado em 20,1. A declaração seguinte (que já vimos anteriormente) vai colorir o rectângulo respectivo:

```
80 PLOT 20,1
```

A secção do fundo da nave foi propositadamente deixada para você executar na secção TENTE RESPONDER.

Agora escolha outra cor para as letras USA.

A parte principal da nave pode ser vermelha. Se pintar as letras sobre vermelho, elas ficam azul-escuro.

```
200 COLOR = 2
```

Neste ponto do programa, o computador muda para azul-escuro. As letras são programadas usando declarações VLIN, HLIN e PLOT. Esta parte do programa vai desenhar a letra U:

Na secção TENTE RESPONDER, ser-lhe-á pedido para programar as outras letras.

```
210 VLIN 5, 9 AT 19 ←     Lado esquerdo do U
220 VLIN 5, 9 AT 21 ←     Lado direito do U
230 PLOT 20,9 ←           Fundo do U
```

tente responder

Não se esqueça de usar as instruções que utilizou nos exercícios 1 e 2.

1. Qual é a declaração que dá instruções ao computador para limpar o *écran* para gráficos?

2. Que declaração usaria para desenhar em verde-escuro?

3. Escreva um programa que ilumine um *pixel* em cada um dos quatro cantos do *écran*.

4. Escreva um programa que desenhe uma recta horizontal no topo do *écran* e uma recta vertical a meio. As duas rectas parecer-lhe-ão como uma grande letra T (escolha uma cor para essas rectas).

5. Desenhe numa folha de gráficos as suas iniciais e depois escreva um programa que as desenhe no computador. Se alguma das letras tiver partes curvas (como a letra C), terá de as transformar em rectas ou diagonais. As letras poderão ser de quaisquer dimensões e cores.

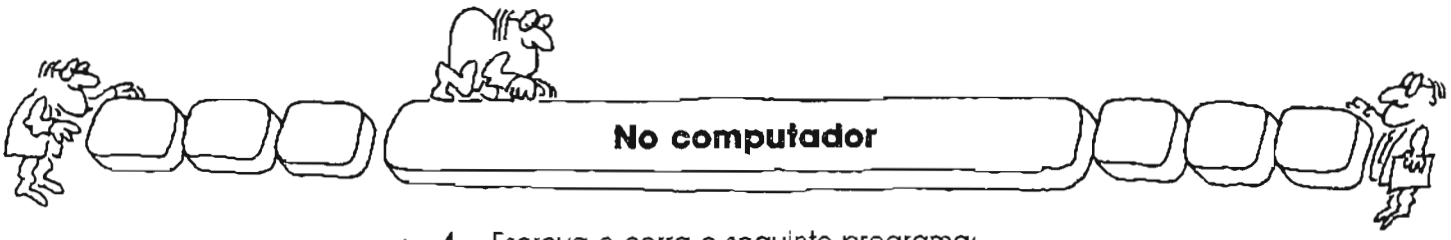
6. Escreva um programa que encha todo o *écran* de gráficos numa cor à escolha; utilize rectas horizontais ou verticais.

7. Escreva um programa que desenhe a base da nave espacial da página 166.

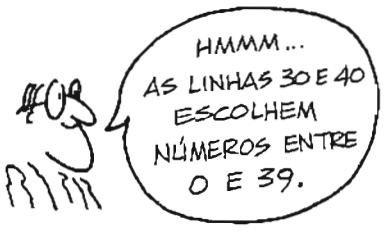
8. Escreva um programa que desenhe as chamas que saem da nave espacial da página 166.

Pode escolher uma cor diferente para cada inicial.

9. Escreva um programa para as letras S e A de USA.



No computador



1. Escreva e corra o seguinte programa:

```
10 GR
20 COLOR = 3
30 H = INT(RND(1)*40)
40 V = INT(RND(1)*40)
50 PLOT H, V
60 GOTO 20
```

2. Pare o programa que está a correr e liste-o. Dadas as dimensões do *écran* de texto você não pode ler todas as linhas. Para alterar todo o *écran* para texto prima o comando:

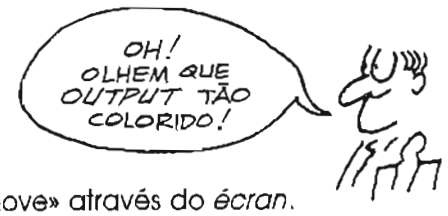
```
TEXT
```

Pode limpar o *écran* escrevendo HOME.

O *écran* de gráficos mudar-se-á para texto. Agora pode listar o programa. Altere a linha 20 para:

```
20 COLOR = INT(RND(1)*15) + 1
```

Corra o programa novamente.



Agora a cor pode ser uma qualquer entre 1 e 15.

Escreva NEW.

3. Desenhe um quadrado que se «move» através do *écran*.

```
10 GR
20 FOR H = 0 TO 39
30 COLOR = 9
40 PLOT H, 20
50 COLOR = 0
60 PLOT H,20
70 NEXT H
0 GOTO 20
```

Como poderá ver o quadrado realmente não se move. O programa colora um *pixel* a cor-de-laranja e depois "apaga-o" coloríndo-o de preto que é a cor de fundo. Depois é o *pixel* seguinte que é colorido e apagado, etc.

Escreva NEW.

4. Desenhe um quadrado que se "mova" no *écran* de cima para baixo.

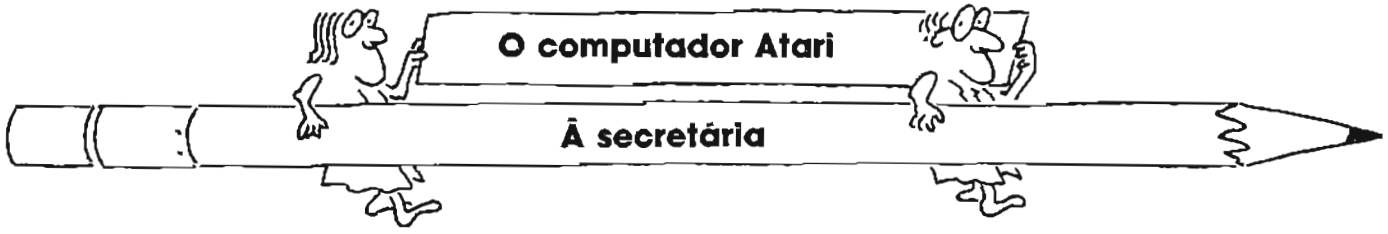
Escreva NEW.

5. Escreva e corra um programa que desenhe as suas iniciais. Use o programa que utilizou em "TENTE RESPONDER", exercício 5.

ALERTA DE ERROS: não use números maiores que 39 ou obterá uma mensagem de erro!

Escreva NEW.

6. Desenhe toda a nave espacial representada na página 166. Parte desse trabalho está feito em "À secretária". Utilize as cores que desejar.



Para desenhar figuras num computador Atari precisa de usar nos seus programas uma instrução especial:

```
GRAPHICS 3 (ou GR.3)
```

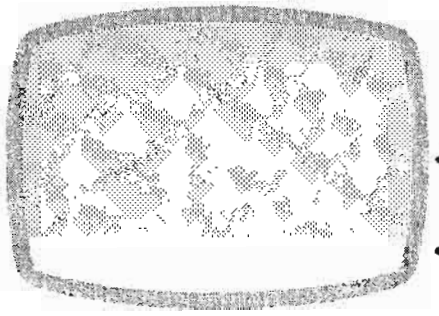
Nos modelos mais antigos pode haver menos números para usar.

Existem onze números (de 0 a 10) que podem ser utilizados com a instrução GRAPHICS; por agora iremos apenas usar GRAPHICS 3.

Esta declaração altera o *écran* preparando-o para receber gráficos:

Texto significa letras, números e símbolos.

apenas podem ser utilizadas para texto as quatro linhas do fundo do *écran*; o resto é reservado à figura que está a ser programada.



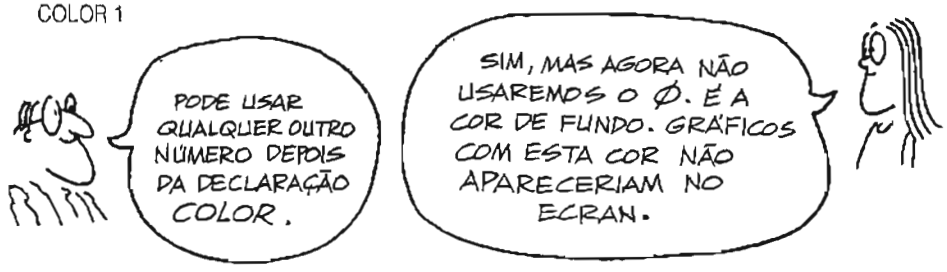
Esta parte do *écran* é usada para gráficos.

Esta parte é usada para texto.

Imagine que a parte superior do *écran* é um gráfico formado por quadrados muito pequenos chamados *pixels*, como a figura da página seguinte. A programação do computador para desenhar figuras no *écran* de gráficos corresponde à coloração dos *pixels* de acordo com um modelo determinado.

Para ver alguma coisa no *écran* tem de indicar ao computador que quer usar cores escrevendo:

```
COLOR 1
```



Para iluminar um *pixel* no *écran* usa-se a declaração PLOT. Identifica-se um quadrado determinado fazendo-lhe corresponder dois números como:

```
PLOT 19, 13
```

Esta instrução identifica um *pixel* na posição 19, 13. Para encontrar esta posição, parta do canto superior esquerdo do *écran*, horizontalmente até à coluna 19 e para baixo até à linha 13. Verá que aquele quadrado corresponde ao topo da letra A da Nave espacial.

Com instruções PLOT e dois números separados por uma vírgula, pode colorir qualquer *pixel* do *écran*. Não esqueça que o primeiro número nos dá a distância a que estamos da margem esquerda do *écran* e o segundo número a distância a contar da margem superior.

Se está a pensar que desta forma levaria muito tempo a «identificar» *pixels* para desenhar uma figura, anote que o computador Atari usa uma instrução que pode simplificar o seu trabalho. Veja o programa seguinte:

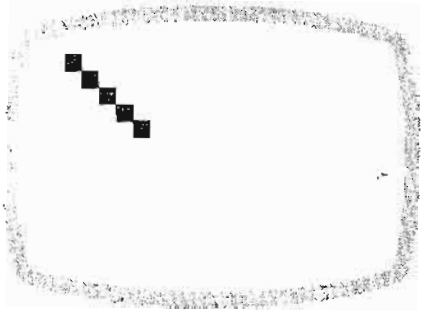
```
10 GR3 ←
20 COLOR 1 ←
30 PLOT 0,0 ←
DRAWTO 5,5 ←
```

Prepara o *écran* para gráficos.
Manda utilizar cor.
Desenha um quadrado na posição 0 a partir da esquerda e 0 a partir de cima.
Uma instrução nova.

Procure estes *pixels* no gráfico.

A nova instrução DRAWTO na linha 40 significa "desenhar uma linha até" Assim o computador depois de identificar um quadrado na posição

0,0 (linha 30) desenhará uma recta desde esta posição até à posição 5,5.
O *output* é uma diagonal, como a representada na figura seguinte:



Vamos agora ver um programa para parte da nave espacial colorida, representada na folha de gráficos. Usam-se as instruções COLOR, PLOT e DRAWTO:

```
1Ø GR.3
2Ø COLOR 1
3Ø PLOT 17,2: DRAWTO 17, 13
4Ø PLOT 18,1: DRAWTO 18, 13
5Ø PLOT 19,Ø: DRAWTO 19, 13
6Ø PLOT 2Ø,1: DRAWTO 2Ø, 13
7Ø PLOT 21,2: DRAWTO 21, 13
8Ø PLOT 16,14: DRAWTO 22, 14
9Ø PLOT 15,15: DRAWTO 23, 15
1ØØ PLOT 14,16: DRAWTO 24, 16
```

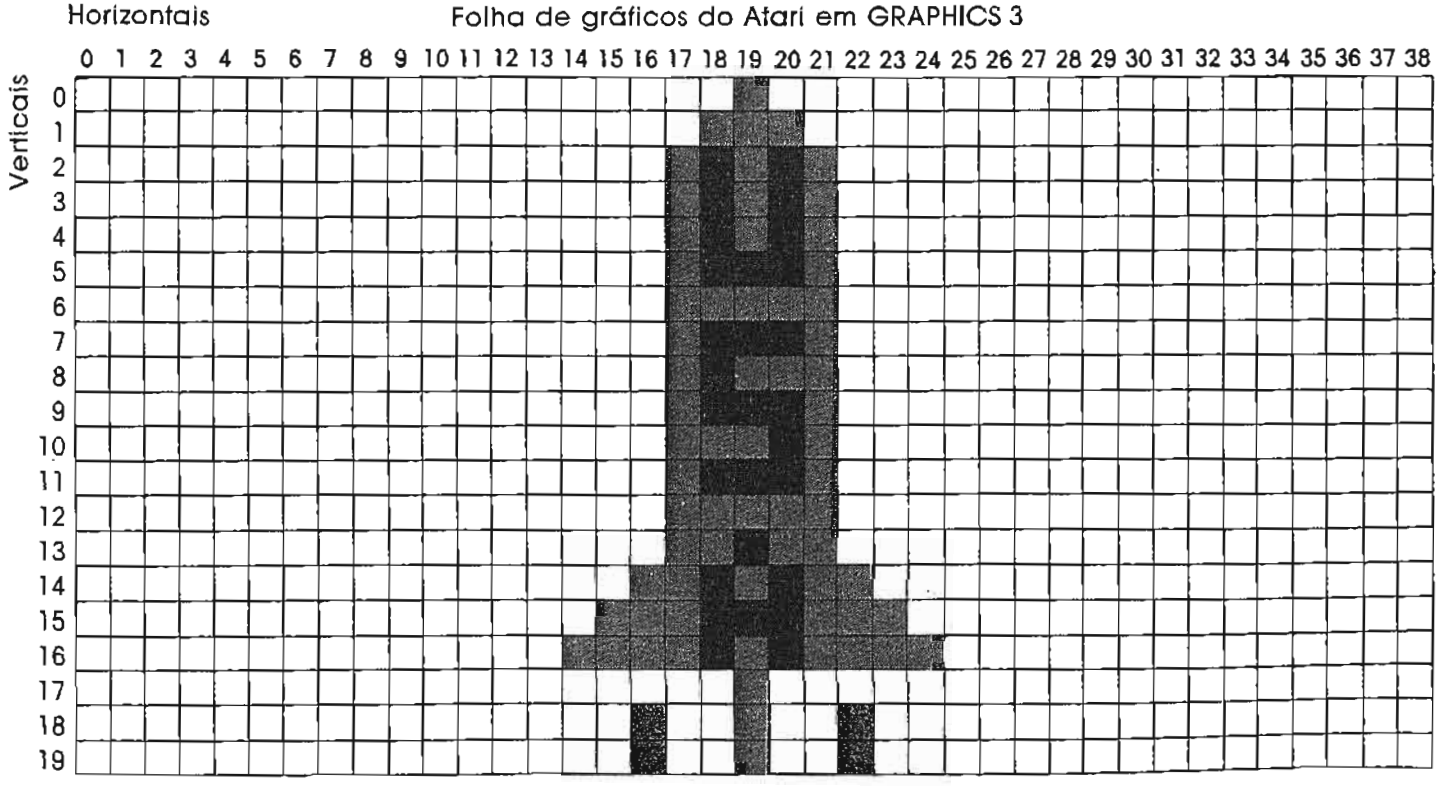
Estas linhas desenham a parte de baixo da nave espacial.

Procure no gráfico estes *pixels* e linhas.

Lembre-se que quando tem duas instruções numa linha deve usar os dois pontos a separá-las.

Se não indicar ao computador as cores que deseja utilizar ele escolherá por si: o negro para fundo dos gráficos, o azul para fundo de texto e o laranja para os quadrados que constituem a figura. Se quiser escolher as suas próprias cores deve usar, juntamente com COLOR, uma nova instrução: SETCOLOR.

Folha de gráficos do Atari em GRAPHICS 3



SETCOLOR significa que deve fixar (em inglês *set*) ou escolher uma cor (em inglês *color*). Esta instrução é seguida por três números, como no exemplo seguinte:

```
SETCOLOR Ø, 12, 2
```

Quando está em GRAPHICS 3 não pode usar o número 3.

Vejamos o significado de cada um dos números. O primeiro número escolhe o que deseja colorir é o chamado registo de cor. Com GRAPHICS 3 pode utilizar os números 0, 1, 2 e 4:

0 e 1 significam escolher uma cor para os quadrados que constituem a figura;

2 significa uma cor para o fundo de texto;

4 significa uma cor para o fundo de gráficos.

Em SETCOLOR 0, 12, 2 o registo de cor é 0, o que significa que se irá escolher uma cor para os quadrados que constituem os gráficos.

O segundo número escolhe a cor. Podemos usar números de 0 a 15:

0 cinzento	6 cor-de-alfazema	11 azul-esverdeado
1 dourado	7 azul	12 verde
2 laranja	8 azul	13 amarelo-esverdeado
3 vermelho-alaranjado	9 azul-claro	14 verde-alaranjado
4 cor-de-rosa	10 turquesa	15 laranja-claro
5 púrpura		

O número 10 provoca um *écran* branco. Não o use!

Sendo 12 o segundo número da instrução SETCOLOR, significa que se escolheu a cor verde: a instrução SETCOLOR vai indicar ao computador para colorir de verde os quadrados dos gráficos (porque o registo da cor é 0).

O terceiro número escolhe o brilho da cor. Pode usar números entre 0 e 14, correspondendo o 0 ao mais claro e o 14 ao mais escuro. O terceiro número da instrução SETCOLOR atrás referida, 2, significa que o computador irá colorir os quadrados de um verde muito claro.

SETCOLOR apenas *escolhe* o que se quer colorir (registo de cor, cor e brilho); para *pôr* a cor no *écran* é necessária uma instrução COLOR, seguida por 0, 1, 2 ou 3, número sempre superior em 1 unidade ao registo de cor de SETCOLOR (se o registo de cor em SETCOLOR for 4 então o número de cor é 0). Eis alguns exemplos:

O registo de cor é o primeiro número depois de SETCOLOR.

SETCOLOR.

```
SETCOLOR Ø, 12, 8
COLOR 1

SETCOLOR 2, 15, 8
COLOR 3

SETCOLOR 1, 7, 8
COLOR 2

SETCOLOR 4, 2, 8
COLOR Ø
```

Olhe para a página 171

Agora já pode escolher cores para o programa da nave espacial. Estas linhas irão substituir as anteriores linhas 10 e 20.

```
10 GR. 3
12 SETCOLOR 4, 9, 14
14 COLOR 0
16 SETCOLOR 0, 0, 12
18 COLOR 1
```

A linha 12 escolhe o azul-claro para fundo de gráficos e a linha 14 põe a cor no *écran*; a linha 16 escolhe o cinzento para cor dos quadrados dos gráficos que constituem a náve espacial, e a linha 18 põe essa cor no *écran*.

Em TENTE RESPONDER ser-lhe-á pedido para programar as outras letras.

Agora está em condições de desenhar a letra U. Mas primeiro tem de mudar a cor, sem o que a cor das letras será a mesma que a da nave espacial e a letra U não sobressairá.

```
100 SETCOLOR 1, 3, 2
110 COLOR 2
120 PLOT 18, 2: DRAWTO 18, 5
130 PLOT 20, 2: DRAWTO 20, 5
140 PLOT 19, 5
```

Escolhe uma cor e um brilho para os quadrados do gráficos e põe a cor no *écran*.

Desenha o U.

A linha 100 é muito importante. Lembre-se que pode usar 0 e 1 para registo de cor, de forma a escolher uma cor e um brilho para os quadrados de gráficos. Como já usou 0 na linha 16, se o usasse novamente o computador alteraria todos os quadrados do gráfico para a nova cor. Como deseja que a letra U tenha uma cor diferente, deve usar um registo de cor diferente para estes quadrados. Por isso se usa o 1.

Os quatro registos de cor 0, 1, 2 e 4 permitem-lhe ter no *écran,* de cada vez, quatro cores diferentes. No programa da nave espacial os quadrados dos gráficos (registos 0 e 1) são cinzentos e vermelho-alaranjado. O fundo dos gráficos (registo 4) é azul-claro. Não foi escolhida qualquer cor para fundo de texto e por isso, automaticamente, ele é azul.

O manual do seu computador Atari tem tabelas que mostram quais os registos de cor que se devem usar. Mas também pode experimentar.

Até aqui vimos apenas a instrução GRAPHICS 3. GRAPHICS 4, 5, 6, 7, 8, 9 e 10 trabalham como GRAPHICS 3, mas funcionam com maior número de *pixels* (mais pequenos) no *écran* de gráficos. Alguns números de gráficos utilizam registos de cor diferentes dos que são utilizados em GRAPHICS 3.

Também pode usar GR.0 para limpar o *écran*.

Para textos usa-se GRAPHICS 0. Quando liga o computador ele é colocado automaticamente em GR.0; enquanto esteve a escrever os programas apresentados nos capítulos anteriores deste livro, você esteve a utilizar GRAPHICS 0.

GRAPHICS 1 e GRAPHICS 2 são especiais. Permitem-lhe escrever **textos gráficos**, ou seja letras, números e símbolos que vão aparecer no *écran* de gráficos. Eis um programa para imprimir texto gráfico em GRAPHICS 1 e texto regular no *écran* de texto.

Com GR.1 e GR.2 não é necessária a instrução COLOR.

```
10 GR. 1
20 PRINT "ISTO E TEXTO."
30 PRINT #6; "ISTO E TEXTO GRAFICO"
```

Na linha 30 PRINT #6; indica ao computador para imprimir em texto gráfico. O *output* do programa parecer-se-á com:

Atenção ao desenho do original e ao facto de que o Atari **não** tem palavras acentuadas ou Ç.

GRAPHICS 2 é idêntico, apenas as letras são maiores.

tente responder

1. Das seguintes três instruções de gráficos: GR.0, GR.1 e GR.3
   a. Que instrução é usada para fazer aparecer o *écran* gráfico?
   b. Que instrução é usada para o texto gráfico?
   c. Que instrução é usada apenas para texto?

2. Indique a instrução que:
   a. Escolha um registo de cor, uma cor e um brilho.
   b. Indique ao computador para desenhar com a cor e o brilho escolhidos.
   c. Desenhe um quadrado.
   d. Desenhe uma recta.

3. O programa abaixo desenha um quadrado no meio do *écran*:

```
10 GR. 3
20 SETCOLOR
30 COLOR
40 SETCOLOR
50 COLOR
60 PLOT 19, 10
```

   a. Escreva as linhas 20 e 30 de tal forma que o fundo fique cor-de-laranja claro (escolha o brilho que quiser).
   b. Escreva as linhas 40 e 50 para tornar o quadrado azul (use o brilho que desejar).

4. Escreva um programa para identificar um *pixel* em cada um dos quatro cantos do *écran*.

Escolha uma cor para os quadrados dos gráficos.

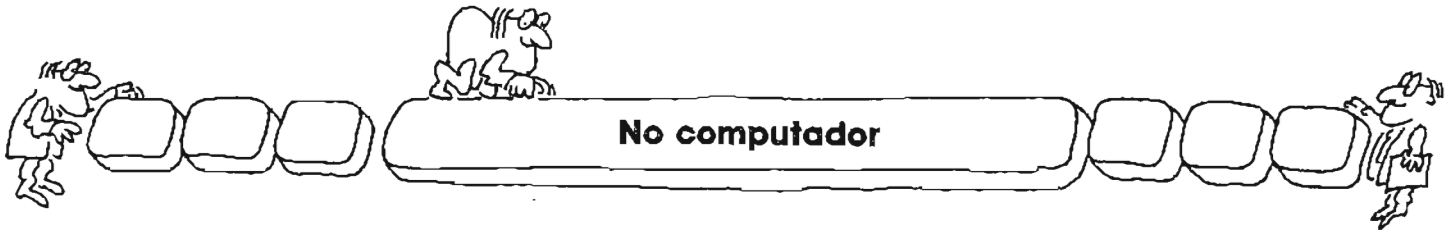
Escolha uma cor para os *pixels*.

Só há dois números que pode usar para texto gráfico.

5. Escreva um programa para desenhar uma linha no topo do *écran* e uma linha vertical no lado direito do *écran*.

6. Usando texto gráfico escreva um programa para imprimir o seu nome no *écran* gráfico. (Recorde que PRINT # 6 é a instrução para texto gráfico).

7. Escreva um programa para desenhar as letras S e A da nave espacial.



No computador

1. Escreva e corra o seguinte programa, usando o seu nome na linha 120.

```
10 GR. 3
30 COLOR 1
70 PLOT 0, 0
120 PRINT "GRAFICOS FEITOS POR MARCO ANTONIO"
```

Verá no canto superior esquerdo do *écran* um quadrado laranja; o fundo de gráficos é negro e o fundo de texto, na parte inferior do *écran*, é azul. GRÁFICOS FEITOS POR (o seu nome) aparece no *écran* de texto.

2. Junte uma outra declaração na linha 70. Se começou a escrever agora, a sua nova linha 70 será apresentada na parte de textos do *écran*. Escreva:

Não esqueça os dois pontos.

```
70 PLOT 0, 0: DRAWTO 19, 19
```

Tente listar o programa. Parte dele desaparece pelo cimo do *écran* de texto. Limpe o *écran* de gráficos escrevendo GR. 0, sem número de linha. Agora poderá listar o programa no *écran* de texto normal: se correr o programa, verá uma linha diagonal.

Pode também premir SISTEM RESET.

3. Escreva GR. 0 e liste o programa. Junte a linha seguinte:

```
20 SETCOLOR 4, 5, 8
```

Altere a linha 30 para:

```
30 COLOR 0
```

Corra o programa. Você mudou a cor de fundo do *écran* de gráficos. Escreva a linha 20 várias vezes, mudando a cor e o brilho da linha, mas mantendo o registo de cor, o primeiro número, 4, pois apenas queremos mudar a cor de fundo.

4. Junte as linhas seguintes:

```
100 SETCOLOR 2, 14, 8
110 COLOR 3
```

Você está a alterar a cor do fundo do texto. Altere os números relativos à cor e ao brilho várias vezes.

5. Altere a cor da diagonal. Junte as linhas seguintes ao programa:

```
50 SETCOLOR 0, 12, 2
60 COLOR 1
```

Corra o programa.

6. Pode utilizar um ciclo FOR-NEXT para alterar a cor da diagonal de forma a fazê-la passar por todas as cores. Junte as seguinte linhas:

```
40 FOR C = 0 TO 15
90 NEXT C
```

Altere a linha 50 para:

```
50 SETCOLOR 0, C, 8
```

Corra o programa e veja como as cores mudam. Mudam depressa de mais? Ponha uma pausa no fim de cada linha. Junte:

```
80 FOR P = 1 TO 200: NEXT P
```

Corra o programa novamente.

Introduza **NEW**.

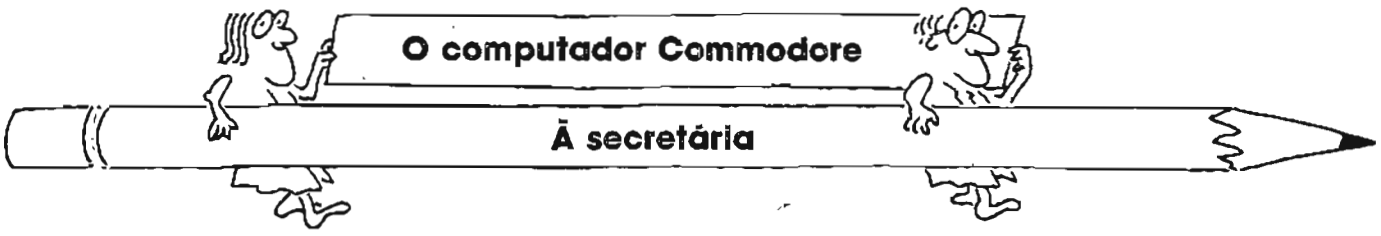
7. Desenhe um quadrado que se "mova" através do *écran*.

```
10 GR.3
20 FOR A = 0 TO 38
30   SETCOLOR 0, 4, 6
40   COLOR 1
50   PLOT A, 10
60   COLOR 0      <-------- COLOR 0 sem SETCOLOR altera os
70   PLOT A, 10             pixels para a cor do fundo,
80 NEXT A                   "apagando-os".
```

Corra o programa. É evidente que nada se está realmente a mover: o computador marcou (em inglês *PLOT*) um *pixel*, "apagou-o"; depois marcou outro *pixel* no espaço seguinte, "apagou-o" novamente, etc.

8. Desenhe um quadrado que se "mova" no *écran* verticalmente desde o topo até ao fundo.

9. Desenhe toda a nave espacial representada na folha de gráficos. Parte do programa já foi apresentado na *secção* "À secretária". O resto do programa foi escrito em "Tente responder". Escolha as cores para o fundo, para a nave espacial e para as letras. Na janela de texto imprima uma mensagem como PROGRAMADO POR (o seu nome).



Imagine que o *écran* do seu computador é um gráfico constituído por pequenos quadrados chamados *pixels,* como a figura da página seguinte. A programação do computador Commodore para desenhar baseia-se no preenchimento de determinados quadrados com caracteres gráficos que não são mais do que símbolos ou pequenos desenhos marcados nas teclas e que podem ser impressos no *écran* utilizando a tecla SHIFT. Por exemplo para imprimir um losango terá que premir SHIFT e a tecla Z.

Suponha que desejava preencher um *pixel* com um círculo negro como o existente no canto superior esquerdo da folha de gráficos. Primeiro assegure-se que o cursor está no topo do *écran;* para isso poderá apagar todo o *écran* usando a seguinte instrução:

```
10 PRINT "□"
```

O coração em negativo tem as cores invertidas.

Para imprimir o coração prima a tecla SHIFT-CLR. Verá aparecer no *écran* o símbolo de um **coração em negativo.**

Já utilizou a tecla CLR para limpar o *écran*. Escrevendo-a numa instrução PRINT como parte do nosso programa, estamos a dar instruções ao computador para limpar o *écran* quando se corre o programa.

A segunda linha do programa é:

```
20 PRINT "•"
```

← Para escrever o círculo prima SHIFT e Q.

Procure o círculo na folha de gráficos.

Quando corre o programa o círculo aparece no princípio da primeira linha.

Olhe para a folha de gráficos: os números vão horizontalmente de 0 a 39 e verticalmente de 1 a 25. Se quisesse pôr o círculo no canto superior direito do *écran*, poderia pôr 39 espaços entre aspas e depois o círculo:

```
20 PRINT "                                      •"
```

No 40.° espaço, pois tem de contar com o 0.

Não tem de contar espaços.

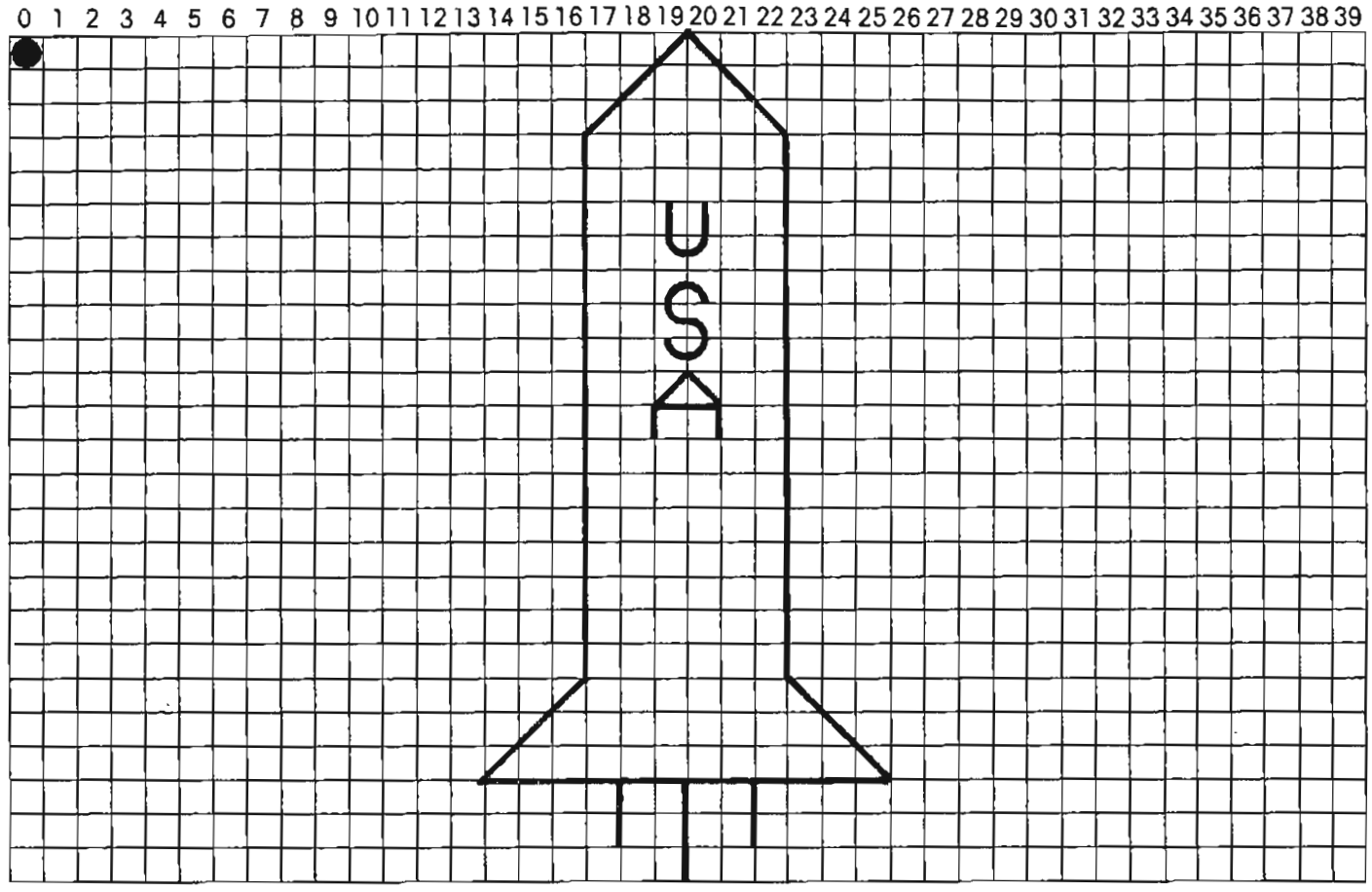
Mas existe uma forma abreviada de escrever esta instrução usando uma nova instrução — TAB — que faz o computador imprimir o círculo no quadrado 39:

```
20 PRINT TAB(39) "•"
```

Horizontais

Folha de gráficos do Computador Commodore

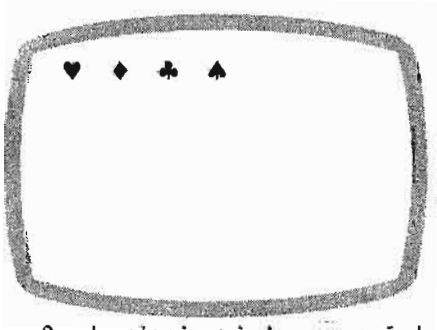
Verticais



Na linha 20 o coração é o coração normal que é obtido premindo as teclas SHIFT e S.

Observe o programa e o *output* abaixo representados. O programa limpa o *écran* e usa instruções TAB para imprimir os símbolos:

```
1Ø PRINT "□"
2Ø PRINT TAB(5) "♥" TAB(1Ø) "♦" TAB(15) "♣" TAB(20) "♠"
```



Se desejar imprimir um carácter gráfico mais abaixo do *écran*, pode usar uma instrução PRINT. Esta instrução, isoladamente, indica ao computador que deve imprimir uma linha branca; aplicando este princípio, o programa seguinte leva o computador a imprimir um losango cinco linhas abaixo, no quadrado 9 a contar da esquerda:

```
1Ø PRINT "□"
2Ø PRINT: PRINT: PRINT: PRINT ←——— 4 linhas em branco
3Ø PRINT TAB(9) "♦"
```

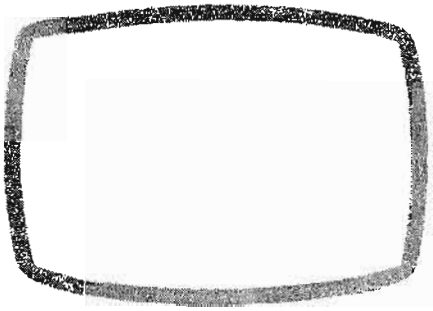
Procure este quadrado na folha de gráficos.

Como há 25 linhas no *écran*, se quiser colocar um carácter gráfico no fundo do *écran* terá de escrever os 24 PRINTs equivalentes a outras linhas em branco. Mas já conhecemos uma maneira abreviada de conseguir fazer isto, utilizando um ciclo FOR-NEXT:

```
10 PRINT "□"
20 FOR X = 1 TO 24
30    PRINT
40 NEXT X
50 PRINT TAB(20) "O"
```

24 linhas em branco

O *output* seria:



Podemos juntar quantos caracteres gráficos quisermos para fazer um desenho. Eis um programa para desenhar a parte superior da nave espacial:

```
10 PRINT "□"
20 PRINT TAB(19) "/\"
30 PRINT TAB(18) "/  \"
40 PRINT TAB(17) "/    \"
```

Olhe para a folha de gráficos para encontrar o *output* das linhas 20, 30 e 40.

Podemos desenhar o corpo da nave espacial utilizando um ciclo FOR--NEXT. A seguinte parte do programa desenha duas rectas descendo desde a linha 4 à 19:

```
50 FOR X = 4 TO 19
60    PRINT TAB(17) "|    |"
70 NEXT X
```

Se só houvesse 16 linhas no modelo poderia também escrever:
50 FOR X = 1 TO 16

Na secção TENTE RESPONDER ser-lhe-á pedido para programar a parte inferior da nave espacial: na secção "No seu computador" ser-lhe-á pedido para programar as letras USA.

tente responder

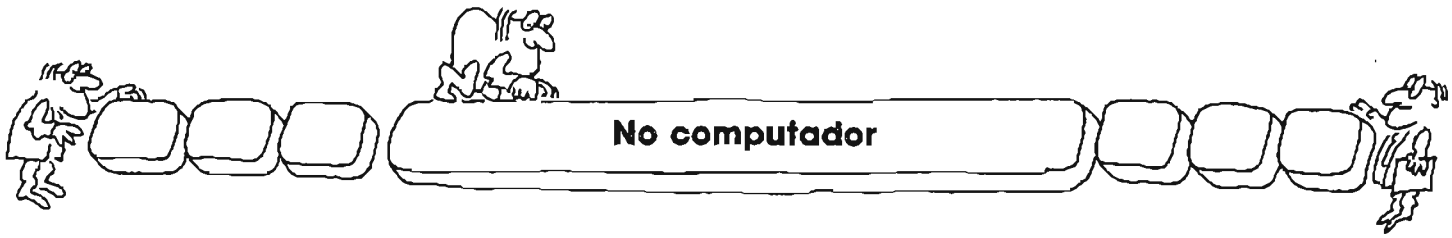
1. Que tecla adicional deve premir quando quer imprimir um carácter gráfico?

2. Desenhe numa folha de gráficos uma figura utilizando qualquer dos símbolos que viu nesta secção. Escreva um programa para desenhar essa figura no *écran*.

3. Continue o programa da nave espacial desenhando o respectivo fundo. Comece na linha 80.

Com o auxílio de instruções TAB e PRINT ponha os símbolos na posição correcta.

4. Continue a nave espacial desenhando as chamas que saiem da sua base.



No computador

1. Pratique escrevendo vários carácteres graficos. Lembre-se que tem de manter premida a tecla SHIFT: ou então utilizar a tecla SHIFT/LOCK que mantem o teclado na posição SHIFT até se premir novamente a tecla SHIFT/LOCK.

2. O computador desenhará um carácter gráfico no local onde estiver o cursor. É importante saber como mover o cursor enquanto se escreve o programa. Primeiro treine a limpeza do *écran*. Escreva:

É uma boa prática limpar o *écran* no início de todos os programas.

```
10 PRINT "♥"
```

Prima SHIFT-CLR para obter o coração.
Não esqueça as aspas.

Corra o programa. Repare que o *écran* está limpo e o cursor no canto superior esquerdo.

3. Agora escreva:

Lembre-se que os dois pontos lhe permitem pôr mais do que uma instrução numa linha.

```
20 PRINT: PRINT: PRINT
30 PRINT "O"
```

← Prima SHIFT — W

Corra o programa: obterá três linhas brancas e o círculo na quarta linha.

4. Outra maneira de mover o cursor é utilizar uma declaração PRINT, aspas e a tecla CRSR ↓. Premindo esta última tecla (dentro de aspas) obtem-se no *écran* um Q em negativo. Escreva as linhas seguintes:

```
40 PRINT "■■■"
50 PRINT "•"
```

Corra o programa: obterá três linhas em branco e depois um círculo negro.

5. Agora imprima um losango além do círculo: deverá usar as teclas CRSR ↑ (entre aspas) e SHIFT; obtém-se um ponto negativo no *écran*. Escreva:

Pode combinar símbolos numa instrução.

```
60 PRINT "■ ♦"
```

← Este é o losango de gráficos (SHIFT-Z).
↑ Este é o sinal obtido pela tecla CRSR ↑

Agora corra o programa.

Recorde que o Q negativo é obtido premindo a tecla CRSR ↓.

6. Já aprendeu a mover o cursor para a direita utilizando uma declaração TAB. Escreva:

```
70 PRINT "■■■■■" TAB(10) "♠"
```

A linha 70 move o cursor cinco linhas para baixo a partir do losango e imprime uma espada na 6.ª linha, 10.ª coluna.

7. Existe um outro movimento importante do cursor: colocá-lo na **posição inicial** (em inglês *home*, de lar), no canto superior esquerdo do *écran*. Para mover o cursor para aquela posição e imprimir um coração no *écran* escreva esta linha:

```
80 PRINT "■♥"
```

Para obter o S negativo prima a tecla HOME (dentro de aspas) sem premir a tecla SHIFT. Agora corra o programa.

8. Liste o programa que iniciámos na pergunta 2. Obteremos:

```
10 PRINT "■"
20 PRINT: PRINT: PRINT
30 PRINT "O"
40 PRINT "■■■"
50 PRINT " • "
60 PRINT "■ ♦"
70 PRINT "■■■■■" TAB(10) "♠"
80 PRINT "■♥"
```

Depois de correr o programa verá no *écran* a mensagem READY. Ao criar uma figura com caracteres gráficos, a mensagem READY obtém-se no lugar da figura. Para a eliminar faça o programa *nunca acabar* escrevendo:

```
90 GOTO 90
```

Recorde que deve premir STOP quando quer recomeçar a escrever.

Corra o programa. A linha 90 vai manter o computador ocupado: como o programa não acaba, a mensagem READY não aparece. Por isso deverá usar uma linha com um ciclo infinito no final de todos os seus programas gráficos.

Introduza NEW.

9. Escreva numa folha de gráficos as suas iniciais usando o tipo de linhas que desejar: finas, grossas, curvas, blocos, triângulos, etc. Depois escreva um programa para desenhar essas iniciais no *écran*.

Introduza NEW.

10. Programe toda a nave espacial da página 178, mas sem as letras USA.

Introduza NEW.

11. Agora acrescente as letras USA. Use caracteres gráficos para fazer as letras. Sugestão: use neste programa as teclas HOME, CRSR ↑ e CRSR ↓.

À secretária

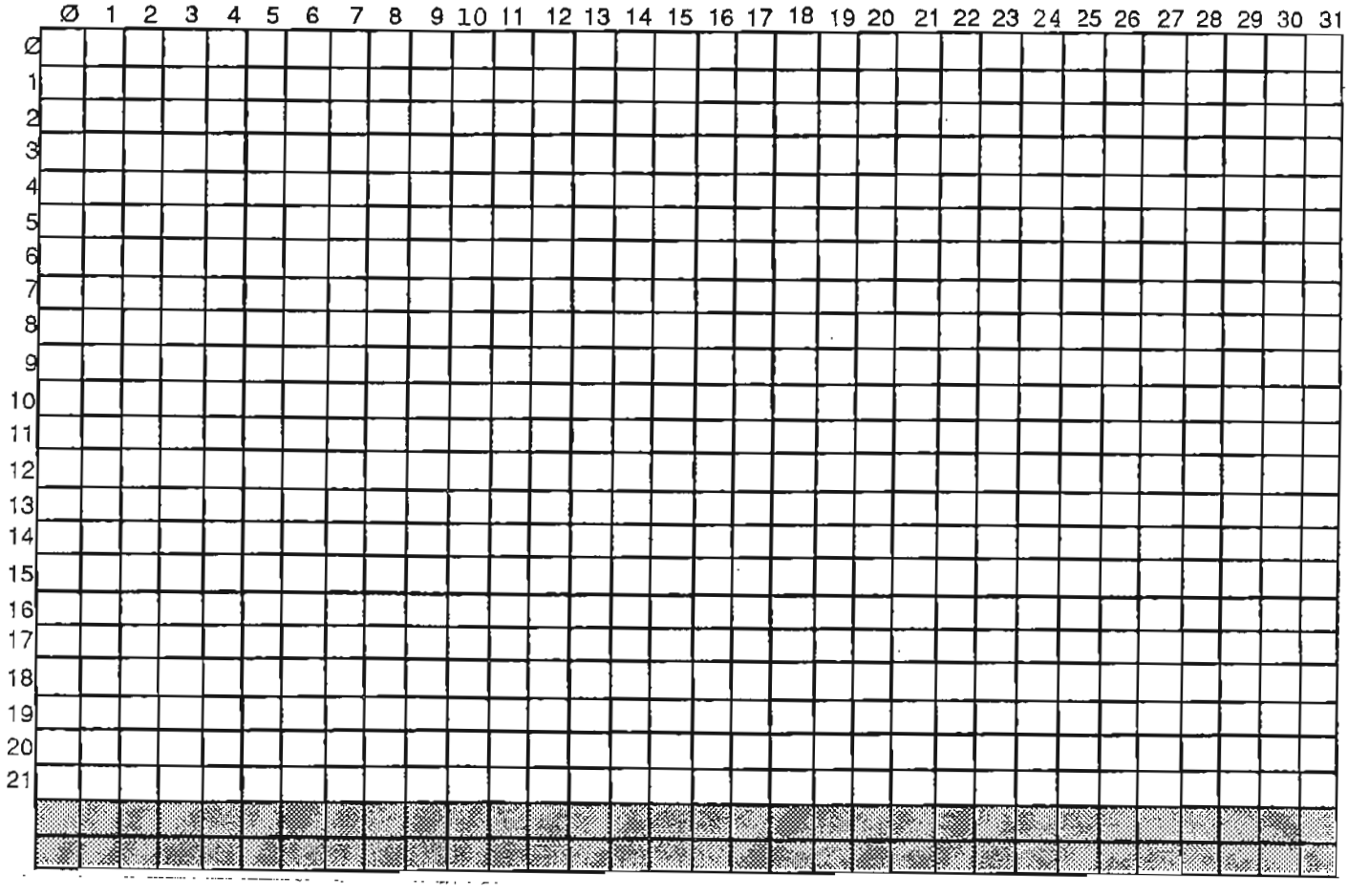
Nalguns computadores terá de escolher o modo compatível com o Spectrum/2048.

O computador ZX Spectrum (e os seus compatíveis, por exemplo o TIMEX 2048) é capaz de misturar texto e gráficos ao mesmo tempo no *écran*. O *écran* produzido pelo Spectrum não ocupa todo o espaço disponível no visor. Possui 24 linhas por 32 colunas de caracteres de texto e encontra-se dividido em duas zonas, como se vê na figura abaixo.

É a zona superior de 22 linhas por 32 colunas que habitualmente podemos utilizar para escrever ou desenhar. As duas linhas inferiores estão reservadas para a introdução de comandos (por exemplo LIST, RUN...), de linhas de programa ou para informações por parte do sistema operativo (mensagens de erro, por exemplo). Durante a execução de programas, são ainda estas linhas as habitualmente utilizadas para a introdução de dados por parte do utilizador.

O computador usa números para se referir a cada uma das linhas e a cada uma das colunas: as linhas vão de 0 a 21 e as colunas de 0 a 31, sendo a posição do carácter da linha e coluna

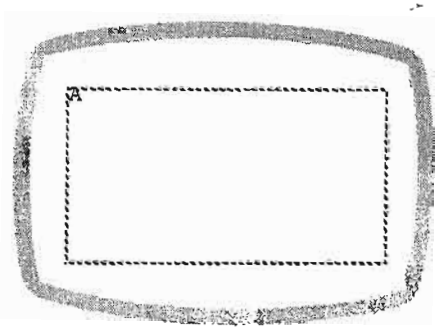
O *écran* de Texto e Gráficos do ZX Spectrum



O no canto superior esquerdo do *écran*. Vamos ordenar-lhe que escreva um "A" nessa posição.

```
PRINT AT 0, 0; "A"
```

O resultado será como este:



Lembre-se que o *écran* do Spectrum não ocupa todo o visor, só podendo escrever e desenhar numa zona aqui representada por um rectângulo a tracejado.

Poderíamos também ter atingido o mesmo resultado fazendo simplesmente PRINT "A", desde que antes tivesse havido um CLS. Contudo, se quisermos escrever no canto inferior direito, teremos mesmo de empregar PRINT AT, e a instrução a dar será:

```
PRINT AT 21, 31; "Z"
```

No caso de usar uma televisão a preto e branco, a cada cor corresponde uma tonalidade de cinzento, sendo o código 0 o mais escuro (preto) e o código 7 o mais claro (branco).

Podemos escrever os caracteres a cores. O Spectrum pode utilizar 6 cores, mais branco e preto, tendo cada uma um código:

```
0 PRETO
1 AZUL
2 ENCARNADO
3 MAGENTA (ROXO)
4 VERDE
5 CYAN (AZUL-CLARO)
6 AMARELO
7 BRANCO
```

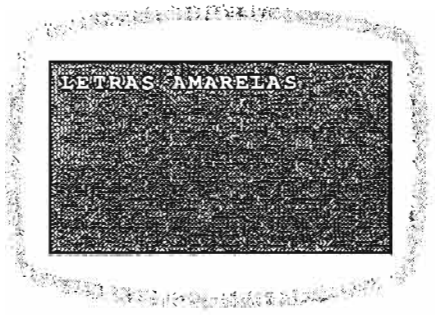
Estas cores podem ser indistintamente utilizadas para a cor da tinta das letras ou para a de fundo (ou papel). A instrução que atribui uma cor à tinta é INK seguida do código da cor, e a que atribui cor ao papel é PAPER, também seguida do código da cor. Vejamos um exemplo para escrever letras amarelas em fundo azul:

O CLS é necessário para o *écran* passar a azul.

```
10 INK 6 .
20 PAPER 1
30 CLS
40 PRINT "LETRAS AMARELAS"
```

Se se fizer RUN, aparecerá um *écran* semelhante ao da figura seguinte, em que as letras amarelas sobressairão do papel azul.

Note que a seguir à instrução PAPER na linha 20 surge CLS que serve para "limpar" o *écran* com esta nova cor.



É neste bordo que aparecem as características faixas horizontais aquando a leitura e gravação de programas em cassette.

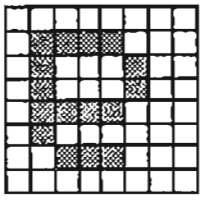
Mas repare que há uma zona do *écran* que se mantém branca. É a moldura ou bordo, na qual não se pode escrever, que para ser alterada para a cor do papel necessita de mais uma instrução, BORDER, seguida do código da cor.

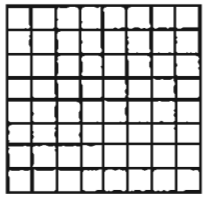
```
35 BORDER 1
```

Se se correr o programa novamente, todo o *écran* ficará azul.

PIXEL = Picture Element "elemento da figura".

Cada quadradinho do *écran*, a que corresponde um carácter, tem 8 pixels de lado, constituindo assim uma quadrícula de 8*8, como se vê na figura seguinte, em que se dá o a como exemplo.



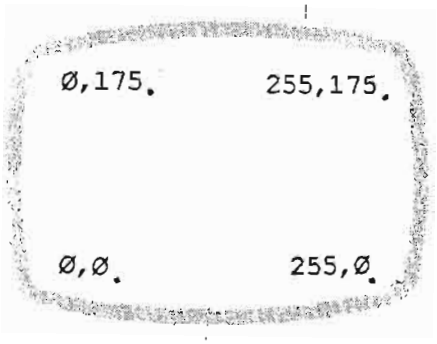


Podemos assim calcular que o *écran* disponível para gráficos de alta resolução tem 32*8 por 22*8, ou seja 256 por 176 pixels. Para mostrar um desses pixels, utilizamos a instrução PLOT, seguida dos valores nos eixos horizontal e vertical.

```
PLOT 0,0
```

Quando um **pixel** aparece destacado do fundo, costuma-se dizer que está "ligado".

Esta instrução faz aparecer um pixel, de cor preta, no canto inferior esquerdo, a que correspondem as coordenadas 00 — é portanto um sistema diferente do que é utilizado na numeração das linhas. Na figura da página seguinte mostram-se os valores de PLOT para os quatro cantos do *écran*. Repare-se que primeiro está o valor do eixo horizontal e depois o do eixo vertical.



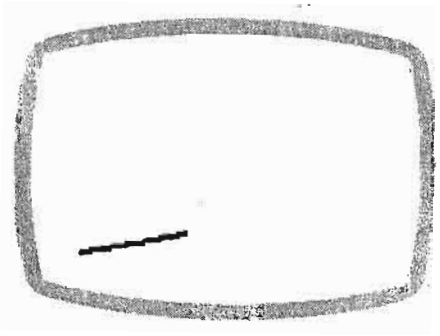
Outra instrução gráfica, que permite desenhar círculos, é CIRCLE. Para utilizarmos esta instrução precisamos de definir as coordenadas do centro e indicar o valor do raio, em **pixels**. Assim para traçar um círculo ao centro do *écran*, com 50 **pixels** de raio, faremos:

```
CIRCLE 127,88,5Ø
```

Existe ainda uma terceira instrução gráfica: DRAW. A sua utilização não é difícil, embora seja um pouco mais complexa. Esta instrução serve para traçar linhas, sejam elas segmentos de recta ou arcos. Vejamos um exemplo:

```
1Ø CLS
2Ø DRAW 5Ø, 25
```

O *écran* abaixo mostra o resultado da execução deste programa.



Os valores indicados em DRAW dizem quanto se deve avançar para a direita e quanto se deve deslocar para cima.

A instrução DRAW traçou um segmento de recta entre dois **pixels**: a origem é o último **pixel** desenhado (obtido por DRAW, PLOT ou CIRCLE, ou o **pixel** de coordenadas ØØ no caso de não ter havido nenhuma destas instruções anteriormente ou de ter havido um CLS); o destino é o **pixel** cujas coordenadas se obtêm adicionando os valores especificados com DRAW, aos valores das coordenadas do último **pixel**.

No caso anterior, as coordenadas do último **pixel** eram 0,0. A linha foi portanto traçada entre o **pixel** 0,0 e o **pixel** 0+50,0+25. Como faria se quisesse agora traçar uma linha entre o **pixel** 50,25 e o **pixel** 100,100?

```
3Ø DRAW 5Ø,75
```

Os valores são 50,75 porque é o que se deve adicionar a 50,25 para se obter 100,100.

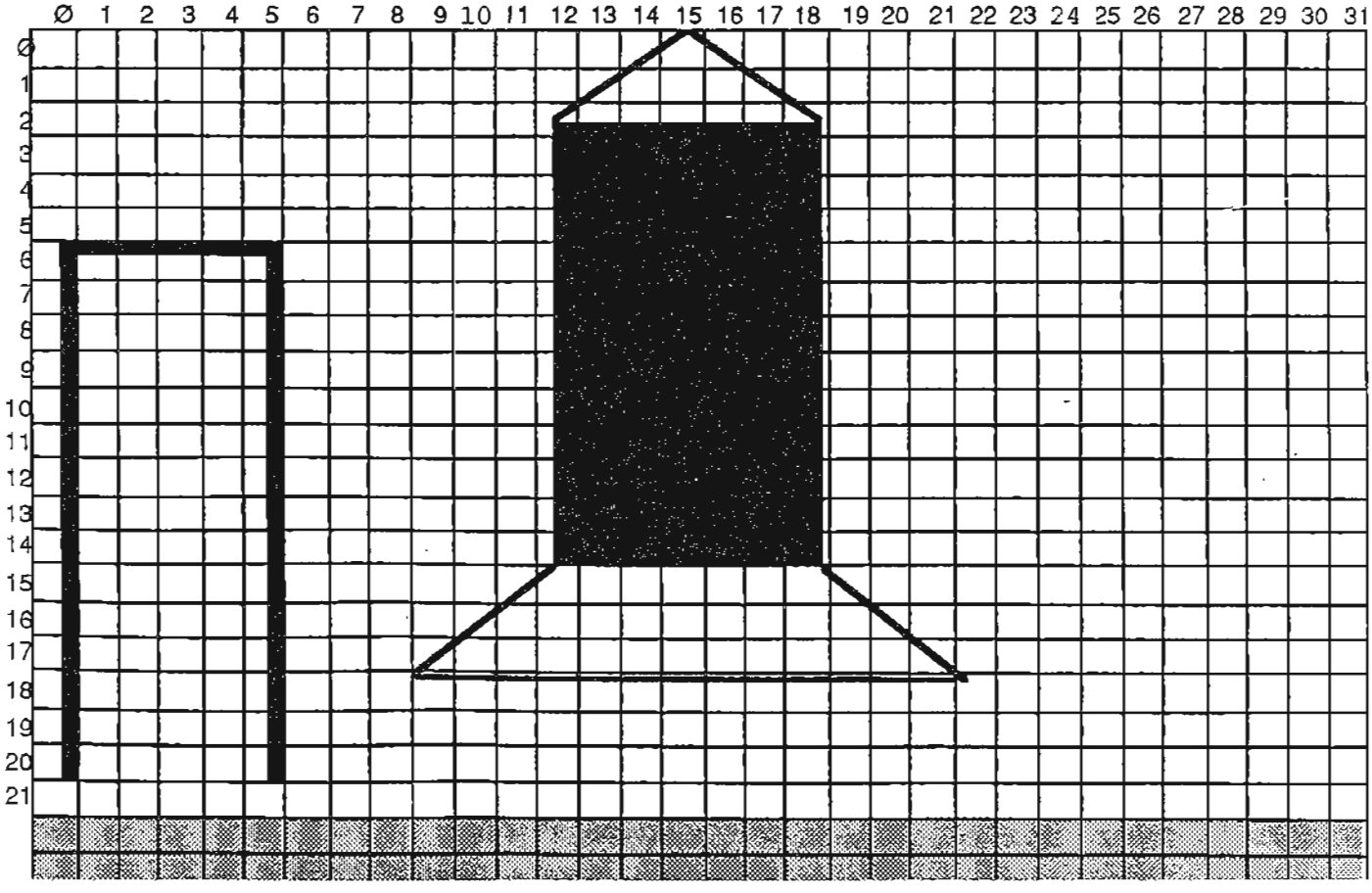
Este exemplo era fácil porque o pixel de origem já tinha sido determinado pela instrução anterior. Mas e se quisermos agora traçar uma linha entre o pixel 200,10 e o pixel 250,150? A instrução DRAW anterior deixou o último pixel traçado em 100,100. Mas é necessário que o último pixel seja 200,10. Para isso temos de recorrer à instrução PLOT, para marcar a nova origem. Depois de o pixel de origem estar determinado, basta executar o DRAW que traça o segmento até 250,150. Como a origem está em 200,10 será: 250-200, 150-10. Ou seja

```
40 PLOT 200,10
50 DRAW 50,140
```

A instrução DRAW traça ainda arcos. Para traçar um semicírculo, faça correr o seguinte programa:

```
10 PLOT 100, 100
20 DRAW 0, 50, 3.14
```

(o terceiro valor acrescentado força o traçado do arco com valor em radianos idêntico a esse valor).



Não se esqueça de fazer NEW.

Vamos agora desenhar uma nave espacial, como a da figura da página anterior. Em primeiro lugar, marcamos o pixel que se encontra no topo da nave:

```
10 PLOT 125, 175
```

Agora podemos traçar um triângulo representando o "nariz" do foguete:

```
20 DRAW -25, -20
30 DRAW 50,0
40 DRAW -25,20
```

Os valores negativos em DRAW indicam que a deslocação é para a esquerda e para baixo.

Segue-se o "corpo" da nave:

```
50 PLOT 100,155
60 DRAW 0, -100
70 DRAW 50,0
80 DRAW 0, 100
```

De seguida desenhamos a "cauda":

```
90 PLOT 100, 55
100 DRAW -25, -25
110 DRAW 100,0
120 DRAW -25, 25
```

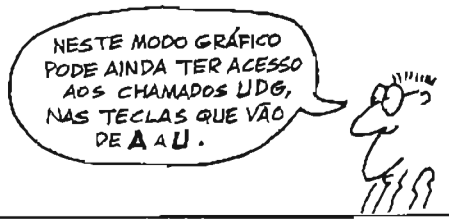
Podemos agora preencher o "corpo" da nave:

O texto das linhas 140 e 150 está "avançado" para a direita para facilitar a identificação e a leitura do ciclo FOR NEXT pelo programador.

```
130 FOR I = 55 TO 155
140 PLOT 100, I
150 DRAW 50,0
160 NEXT I
```

Se corrermos novamente o programa poderemos aperceber-nos da rapidez com que o Spectrum e compatíveis desenham no *écran*.

Existe ainda a possibilidade de se desenhar com caracteres semi-gráficos, sendo para isso necessário passar o cursor para o modo de texto-gráfico. Consegue-se isso colocando o computador em modo *caps shift* e a tecla 9. A partir de agora, todas as teclas numéricas passam a semi-gráficas. A título de exemplo vamos desenhar uma "torre de lançamento" para o foguete.



Os caracteres semi-gráficos utilizados obtêm-se com as teclas dos números 5,3 e 7 em modo gráfico (cursor G).

```
170 PRINT AT 6,0; "▛▀▀▀▀▀▀▀▀▀▀▀▜"
180 FOR I = 7 TO 20
190   PRINT AT I,0; "▌           ▐"
200 NEXT I
```

tente responder

1. Traçar uma diagonal com asteriscos:

```
10 FOR I = 0 TO 21
20   PRINT AT I,I; "*"
30 NEXT I
```

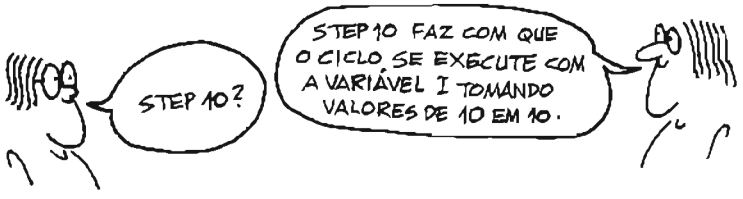
2. Vamos preencher o *écran* com caracteres de várias cores. Para esse efeito utilizaremos dois ciclos aninhados um no outro (repare que o ciclo da variável J está completamente dentro do ciclo da variável I). O efeito é que para I = 1, J varia entre 0 (o código da cor preta) e 7 (o código da cor branca), depois para o I = 2, para o I = 3 e assim sucessivamente, preenchendo todo o *écran*...

Podemos ainda variar o brilho da cor utilizada, com a instrução BRIGHT. Veja no manual do seu computador.

```
10 FOR I = 1 TO 88
20  FOR J = 0 TO 7
30   PRINT PAPER J; "  ";
40  NEXT J
50 NEXT I
```

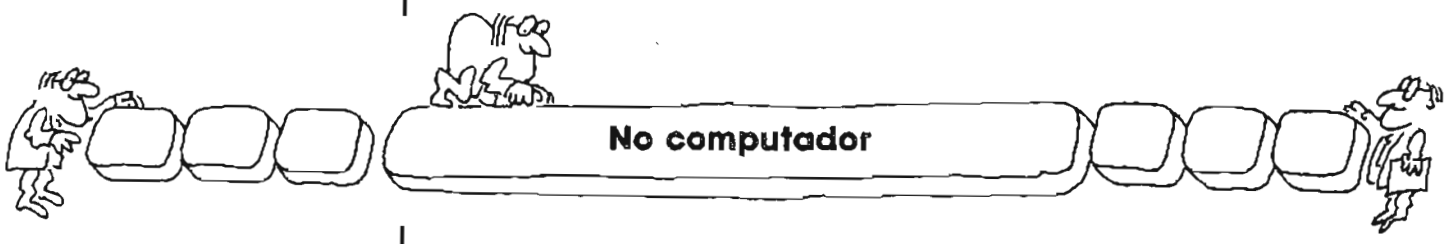
3. O programa que se segue traça uma série de círculos concêntricos:

```
10 FOR I = 1 TO 100 STEP 10
20   CIRCLE 127,88,I
30 NEXT I
```



4. Continue o desenho da nave, fazendo com que saiam "labaredas" da sua "caúda".

5. Desenhe uma nuvem utilizando a instrução DRAW, socorrendo-se da capacidade desta instrução em desenhar arcos.



No computador

1. Comece por experimentar escrever o seu primeiro nome em várias zonas, por exemplo nos quatro cantos e ao centro.

2. Com as instruções CIRCLE e PRINT AT, procure desenhar o mostrador de um relógio. Comece por colocar os números e depois circunscrever o círculo.

3. Vamos fazer um programa para alterar as cores do bordo muito rapidamente:

```
10 FOR I = 0 TO 7
20  BORDER I
30 NEXT I
```

PAUSE obriga o computador a esperar um pouco antes de continuar.

Se quiser ver as cores a mudarem mais lentamente, experimente utilizar uma instrução PAUSE com vários valores até encontrar o mais adequado (por exemplo PAUSE 50).

4. A boa resolução do *écran* deste computador permite obter alguns padrões curiosos, através da pequena variação de um ou dois pixels entre o traçado de linhas. Vejamos um exemplo:

```
10 FOR I =  TO 87
20  PLOT 0,0
30  DRAW 255,I*2
40 NEXT I
50 FOR I = 88 TO 175
60  PLOT 255,175
70  DRAW -255,176-2*I
80 NEXT I
```

5. Desenhe as suas iniciais no *écran*.

Lembre-se que cada carácter é o resultado de um quadriculado de 8 × 8 **pixels.** No caso do espaço, este quadriculado não tem qualquer **pixel** "ligado".

6. Vamos construir um programa que faça aparecer aleatoriamente quadrados (ou antes, o carácter espaço) de várias cores no *écran*. Para isso temos de lembrar que existe uma função que permite calcular números pseudo-aleatórios entre dados limites: a função RND.

```
10 PRINT AT INT(RND*22), INT(RND*32); PAPER INT(RND*8); " "
20 GOTO 10
```

(este programa apenas poderá ser interrompido por Caps Shift + Break, já que se trata de um ciclo infinito).

Não se esqueça do espaço entre as aspas da linha 10.

7. Podemos também fazer algumas pequenas animações. Para dar a ideia de movimento a um asterisco, ao longo de uma linha, teremos de fazer um ciclo que o "move" e que, em cada novo movimento, apaga a posição anterior.

Esta técnica de animação é comum a todos os programas que parecem exibir movimento de objectos no *écran*.

```
10 FOR I=0 TO 31
20  PRINT AT 12,I;"*"
30  PAUSE 5
40  PRINT AT 12,I;" "
50 NEXT
```

Capítulo 25

TOME A INICIATIVA E PROGRAME

Agora que já aprendeu muitas instruções da linguagem BASIC é altura de as juntar de forma a criar um programa original. Este seu programa poderá ser um teste de computador, um jogo, uma visualização gráfica, um desenho animado (em movimento) ou qualquer outra ideia que você possa ter.

Antes de começar a planear o seu programa, experimente mais algumas instruções BASIC aceites pela maior parte dos computadores. Verifique com alguém conhecedor ou consulte o manual do seu computador para se assegurar que as pode usar.

Este programa, usa a instrução TAB significando a linha 20 que a palavra ADEUS é avançada dez espaços.

TAB é uma instrução BASIC que significa "indentar".

```
10 PRINT "OLA"
20 PRINT TAB(10) "ADEUS"
RUN
OLA
          ADEUS
```

ADEUS é impresso a partir do décimo espaço.

Veja o seguinte programa e o respectivo *output*:

```
10 FOR X 1 TO 3
20   PRINT TAB(X) "OLA"
30 NEXT X
40 FOR X = 3 TO 1 STEP -1
50   PRINT TAB(X) "ADEUS"
60 NEXT X
RUN
OLA
  OLA
    OLA
    ADEUS
  ADEUS
ADEUS
```

TAB(X) torna-se TAB(1), TAB(2), TAB(3) OLÁ é impresso no primeiro, segundo e terceiro espaços.

ADEUS é impresso no terceiro, segundo e primeiro espaços.

LEFT$ lê-se "Left-string".

A instrução LEFT$ designa caracteres situados a partir da extremidade esquerda de uma cadeia, tal como exemplificado no programa e no *output* seguintes:

O computador Atari não utiliza as declarações LEFT$, RIGHT$ e MID$.

```
10 A$ = "SERRA"
20 PRINT LEFT$(A$,3)
RUN
SER
```

Imprime os **3 caracteres da esquerda** de A$

E até se pode divertir com estas instruções:

```
10 A$ = "SERRA"
20 FOR X = 1 TO 5
30   PRINT LEFT$(A$,X)
40 NEXT X
RUN
S
SE
SER
SERR
SERRA
```

Imprime os X caracteres da esquerda de A$ (X varia de 1 a 5)

Outra instrução semelhante é RIGHT$ que designa os caracteres situados a partir da extremidade direita da cadeia; a exemplificação consta do programa e do *output* seguintes:

```
10 A$ = "SUPER!"
20 PRINT RIGHT$(A$,4)[27] <-- Imprime os 4 caracteres da direita de A$
RUN
PER!
```

A declaração MID$ designa os caracteres no meio de uma cadeia; o programa e o *output* seguintes são elucidativos:

```
10 A$ = "COMPUTADOR"
20 PRINT MID$(A$,3,2)[28]
RUN
MP
```

MID$(A$,3,2)
Cadeia — Começa na 3.ª posição — Imprime 2 caracteres

Eis uma maneira de imprimir lentamente uma letra de cada vez no *écran*. Experimente o programa num computador para ver como o *output* é impresso.[29]

```
10 A$ = "COMPUTADOR"
20 FOR X = 1 TO 8
30   PRINT MID$(A$, X,1)
40    FOR P = 1 TO 50: NEXT P
50 NEXT X
```

A linha 30 imprime um carácter de A$, começando na posição X (X varia de 1 a 8). A linha 40 funciona como uma pausa.

Comece a trabalhar

Apresentam-se aqui algumas sugestões e linhas de orientação para escrever um programa original e que se podem usar com qualquer das suas ideias. É importante que o planeamento de um programa seja feito com cuidado; deve-se sempre saber o que se quer que um programa faça e qual o tipo de *output* desejado. É útil começar por fazer um esboço das partes principais do programa e imaginar como programar cada uma delas. Só depois se deve escrever no teclado do computador as respectivas instruções.

Não é necessário escrever todo o seu programa num papel. Verifique no manual a possibilidade de gravar o programa numa cassette ou numa disquete. Você pode dar um nome ao seu jogo.

Teste de computador. No capítulo 20 escreveu um pequeno teste de computador. Vai agora imaginar um outro que deverá ser maior e mais interessante. Comece por arranjar um tema para o teste e escreva cinco ou dez perguntas sobre o assunto. Tente pensar em diferentes «recompensas» para cada pergunta. Por exemplo, se o utilizador obtém uma resposta certa, em vez de apenas imprimir a mensagem ACERTOU, poderá tentar o seguinte:

```
110 FOR X = 1 TO 100
120   PRINT "ACERTOU!"
130 NEXT X
```

```
110 FOR X = 1 TO 12
120   PRINT TAB(X) "ACERTOU!"
130 NEXT X
```

Pode ainda imaginar uma visualização gráfica como uma cara a sorrir ou carrancuda conforme a resposta estiver certa ou errada.

Jogo de improviso: No capítulo 19 já escreveu um jogo de improviso. Desta vez vai fazer uma pequena história com poucas frases. Sublinhe as palavras-chave da sua história. Uma frase poderia ser, por exemplo:
O José viu dois macacos a baloiçarem num trapézio.

Depois invente uma pergunta para cada palavra sublinhada. Para que o utilizador possa responder às perguntas utilize instruções INPUT.

```
10 PRINT "DIGA O NOME DE UM AMIGO"
20 INPUT A$
```

Depois use instruções PRINT para imprimir a frase base utilizando variáveis em vez das palavras sublinhadas.

Jogo de Computador: É possível programar muitos jogos para um computador. No capítulo 21 viu-se um jogo cujo objectivo era adivinhar números. E se escrevesse um jogo para adivinhar letras? Veja o programa seguinte que utiliza a declaração MID$ para indicar ao computador a escolha de uma letra ao acaso da cadeia:[30]

```
10 A$ = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
20 N = INT(RND(1)*26) + 1
30 B$ = MID$(A$, N, 1) <---- Escolhe 1 carácter da posição N da cadeia.
```

A letra escolhida pelo computador e que está em B$ é comparada com o palpite do utilizador considerando o computador a letra A como "baixa" e a Z como "alta". Assim A é "menor" que Z, M "maior" que D, P "menor" que Q, etc. A programação do jogo é semelhante à do de adivinhar números.

Deve dar um nome ao jogo.

Vamos agora ver um outro jogo divertido. O computador escolhe um número ao acaso (pode ser um número grande) e fá-lo aparecer no *écran* durante 1 segundo ou menos (para regular esse tempo use um ciclo FOR--NEXT). Depois limpe o *écran* e pergunte ao utilizador que número era. Se acertou, o utilizador marca 1 ponto: se se enganou, é o computador que marca 1 ponto. A parte do programa que faz a pontuação é a seguinte:

C é a pontuação do computador e U a do utilizador.

```
110 PRINT "NÃO, O NÚMERO ERA"; N
120 C = C + 1 <---------- O computador acertou e aumenta 1 à
130 GOTO 10                sua pontuação.
140 PRINT "ACERTOU!"
150 U = U + 1 <---------- O utilizador acertou e aumenta 1 à sua
160 GOTO 10                pontuação.
```

No final do programa a pontuação final aparece impressa:

```
200 PRINT "A SUA PONTUACAO E "; U
210 PRINT "A MINHA PONTUACAO E "; C
```

Há muitos outros tipos de jogos que podem ser programados, muitos dos quais se baseiam em números aleatórios (ao acaso).

Desenho: Para programar um desenho comece por desenhar a figura numa folha de gráficos indicando, se o seu computador tiver cor, aquela que quer que cada parte apresente. Também pode utilizar números aleatórios para escolher cores ou localizações no *écran*.

Desenhos animados: No capítulo 24, os utilizadores dos computadores Atari e ZX Spectrum programaram um "quadrado" ou "caixa" a mover-se através do *écran*. Para "mover" um desenho maior, deve desenhá-lo, apagá-lo, desenhá-lo num espaço ao lado do inicial, voltar a apagá-lo, etc.

FICHA DE CONTROLE

Escolha a melhor resposta para cada pergunta:

1. O comando BASIC que indica ao computador para processar um programa é:

   a. NEW
   b. GOTO
   c. RUN

2. O comando BASIC que apaga o programa da RAM — a memória de acesso aleatório — é:

   a. NEW
   b. LIST
   c. RUN

3. O comando BASIC que faz aparecer um programa no *écran* é:

   a. NEW
   b. GOTO
   c. LIST

4. Qual é o *output* da seguinte instrução:
   10 PRINT "6 + 3 = "; 6 + 3

   a. 9
   b. 6 + 3
   c. 6 + 3 = 9

5. 10 PRINT "OLA"
   20 GOTO 10
   Este programa é um exemplo de:

   a. um ciclo FOR-NEXT
   b. um ciclo infinito

6. Escolha a instrução correcta:

   a. LET 3 = N
   b. LET N$ = SAPATO
   c. LET N = 3

7. Se A = 10 e B = 2 diga qual o *output* do computador para: PRINT A*B, A/B

   a. A*B A/B
   b. 20 5
   c. A*B = 20

8. A instrução que indica ao computador que deve parar, imprimir um ponto de interrogação no *écran* e esperar pela resposta do utilizador é:

   a. INPUT
   b. GOTO
   c. IF-THEN

9. Qual é o *output* do seguinte programa:
   10 C = 6
   20 IF C = 2 THEN PRINT "VERMELHO"

   a. VERMELHO
   b. não há *output*

10. Limpar o *écran* apaga o programa da memória do computador.

    a. verdadeiro
    b. falso

11. Qual das seguintes instruções é utilizada para dizer ao computador exactamente quantas repetições num ciclo deve fazer?

    a. FOR-NEXT
    b. IF-THEN
    c. READ-DATA

12. A aptidão de um microcomputador para fazer figuras e gráficos é conhecida como:

    a. programação
    b. gráficos por computador
    c. visualização de texto

NOTAS DO REVISOR TÉCNICO

[1] Hoje em dia temos já um aperfeiçoamento, a VLSI (Integração em Muito Grande Escala) que permite centenas de milhar de circuitos.

[2] Além disso o exemplo empregue não resume as múltiplas utilizações do computador na educação, não sendo o que melhor demonstra as suas potencialidades.

[3] Ou correspondente CAT (Computer Assisted Instruction, EUA) ou CAL (Computer Assisted Learning, Grã-Bretanha) ou EAO (Enseignement Assisté par Ordinateur, França).

[4] O PET é um modelo da Commodore, lançado em 1977 que foi o primeiro microcomputador compacto, com *écran*, teclado e gravador de cassettes numa única estrutura.

[5] Por ser completamente diferente, aconselha-se o leitor não iniciado a consultar o manual destes computadores.

[6] No ZX Spectrum a primeira tecla premida gera uma instrução, mas pode seguir o exemplo fazendo inicialmente LET (letra L).

[7] O ZX Spectrum nem aceitará a entrada, mostrando um ponto de interrogação, sendo a única solução apagar toda a linha — por exemplo fazendo SHIFT 1 (EDIT).

[8] Em qualquer dos casos não esquecer de premir RETURN ou ENTER...

[9] No ZX Spectrum não é possível fazer este erro, uma vez que PRINT resulta de tecla P e não é alterável.

[10] No ZX Spectrum deverá substituir END por STOP.

[11] Seria talvez mais adequado utilizar-se paralelogramas  para as instruções indicadas, uma vez que estes simbolizam as entradas e saídas (input + output) dos programas.

[12] Como já saberá, só alguns computadores usam directamente a acentuação. Por isso é provável que no seu tenha que escrever NAO em vez de NÃO...

[13] Também conhecidos em programação por "separadores".

[14] No ZX Spectrum, como só há 2 zonas, a terceira passa à primeira da linha seguinte e assim sucessivamente.

[15] Não se esqueça que em muitos computadores não poderá acentuar directamente as palavras ou empregar o ç pelo que terá de os substituir. Ex: ATÉ LOGO por ATE LOGO.

[16] Recorde que a maior parte dos computadores domésticos não permitem acentuação ou ç, directamente: terá nesse caso de escrever DIRECCOES.

[17] No ZX Spectrum, ao fim do preenchimento do *écran* aparecerá a mensagem "SCROLL?". O programa continuará se carregar em ENTER.

[18] No ZX Spectrum o computador escreverá a mensagem "SCROLL" ao fim do preenchimento de cada *écran*.

[19] Embora em Portugal se use a vírgula para assinalar que um número não é inteiro, os computadores usam o ponto.

[20] O ZX Spectrum é um dos que aceita apenas uma letra seguida de cifrão ($).

[21] Nem todos os computadores funcionam assim. Veja mais à frente no texto.

[22] Nalguns computadores, de cada vez que se fizer RUN, a sequência é a mesma. Se se pretender que seja diferente pode tentar acrescentar a instrução RANDOMIZE, numa das primeiras linhas do programa, antes da instrução RND. Por exemplo 5 RANDOMIZE.

[23] No caso do ZX Spectrum nem se utiliza, bastando LET X = RND. Em compensação no Spectrum a instrução LET tem de estar presente, enquanto noutros computadores não.

[24] No ZX Spectrum continua a ser obrigatório pôr as cadeias alfanuméricas entre aspas, mesmo nas instruções DATA.

[25] Esta secção foi escrita propositadamente para este livro, tendo em conta a sua grande utilização doméstica em Portugal. Neste caso o texto é da inteira responsabilidade do revisor técnico.

[26] O ZX Spectrum não tem estas instruções, mas pode-se atingir o mesmo efeito escrevendo:
20 PRINT AS(TO3).

[27] Para o ZX Spectrum: 30 PRINT AS (LEN AS—3 TO).

[28] Para o ZX Spectrum: 20 PRINT AS (3 TO 4).

[29] Para o ZX Spectrum: 30 PRINT AS (X).

[30] Para o ZX Spectrum:
```
20 LET N = INT(RND*26) + 1
30 LET B1 = A$(N)
```

[31] Para ser mais preciso 1K = 1024 bytes e 16K = 16 384 bytes.

GLOSSÁRIO

ábaco (abacus)
antigo dispositivo de cálculo constituído por uma armação de madeira atravessada por fios ou arames onde se movem contas.

Aiken, Howard
engenheiro americano que projectou o primeiro computador electromecânico, o Mark I.

alfanumérico
v. cadeia

algoritmo
plano em que se indica passo a passo a forma de resolver um problema.

analista de sistemas
pessoa que analisa uma situação e projecta um sistema informático para a solucionar.

Babbage, Charles
cientista britânico que desenvolveu e parcialmente construiu a Máquina Analítica, precursora dos modernos computadores digitais.

banco de dados
conjunto de informações directamente exploráveis pelo computador geralmente armazenadas em dispositivos de entrada/saída tais como bandas e discos.

BASIC
acrónimo de **B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode: é a linguagem de computador utilizada na maior parte dos microcomputadores e também em muitos minicomputadores e grandes sistemas.

bibliotecário de software *(software librarian)*
numa empresa ou organização, a pessoa encarregada do arquivo de uma grande colecção de *software*.

bit
dígito binário, 0 ou 1, usado para representar o estado "desligado" ou "ligado" (*off/on*) de um circuito eléctrico; é a mais pequena unidade de informação digital que um computador processa.

bug
qualquer engano ou erro num programa de computador.

byte
unidade geralmente composta por oito *bits;* cada carácter do teclado do computador é representado por um *byte.*

cabeça de escrita/leitura *(read/write head)*
dispositivo localizado no interior de uma unidade de discos que lê a informação existente no disco, também podendo "escrever" e apagar informação.

cadeia ou cadeia alfanumérica *(string)*
qualquer combinação de letras, números e símbolos (excepto aspas).

calculador graduado *(stepped reckoner)*
máquina de calcular inventada em 1694 por Gottfried Leibnitz e que podia somar, subtrair, multiplicar e dividir.

caneta luminosa *(light pen)*
dispositivo de entrada, parecido com uma caneta, que reage à intensidade da luz num *écran* de vídeo.

carácter *(character)*
todo e qualquer número, letra ou símbolo representado num teclado de computador.

cartões perfurados *(punched cards)*
cartões contendo orifícios ou perfurações que representam dados ou programas.

ciclo infinito *(infinite loop)*
conjunto de instruções de um programa que se repetem continuamente.

circuito *(circuit)*
via através da qual passa a electricidade.

circuito integrado *(integrated circuit — IC)*
circuito eléctrico construído por muitos transistores que conduz a corrente eléctrica muito mais rapidamente do que um só transistor; os circuitos integrados foram pela primeira vez usados na terceira geração de computadores.

COBOL
acrónimo de **C**ommon **B**usiness **O**riented **L**anguage; é uma linguagem de computador utilizada sobretudo em aplicações de gestão.

Código Universal de Produtos *(Universal Product Code-UPC)*
série de barras e de números impressos nas embalagens de bens (geralmente alimentares) e que são lidas por um leitor óptico *(scanner)*.

comando *(command)*
palavra (em linguagem BASIC) que geralmente indica ao computador para executar algo com um programa: RUN é um exemplo de um comando.

computador *(computer)*
máquina projectada para receber, armazenar, processar e fornecer informação processada.

computador analógico *(analog computer)*
computador que mede condições que se alteram de uma maneira contínua como a temperatura e a pressão e as converte em quantidade.

computador digital *(digital computer)*
computador multi-uso que aceita como *inputs* letras, números e símbolos e os converte em dígitos binários, forma sob a qual vão ser armazenados e processados.

concatenar *(concatenate)*
palavra que significa "ligar, pôr junto, encadear"; na programação podem-se concatenar cadeias alfanuméricas.

cursor *(cursor)*
símbolo que aparece no *écran* vídeo e que indica onde vai ser impresso o carácter seguinte.

dados *(data)*
informação; também se refere à informação que vai ser introduzida no computador *(inputs)* ou à que resulta do processamento do programa *(outputs)*.

declaração (statement)
palavra em BASIC que quando usada num programa indica ao computador o que deve fazer. Instrução.

disco *(disk)*
dispositivo circular e plano cuja superfície magnética é capaz de armazenar programas de computador; alguns discos são flexíveis (ver *discos flexíveis*) e outros são rígidos (ver *discos rígidos*).

disco flexível ou disquete *(floppy disk)*
dispositivo de armazenamento de dados sob forma magnética, geralmente usado em microcomputadores.

disco rígido ou duro *(hard disk)*
dispositivo de armazenamento de dados sob forma magnética que permite armazenar grandes quantidades de informação e recuperá-la rapidamente.

dígito *(digit)*
número simples de 0 a 9.

dispositivo de entrada *(input device)*
parte de um computador que introduz a informação na respectiva memória; também conhecida por *unidade de entrada*.

dispositivo de entrada/saída (dispositivo I/O) *(input/output device)*
dispositivo que funciona como unidade de entrada e como unidade de saída.

dispositivo de saída *(output device)*
parte do computador que exibe, imprime ou regista os resultados; também conhecida por *unidade de saída*.

dummy *(falso)*
parte dos dados de um programa sem qualquer significado especial.

Eckert, J. Presper
engenheiro americano que, juntamente com John Mauchly, projectou os computadores ENIAC e UNIVAC.

écran vídeo *(vídeo screen)*
dispositivo de saída utilizado para visualizar informação; também é conhecido por *tubo de raios catódicos (TRC ou CRT),* monitor ou vídeo display unit (VDU).

electromecânico *(electromechanical)*
construído quer com partes eléctricas quer com partes mecânicas (móveis).

ENIAC
acrónimo de **E**lectronic **N**umerical **I**ntegrator **A**nd **C**alculator, o primeiro computador electrónico projectado em 1946 por Eckert e Mauchly.

engenheiro de computadores *(computer engineer)*
pessoa que projecta os circuitos usados nos computadores.

entrada *(input)*
informação fornecida ao computador.

equipamento periférico *(peripheral equipment)*
hardware (como um gravador de fita ou impressora), ligado ao computador principal.

erro de sintaxe *(syntax error)*
erro nas regras estruturais de uma linguagem de programação; geralmente é um erro de ortografia ou pontuação.

fita de cassete *(cassette tape)*
fita de material magnético onde se registam programas de computador sob a forma de impulsos magnéticos; juntamente com um gravador é utilizada como dispositivo de entrada/saída de microcomputadores.

fita magnética em bobines *(reel-to-reel tape)*
tipo de dispositivos de armazenagem I/O usado principalmente nos grandes sistemas e nos minicomputadores.

fluxograma *(flowchart)*
diágrama representativo da sequência de passos de um algoritmo ou programa.

folha de cálculo *(electronic spreadsheet)*
programa que permite ao utilizador resolver, por exemplo, problemas de orçamentação através da visualização de uma tabela constituída por linhas e colunas.

FORTRAN
acrónimo de **For**mula **Tran**slator, uma linguagem de programação utilizada geralmente em aplicações científicas e de engenharia.

gestão de base de dados *(data-base management)*
programa que permite ao utilizador armazenar, classificar e recuperar informação.

gestor de processamento de dados *(data processing manager)*
pessoa que tem a seu cargo todo o *hardware* e *software* de uma empresa.

gráficos *(graphics)*
output constituído por desenhos e figuras, que não por texto.

gravador de fita *(tape recorder)*
dispositivo de entrada/saída (I/O) geralmente usado em microcomputadores.

hard copy
saída de computador em suporte de papel, originado numa impressora; também conhecido por *printout* ou impressão.

hardware
elementos físicos que constituem o computador; maquinaria, aparelhagem e acessórios que o constituem.

Hollerith, Herman
engenheiro americano autor do projecto da Máquina de Tabulação utilizada para classificar os dados do recenseamento da população dos Estados Unidos de 1890 e que trabalhava com cartões perfurados.

I/O
de *input/output;* referente a entrada/saída.

impressora *(printer)*
dispositivo de saída que imprime *outputs* em papel.

indicador *(prompt)*
nalguns computadores o símbolo que aparece no *écran* é usado para indicar que estão prontos a aceitar *inputs.*

integração em grande escala *(large scale integration—LSI)*
processo de colocar milhares de circuitos integrados numa única pastilha *(chip)* de silício.

Jacquard, Joseph
tecelão francês que em 1801 projectou um tear baseado na utilização de cartões perfurados para registar os modelos dos desenhos feitos nos seus tecidos.

K
deriva de *quilo* ou 1000; abreviatura utilizada para simplificar as referências à dimensão da memória de um computador: 16 K bytes corresponderão a 16 000 bytes[31].

Leibnitz, Gottfried
cientista alemão que, em 1694, construiu o Calculador Graduado um dispositivo de cálculo.

leitor de cartões perfurados *(card reader)*
dispositivo de entrada que transfere dados de cartões perfurados para a memória do computador.

leitor óptico *(optical mark reader ou scanner)*
dispositivo de entrada que reconhece as formas dos caracteres ou marcas escritas num papel.

Logo
termo grego que significa "pensamento"; linguagem projectada para permitir às crianças comunicarem e interactuarem com um computador de uma maneira mais natural.

Lovelace, Ada Augusta
matemática inglesa; auxiliou Charles Babbage no projecto da sua Máquina Analítica e escreveu sobre o seu trabalho.

mainframe computer *(grande sistema)*
grande sistema de computadores que pode executar muitos trabalhos em simultâneo.

Máquina Analítica *(Analytical Engine)*
dispositivo mecânico de cálculo, projectado (mas nunca totalmente construído) pelo cientista inglês Charles Babbage em 1835; foi o antepassado dos modernos computadores digitais.

Máquina Aritmética *(Arithmetic Machine)*
máquina de calcular inventada por Blaise Pascal em 1642 e que podia fazer adições e subtracções.

máquina de perfuração *(keypunch machine)*
máquina utilizada para fazer perfurações em cartões que vão ser usados como *input* do computador.

Máquina de Tabulação *(Tabulating Machine)*
máquina projectada em 1887 por Herman Hollerith, contava e registava dados a partir de cartões perfurados e foi utilizada para determinar os resultados do recenseamento da população americana em 1890.

Mark I
primeiro computador electromecânico; foi projectado por Howard Aiken em 1944.

Mauchly, John
engenheiro americano que juntamente com J. Presper Eckert projectou os computadores ENIAC e UNIVAC.

memória *(memory)*
parte de um sistema de computadores onde a informação é armazenada.

Memória de acesso aleatório (RAM — *Random Acess Memory*)
memória temporária do computador onde são armazenados os dados e programas que vão sendo introduzidos como *inputs*.

Memória apenas de leitura (ROM — *Read Only Memory*)
memória permanente do computador.

microcomputador *(microcomputer)*
dispositivo de entrada utilizado geralmente para ler o Código Universal de Produtos (ou Código de Barras) nos produtos embalados; leitor óptico.

minicomputador *(minicomputer)*
computador de médio porte que já pode executar várias tarefas em simultâneo.

nanosegundo *(nanosecond)*
unidade utilizada para medir a velocidade ou os tempos de cálculo de um computador e correspondente a um milésimo milionésimo do segundo (1/1 000 000 000s)

Napier, John
matemático escocês que em 1612 inventou um dispositivo de cálculo conhecido por «Ossos de Napier».

notação científica *(scientific notation)*
maneira abreviada de escrever números muito grandes ou muito pequenos; num computador 6.987 E + 09 representa 6 987 000 000.

operador de computadores *(computer operator)*
pessoa que faz funcionar um computador, sobretudo num grande sistema, nas operações de arranque, paragem e manutenção corrente.

operador de registo de dados *(data entry person)*
pessoa que opera uma máquina perfuradora ou introduz dados no teclado de um terminal.

Ossos de Napier *(Napier's Bones)*
conjunto de varas numeradas, desenvolvido por John Napier e utilizado para efectuar cálculos.

palavra de senha *(password)*
palavra ou número de código que deve ser introduzido no computador antes do utilizador poder ter acesso a um programa ou a outros dados.

PASCAL
linguagem de computador cujo nome evoca o de Blaise Pascal; utilizada tanto em microcomputadores como em sistemas maiores.

Pascal, Blaise
matemático francês que em 1642 inventou a Máquina Aritmética.

pastilha *(chip)*
ver *pastilha de circuitos integrados.*

pastilha de circuitos integrados *(integrated circuit chip — ICC)*
pastilha de silício de dimensões muito pequenas contendo milhares de circuitos integrados e utilizada na quarta geração de computadores.

printout
output de uma impressora, também conhecido por *had copy* ou impressão.

processamento de texto *(word processing)*
capacidade do computador que permite ao utilizador mover ou alterar palavras, frases e parágrafos sem as voltar a escrever integralmente. Programas que permitem a manipulação de textos.

programa *(program)*
conjunto de instruções escritas em linguagem de computador e que lhe indicam o que ele deve executar.

programador *(programmer)*
pessoa que escreve programas de computador.

reconhecimento de caracteres em tinta magnética *(magnetic ink character recognition—MICR)*
tipo de entrada em que um leitor MICR interpreta os caracteres impressos com uma tinta magnética especial.

RAM — ver Memória de Acesso Aleatório

robot
computador projectado para desempenhar tarefas mecânicas.

ROM — ver Memória apenas de leitura

saída *(output)*
resultado do processamento de um programa.

scanner
dispositivo de entrada utilizado geralmente para ler o Código Universal de Produtos (ou Código de Barras) nos produtos embalados, leitor óptico.

silício (silicon)
elemento vulgar na crosta da Terra e de que são feitas as pastilhas *(chips) de computador.*

sintetizador de voz (voice synthesizer)
dispositivo de saída que traduz o *output* do computador em sons semelhantes aos da voz humana.

sistema de numeração binária *(binary number system)*
sistema de numeração que utiliza apenas dois dígitos, o 0 e o 1.

software
programas para um computador; geralmente o termo refere-se aos programas armazenados em dispositivos tais como discos ou fitas.

software integrado *(integrated software)*
combinação de dois ou mais programas de *software* num programa maior.

tear de cartões perfurados *(punched card loom)*
máquina de tecelagem aperfeiçoada por Joseph Jacquard em 1801; usava cartões perfurados para registar os modelos dos desenhos feitos nos tecidos.

teclado *(keyboard)*
dispositivo de entrada utilizado na escrita de dados.

técnico de computadores *(computer technician)*
pessoa que repara computadores e que também pode trabalhar com um engenheiro no projecto de computadores.

terminal ***(terminal)***
dispositivo de entrada/saída (I/O); dispõe de um teclado para entrada de dados e de um *écran* vídeo ou de uma impressora para saída; pode estar ligado a um grande computador central, a um minicomputador e, mais recentemente, a um microcomputador.

texto ***(text)***
tudo o que se refere a números, letras ou símbolos indicados no teclado; termo usado em oposição a gráficos.

traçador de gráficos ***(plotter)***
dispositivo de saída que desenha figuras ou gráficos num papel, com grande precisão.

transistor
dispositivo utilizado para conduzir a corrente eléctrica na segunda geração de computadores.

tubo de raios catódicos ***(cathode ray tub — CRT)***
ver *écran vídeo*.

tubo de vácuo ***(vacuum tube)***
dispositivo usado para a condução da corrente eléctrica na primeira geração de computadores.

Unidade Aritmética e Lógica ***(arithmetic/logic unit)***
uma das duas partes principais da Unidade Central de Processamento, onde se fazem cálculos e comparam números.

unidade de banda ***(tape drive)***
dispositivo de entrada/saída (I/O) que lê e armazena informações em banda magnética.

Unidade Central de Processamento ***(central processing unit—CPU)***
parte de um computador que processa a informação; é constituído por duas partes principais, a Unidade de Controlo e a Unidade Aritmética e Lógica.

Unidade de Controlo ***(control unit)***
uma das partes principais da Unidade Central de Processamento; dirige o fluxo de informação através das várias partes do computador.

unidade de discos ***(disk drive)***
dispositivo de entrada/saída que carrega um programa ou os dados armazenados num disco para um computador.

UNIVAC
acrónimo de **Univ**ersal **A**utomatic **C**omputer; projectado por Eckert e Mauchly em 1951; foi o primeiro computador comercial.

utilizador ***(user)***
pessoa que usa um programa de computador.

variável ***(variable)***
quantidade que pode variar de valor enquanto o programa corre; há dois tipos de variáveis: as numéricas e as de cadeias alfanuméricas.

von Neumann, John
matemático americano que introduziu na primeira geração de computadores a ideia de aí armazenar um programa.

ÍNDICE REMISSIVO

Resumo das Instruções e Comandos da Linguagem BASIC

Instrução	Exemplo	Resultado
	INSTRUÇÕES DE ENTRADA/SAÍDA	
PRINT	PRINT "OLA"	Faz aparecer a palavra OLA no CRT.
	PRINT 5 + 3	Faz aparecer 8 o valor de 5 + 3
END	END	Dá instruções ao computador para terminar o programa.
LET	LET X = 3	Atribui o valor 3 à variável X
INPUT	INPUT N	Exibe um ? e espera que o utilizador escreva um valor para N.
READ-DATA	READ B:DATA 34	O computador lê (READ) o valor da declaração DATA e armazena-o como B
STOP	STOP	Semelhante a END.
	INSTRUÇÕES DE CONTROLO DE FLUXO	
GOTO	GOTO 30	Dirige o computador para a linha 30.
IF-THEN	IF X = 5 THEN GOTO 80	Se o valor de X for igual a 5 o computador saltará para a linha 80. Se não for igual a 5, executar-se-á a linha seguinte do programa.
FOR-NEXT	FOR C = 1 TO 10:NEXT C	Dá instruções ao computador para repetir as declarações entre as instruções FOR e NEXT, 10 vezes.
	INSTRUÇÕES DE FORMATOS	
vírgula (,)	PRINT 1, 2, 3, 4	Indica ao computador para imprimir os números 1, 2, 3 e 4 tabulados ao longo do *écran* em colunas.
ponto e vírgula (;)	PRINT "HI ";N$	Indica ao computador para imprimir o valor de N$ a seguir à palavra HI na mesma linha do *écran*.
	SÍMBOLOS ARITMÉTICOS E LÓGICOS	
+	PRINT 6 + 2	Indica ao computador para adicionar.
–	PRINT 23-12	Indica ao computador para subtrair.
*	PRINT 4 * 5	Indica ao computador para multiplicar.
/	PRINT 30/6	Indica ao computador para dividir.
<	X < Y	X é menor que Y.
>	F > G	F é maior que G.
=	T = U	T é igual a U.
< >	E < > D	E ou é maior ou menor que D (E é diferente de D).
< =	A < = B	A é menor ou igual a B.
> =	Z > = Y	Z é maior ou igual a Y.
	FUNÇÕES	
RND	N = RND(25)	Imprime um número aleatório entre 1 e 25.
INT	N = INT(RND(10*25) + 1	Imprime um número aleatório entre 1 e 25.
	COMANDOS DE SISTEMA	
RUN	RUN	Indica ao computador para executar um programa.
LIST	LIST	Faz a listagem das declarações que constituem o programa.
NEW	NEW	Apaga da memória o programa que lá estiver nesse momento.

A cultura tecnológica desempenha, na sociedade dos nossos dias, um papel cada vez mais fundamental. O presente *Manual de Introdução aos Computadores*, de Ellen Richman, incide sobre uma das áreas mais relevantes dessa cultura, que é a que diz respeito à utilização de computadores e, em particular, de computadores pessoais.

A autora apresenta de modo claro e sucinto as principais aplicações dos computadores (processamento de texto, base de dados, folhas de cálculo, etc.), bem como alguns aspectos de programação elementar numa linguagem adequada a pequenos programas simples: o BASIC.

A clareza e o rigor do texto permitem a utilização deste livro por crianças, jovens e adultos que pretendam adquirir a "literacia computacional" indispensável ao uso da informática.