MICRO COMPUTADOR

CURSO BASICO

Sumário

VOLUME 1

Chips & bytes

Conexões

Fundamentos

Hardware

Os precursores

Perspectivas

VOLUME 1

Por dentro do hardware

Programação Basic

Software

Sumário

VOLUME 2

Chips & bytes

Conexões

Fundamentos

Hardware

Os precursores

Perspectivas

Por dentro do hardware

VOLUME 2

Programação BASIC

Software

Som e luz

Domine seu micro

O hardware, conjunto de componentes físicos do computador, não funciona sem o software conveniente. Conheça o significado deste termo fundamental e avalie os softwares existentes.

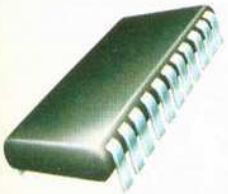
Software é a parte invisível de um sistema de computador. Sem ele, o computador é uma simples massa inerte de componentes eletrônicos e nada pode realizar.

Com uma olhada nos chips de silício de um computador verifica-se que consistem em milhares — talvez milhões — de interruptores eletrônicos. Assim como o interruptor de uma lâmpada comum não pode ligar e desligar sozinho, os interruptores do computador precisam ser acionados, mas não simultaneamente: cada um deles deve ser ligado ou desligado separadamente e na seqüência exata em relação às centenas de outros interruptores. É esta a função do software.

O software consiste em instruções que fazem o computador funcionar. As instruções são dadas sob a forma de números que, apresentados à CPU (o coração do computador; ver p. 4), ligam e desligam os interruptores internos, fazendo-os executar operações específicas. O computador só "compreende" esses números se estiverem na forma binária (mediante a conversão nos dígitos 1 e 0, como é explicado no capítulo Bits e bytes, p. 32).

Esses dois dígitos, inteligíveis para o computador (no sentido de que o instruem para a execução de determinadas tarefas), são o resultado final de uma longa série de operações, originadas na mente do programador. Podem-se encontrar os programas ("programa" é o termo que designa qualquer unidade de software) sob formas diversas. A única afirmação precisa que se pode fazer sobre todos os programas é a de que devem resultar em algo compreensível pelo computador.

Veja um exemplo: um engenheiro de tráfego deseja controlar uma série de semáforos, utilizando um computador. Para isso, o computador que fará o controle vai precisar de um programa que o instrua a induzir a seqüência correta de mudança de cores (todos os faróis verdes ao mesmo tempo seria inconcebível). Antes de desenvolver esse software, o en-



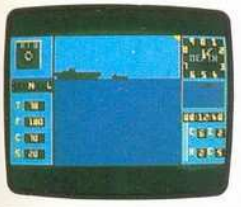
ROM

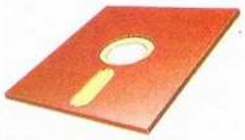
A ROM (Read Only Memory) é um dos principais dispositivos de memória do computador que permite o armazenamento permanente de programas. A maioria dos microcomputadores é equipada com chips ROM que contêm a linguagem BASIC de programação. Outras ROMs podem ser compradas para aprimorar o desempenho do computador pelo acréscimo de mais uma linguagem, ou para transformá-lo em um processador de palavras.

CASSETE

Geralmente, o software é fornecido em fitas cassete idênticas às de gravação de som e é nessa forma que em geral os programas de jogos se encontram à venda.

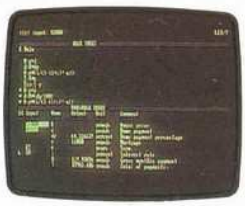
O computador é alimentado mediante a conjugação do equipamento a um gravador cassete comum que "toca" a fita do programa. A fita pára após o carregamento e geralmente o computador não precisa voltar a "consultá-lo".





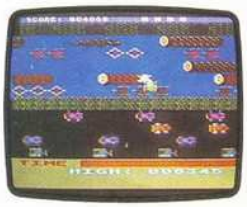
DISCO FLEXÍVEL

Pode-se armazenar software gravando-o em um disco feito com um filme magnético. A gravação é efetuada em trilhas como as faixas de LP comum, através de uma cabeça magnética de leitura/gravação, que também lê o programa, quando necessário. Os discos possuem grande capacidade de armazenamento e operação em alta velocidade.



CARTUCHO

É basicamente uma ROM embalada em proteção adequada. Alguns microcomputadores têm encaixes de fácil acesso, nos quais os cartuchos são inseridos. Os softwares geralmente vendidos em cartucho são linguagens de programação (como o BASIC) ou sofisticados videogames.



Para fazer funcionar seu computador é necessário "alimentá-lo" com software (uma série de dados eletrônicos). Nos dispositivos aqui mostrados, esses dados são armazenados. Representam os quatro métodos mais comuns de fornecimento de software, tendo cada um vantagens específicas. Os softwares são feitos "sob medida" para cada marca de computador — um programa desenvolvido para determinada marca não funcionará necessariamente em outra.

genheiro deve decidir com cuidado que tarefas o computador executará.

Normalmente, as instruções seriam formuladas em sentenças da linguagem comum (ex.: em determinado momento, quero que o sinal n.º 1 mude para amarelo, mantendo ao mesmo tempo o vermelho; em seguida, que as luzes vermelha e amarela se apaguem e a verde acenda). Mas, assim estruturadas, essas instruções são incompreensíveis ao computador; portanto, precisam ser convertidas em um programa. Para tal, deve ser usada uma linguagem de programação, como BASIC, por exemplo.

Esse tipo de linguagem permite que pensamentos logicamente ordenados em linguagem comum sejam reescritos de forma inteligível para o interpretador BASIC. Interpretador BASIC é, na verdade, um programa que converte o programa original (escrito em BASIC) em uma forma compreensível à CPU — Unidade Central de Processamento. Nesta forma, o software é chamado "linguagem de máquina" ou "código de máquina".

O software que você comprar para seu computador estará quase sempre em linguagem de máquina e armazenado de modo imediatamente acessível ao computador. Às vezes, o software é armazenado na memória ROM (interna) do computador. Mais freqüentemente, é fornecido em cassete ou disco flexível. Esses elementos não são exatamente software: apenas os meios pelos quais o software é fornecido. Para uso do computador, é preciso transferir o software do cassete (ou do disco flexível, ou da ROM) para o computador. Só após o carregamento (nome do processo de transferência de software) o programa poderá ser executado.

Compra de software

Com algum dinheiro disponível, talvez você pense: "Acho que vou comprar um carro". Mas ninguém pensaria: "Acho que vou comprar um equipamento", pois as perguntas decorrentes seriam: "Que tipo de equipamento? Que trabalho espero que realize?"

Ocorre o mesmo com o software. O computador não funciona sozinho: é o software com que você o equipa que o transforma em videogame, em máquina de escrever automatizada ou em máquina de contabilidade. Assim, a primeira decisão a tomar refere-se ao que você deseja de seu computador.

Feito isso, o problema seguinte é definir-se sobre o software adequado a suas necessidades. No processo de busca do software mais conveniente, você irá naturalmente aperfeiçoando sua escolha, ao analisar suas necessidades concretas. Se o ponto de partida for como divertir as crianças no domingo à tarde, o passo seguinte será descobrir que tipos de programa podem fazer isso: os jogos abrangem desde videogames do tipo "caça aos inimigos" até complexas e motivadoras simulações de fantasia. Feita a escolha, a etapa seguinte é verificar se o software está ou não disponível para o seu equipamento.

Uma vez que diferenças entre computadores não são apenas superficiais (cada um tem seus próprios componentes eletrônicos internos e requer desenvolvimento especial de software), dificilmente os modelos são compatíveis: um programa feito para o

FIRMWARE

A origem do termo hardware (equipamento físico) é evidente — é a parte física e eletrônica do computador: conexões para fornecimento de energia, teclado, chips de silício etc. Já o termo software (suporte lógico) tem origem em sua natureza intangível, pois consiste simplesmente em uma série de instruções.

No início da era da computação — há uns trinta anos —, o software era codificado e armazenado em fitas de papel perfuradas, semelhantes às de telex. Mais tarde, cassetes e discos magnéticos as substituíram. Na década de 70, inventou-se uma nova técnica para armazenamento de software em ROMs: os chips. Os chips ROM têm instruções software neles incorporadas no processo de fabricação. Essa combinação de software "intangível" e hardware "concreto" chama-se firmware (suporte lógico inalterável).

Manipulação de palavras

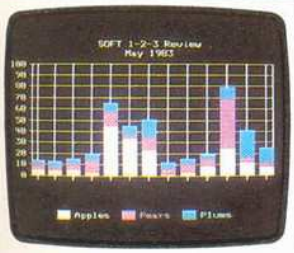
Processador de texto

Com um software processador de palavras, seu computador ultrapassa a máquina de escrever. Mesmo boas datilógrafas cometem erros, mas com um processador de palavras as cartas serão sempre impressas com perfeição e grande produtividade. O teclado do computador substitui o da máquina de escrever, e a tela substitui o papel. As palavras escritas aparecem imediatamente na tela, assim como na folha datilografada. Mas aí terminam as semelhanças e começam as vantagens do computador.

Podem-se corrigir erros, na tela, instantaneamente, reescrever ou suprimir palavras e até eliminar parágrafos. Além disso, se a reordenação de sentenças permitir melhor expressão de seu pensamento, você pode fazê-lo na própria tela. As palavras ou sentenças que deseja reordenar na "página" são suprimidas temporariamente (o programa processador de palavras elimina-as da tela, armazenando-as na memória do computador) e você pode depois inseri-las onde quiser.

Quando o documento atingir exatamente a forma desejada, poderá ser impresso usando-se a impressora do computador ou ser armazenado em cassete ou disco flexível, para utilização posterior.

Balanço



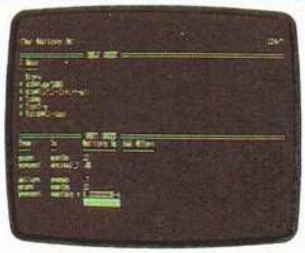
Pacote de contabilidade

Uma vez que os computadores efetuam operações matemáticas, é natural haver muitos programas que auxiliem na área financeira. Numerosos são os softwares desse tipo: da manutenção automática de livros fiscais ao serviço completo de contabilidade. Esses programas precisam ter acesso a grande quantidade de informações e armazenar grande quantidade de registros. Por isso, exigem, geralmente, ao menos uma unidade de discos flexíveis para atender às exigências de armazenamento em larga escala.

Programas contábeis geralmente funcionam por meio de uma série de perguntas (apresentadas na tela do computador) e de respostas (fornecidas pelo operador). O programa utiliza as informações registradas pelo operador, realiza os cálculos necessários e armazena os resultados no disco ou os imprime na forma adequada.

Esses programas incluem emissão automática de faturas, reorganização de estoque, controle de livros fiscais e acompanhamento de trabalhos em andamento. Esse tipo de software, embora caro, pode significar bom investimento, pois diminui despesas com funcionários e fornece resultados mais rapidamente.

Arquivamento



Bancos de dados

Os computadores podem acessar arquivos de informações muito mais depressa do que o fazem as pessoas; quanto maior a quantidade de informações desejada, maior a utilidade do computador. Em sua forma mais simples (e barata), o banco de dados é um pouco mais do que uma caderneta de endereços computadorizada, em que são localizados nomes, endereços e números telefônicos. Programas de bancos de dados mais sofisticados e caros executam operações muito mais complexas.

Para se ter idéia do potencial de um banco de dados, pense em um botânico que compila dados para um livro sobre cogumelos exóticos e venenosos: ele fará arquivos extensos sobre as diferentes espécies e habitats, extrairá notas de obras de consulta amplamente variadas e organizará listas infindáveis de especialistas.

Antes de existirem os computadores, essas informações tinham de ser escritas em cartões e arquivadas. Com um computador e um programa de banco de dados as informações são armazenadas na memória e o botânico obtém respostas imediatas a suas questões: se necessitar de uma lista em ordem alfabética de todos os livros que incluem as palavras "venenoso", "cogumelos" ou "fungos", o banco de dados pode fornecê-la.

Os bancos de dados são, em geral, bastante caros e normalmente encontram-se disponíveis apenas em discos flexíveis.

Manipulação de números

"Folha eletrônica"

A folha eletrônica (spreadsheet) é a solução fornecida pelo computador para questões hipotéticas, antes resolvidas com calculadora e muito papel. O comércio de qualquer produto abrange muitas variáveis e a mudança de uma delas geralmente afeta as demais.

Observe problemas que um dono de sorveteria gostaria de resolver: "Teríamos mais lucro com a redução do preço do sorvete mantendo o mesmo número de balconistas ou devemos aumentar o preço e empregar mais dois?"

A escolha provavelmente influenciará toda a transação comercial — reduções nos preços podem representar aumento nas vendas mas diminuição nos lucros. A folha eletrônica vai proporcionar resultados imediatos para esse tipo de problema.

Todos os números essenciais a serem manipulados estão dispostos em um quadriculado de linhas e colunas e a relação entre cada coluna e linha é especificada (por exemplo, cada número na coluna C é obtido subtraindo-se o número da coluna A do número da coluna B). Uma vez agrupados todos os dados reais e hipotéticos, a alteração de qualquer número isolado pode ser efetuada e seu efeito sobre os outros números é imediatamente visível.

Folhas eletrônicas são utilizadas por empresários para planejar custos, ou engenheiros e cientistas, que precisam utilizar dados numéricos muito variáveis e exigem geralmente unidades de disco e impressoras.

Lazer

Jogos

Os computadores não servem apenas para processamento de números e palavras. Podem fornecer horas de lazer, se usados com um dos muitos programas de jogos existentes (xadrez, gamão e diversões eletrônicas, como o "pouso lunar" e simulações de vôo). Há também jogos de aventuras extraordinariamente complexos que duram dias e mesmo semanas (ver p. 22). Muitos deles não são apenas divertidos: têm considerável valor educacional.

Esses jogos exigem atenção intensa e o fornecimento contínuo de dados pelo jogador. O fornecimento dá-se geralmente por meio do teclado; uma tecla pode ser usada para bombardear um "laser" ou um "missil" ou para controlar o movimento de alguma coisa na tela. O número de teclas usadas varia em função do jogo e dos controles exigidos pelo programa.

Uma alternativa comum ao fornecimento de dados por teclado são os comandos joystick. São ligados ao computador, funcionam de modo semelhante ao dos manches de avião, permitem maior controle e tornam os jogos ainda mais divertidos.

Software pronto para uso

A maioria dos softwares é vendida em lojas na forma de disco flexível ou em cassete, acompanhados de um manual. Apple Writer ou Editex são programas padrões processadores de palavras. Consistem em um único disco flexível e em um manual que inclui explicações completas e concisas, de modo que iniciantes podem usá-lo imediatamente. Os padrões de documentação são muito variados. Alguns softwares vêm com manuais tão incompletos e mal escritos que pode ser difícil, e mesmo impossível, usá-los.

CP 500 não funcionará no Unitron AP II (a não ser que se fabrique algum modelo especial deste). Por esta razão, você deve sempre comprar software especialmente produzido para seu equipamento.

Mesmo sabendo disso, você ainda não tem condições para efetuar a compra. O próximo ponto a ser considerado são as limitações físicas do seu equipamento. Verifique qual é a sua capacidade de memória: se for de 16 K RAM, veja se o jogo escolhido necessita de memória suplementar. De modo geral, jogos mais interessantes e sofisticados exigem programas mais longos; sendo assim, será necessária maior quantidade de memória. Lembre-se também de que o software é encontrado em grande variedade de formas físicas (ver a página anterior). Se um programa é vendido apenas em disco flexível e você tiver só um gravador cassete, não poderá usá-lo antes de comprar uma unidade de discos de alto custo. Alguns softwares (especialmente os jogos) exigem outros suplementos, como controles joystick. Provavelmente uma impressora será dispensável para os jogos, ao contrário do que ocorre com os softwares comerciais que, com freqüência, necessitam de uma para registrar os resultados.

Tipos de software

De certa forma, jogos são uma questão à parte: afinal, sua função é de entretenimento. A maioria dos softwares é desenvolvida para execução de um tipo específico de trabalho, do modo mais fácil e rápido. São inúmeras as maneiras de os programadores de software conseguirem aumentar a rentabilidade e eficiência deste. Consideremos a datilógrafa ineficiente, cujo trabalho não satisfaz ao chefe. Com um microcomputador e um tipo de software chamado processador de palavras, o computador substitui a máquina de escrever, realizando as correções na tela. Depois de corrigida, a página inteira pode ser transcrita em papel automaticamente, ao toque de uma simples tecla, evitando, desse modo, erros e perda de tempo.

Outra tarefa cansativa que se presta à computadorização é a administração financeira. Muitas atividades que exigiam grande número de funcionários — como o balanço de contas de uma empresa e o cálculo de salários — são agora executadas por um software especialmente desenvolvido para esse fim. Os próprios programas já são bastante especializados; é improvável que uma só unidade de software possa atender a todas essas necessidades. Os tipos de software incluem: programas de cálculo de salários e emissão de holerites, programas de controle de estoque, para saber quais as mercadorias vendidas ou os gastos efetuados (em alguns casos, o programa automaticamente encomenda novos estoques), e até programas que auxiliam no cálculo de tamanhos e tipos mais econômicos de papel para impressão de livros e revistas.

Outra tarefa extremamente bem executada por computadores é o arquivamento e classificação de informações. Este tipo de programa é denominado banco de dados. Os bancos de dados podem substituir unidades inteiras de arquivo de referência cruzada, possibilitando rápido acesso a qualquer informação.

A última classe importante de software é a conhecida como folha eletrônica. Permite o preparo e a correção ilimitados de complexos orçamentos e previsões financeiras, dispensa o uso de enormes quantidades de papel e substitui a conhecida calculadora.

Todos os tipos de software aqui descritos são vendidos "prontos para uso": quem os desenvolveu originariamente visualizou uma série de soluções específicas para problemas específicos. Todavia, poderá ocorrer que não se encontre à venda o software para instruir seu computador exatamente na execução do que você precisa. O que fazer, neste caso? Uma solução, embora dispendiosa, é contratar um programador que desenvolva um programa de acordo com suas necessidades. Outra, é aprender você mesmo a desenvolver programas. Munido de uma linguagem como o BASIC é possível criar programas que instruam seu computador na execução das mais surpreendentes operações. E o único gasto consiste no tempo usado para desenvolver o programa.

Jogos e brincadeiras

Castelos e dragões, mercado financeiro e lutas espaciais, divertimento e educação — tudo está nos jogos de computador.

O computador pode ser muito divertido: o mundo dos jogos eletrônicos é um caleidoscópio fascinante de brincadeiras, enigmas e novos desafios.

Os populares fliperamas estão sendo substituídos por baratos e potentes microcomputadores, que proporcionam muita variedade e emoção até mesmo enquanto você está aprendendo a jogar. Esses jogos nos micros conseguiram atrair mais pessoas para a computação do que qualquer software de contabilidade. Você não deve pensar que, por isso, não está utilizando bem a máquina. Os jogos estão aí porque divertem.

Aventura

Nem todos os jogos precisam ter som e cor para que sejam apaixonantes. Toda uma nova gama deles tem aparecido com a difusão do micro; jogos que estimulam a imaginação com palavras, da mesma forma que livros e revistas sempre o fizeram.

Esses são os chamados "jogos de aventura", e os primeiros programas foram feitos por programadores para brincar com seu grande computador no tempo livre de que dispunham. A idéia é que o programa cria um mundo que o jogador explora dirigindo seu *alter ego*; mas a direção é dada com palavras digitadas no teclado e não com joysticks.

O pequeno caractere da tela é movido quando se digitam instruções tais como "norte" e "acima", e o computador mostra uma descrição do ambiente e de qualquer objeto próximo. O jogador pode "pegar", "largar", "virar" ou "quebrar" os objetos, ou pode experimentar alguma ação que lhe pareça interessante. O único limite é a imaginação.

Um programa de computador pode ser um labirinto de salas e porões, misturado com baús e soldados alemães, como no conhecido jogo Castelo Wolfenstein, ou pode ser uma nave espacial extraterrestre abandonada, ou mesmo uma casa de campo onde se procura solucionar um crime.

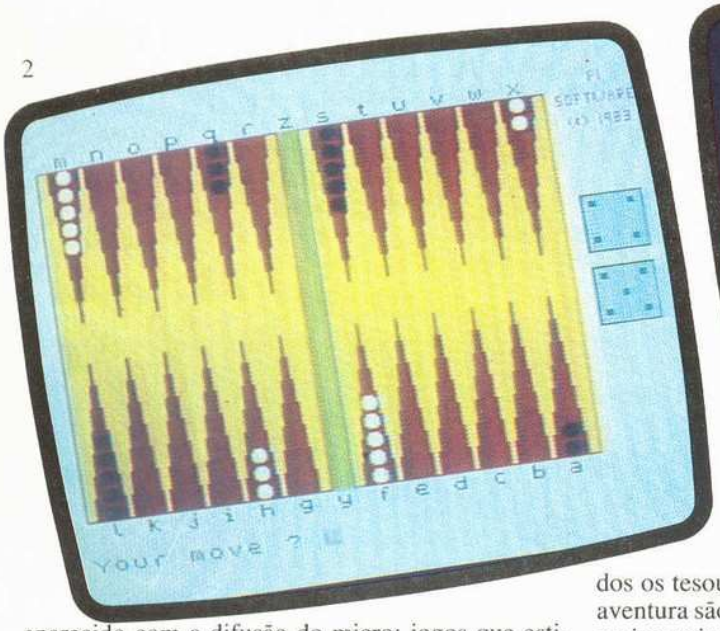
Seja qual for o cenário, o jogador tem de explorar, achar objetos valiosos ou tesouros e resolver enigmas intelectuais. A contagem total só aparece quando todos os problemas forem solucionados e todos os tesouros encontrados. Os melhores jogos de aventura são como um bom romance e pode-se levar muito mais tempo para completá-los do que para ler um livro.

Mesa e tabuleiro

Muito naturalmente, os jogos preferidos de mesa e de tabuleiro foram transferidos para os microcomputadores, tão logo a tecnologia pôde fazê-lo. Não é necessário encontrar um oponente, pois o computador o substitui, e, se você errar uma jogada, pode corrigi-la, sem que o computador o acuse de trapacear. A máquina pode também aperfeiçoar o jogo do parceiro, apontando e corrigindo erros que você cometa.

1

2



3

4

6

O xadrez por computador alcançou um bom padrão e os gráficos do micro podem agora produzir traçados completos e detalhados do tabuleiro com um movimento suave da peça. Diferentes estratégias e aberturas foram exaustivamente analisadas e um programa de computador poderia facilmente ser campeão mundial. Os jogos de gamão, bridge e outros têm estado disponíveis em várias máquinas domésticas. Os computadores são fortes adversários no tabuleiro, na mesa ou na tela.

Jogando e aprendendo

Muitas empresas de programação se especializam em jogos educativos, com o computador ministrando testes e quebra-cabeças e aplaudindo quando a resposta está certa. Ou, às vezes, um jogo parece simplesmente um jogo mas contém informações educativas, faz somas, soletra e até impõe a lei da oferta e da procura.

Favorito entre os jogos é aquele em que as crianças dirigem, sobre uma folha de papel, um robô que segura uma caneta e é conhecido como "tartaruga", devido à sua forma. Assim, as crianças se divertem desenhando, enquanto aprendem geometria.

Fliperamas

Os fliperamas, jogos de muita ação e movimento, já atraíram bilhões de fichas para suas máquinas. Nos microcomputadores, você pode divertir-se com os mais conhecidos: ataques de invasores, sapos que saltam, aventuras de Tarzan e gorilas gigantescos.

Mas as empresas de programação têm suas próprias idéias e criaram jogos que concorrem com os fliperamas em emoção e em gráficos espetaculares. Com um micro há maior possibilidade de escolha de jogos rápidos e emocionantes, todos à sua disposição, sem a presença voraz da máquina caça-níqueis. Esses jogos testam os microcomputadores — e os programadores — até o limite de sua possibilidade.

Estratégia grandiosa

Os jogos que requerem concentração e planejamento, as táticas e a tenacidade de um general comandando uma batalha também se transferiram para o microcomputador.

O jogador pode ser qualquer general, em qualquer guerra, deslocando exércitos e tentando descobrir os planos do adversário, o computador. Este se apresenta na sua melhor forma, atuando como árbitro e controlador do tabuleiro, à medida que decifra regras densamente escritas e pedaços de papelão que se perdem com facilidade — o que fez dos jogos de guerra de tabuleiro os preferidos apenas de uma minoria.

O jogador pode ser também o rei de um pequeno país, planejando um meio de administrar a colheita e o tesouro e, ao mesmo tempo, mantendo os trabalhadores felizes e alimentados e os ladrões longe de seus domínios.

Da mesma forma, o jogador pode estar encarregado do fornecimento de energia de um país, avaliando os preços de minérios, do petróleo e da energia nuclear, contra seus perigos e efeitos a longo prazo. O computador pode ajudá-lo a avaliar e aperfeiçoar os seus planos de conquistar o mundo.

Voando alto

Os programas de jogos podem colocá-lo na cabina de um avião veloz, lendo os instrumentos e manejando os controles para decolar e aterrissar com perfeição, numa variedade de simulações de aeroportos verdadeiros; os jogos podem transformá-lo no piloto de uma missão espacial, que avista a Terra através de vigias; ou podem torná-lo um magnata como Rockefeller nas bolsas de valores internacionais, formando e destruindo empresas poderosas.

Precisão é tudo nos jogos de simulação. Siga as regras do mundo real e o jogo lhe mostrará o que realmente aconteceria. Mas, se cometer um erro, você não ficará preso nos destroços nem precisará saltar de uma janela de Wall Street. Os computadores são mais complacentes do que o mundo real!

5

1 Magia: Entre num mundo misterioso. Seus companheiros podem ajudar. Mas quem escolher? Um guerreiro, uma jovem ou um cientista?

2 Gamão: O seu adversário neste jogo é apoiado por uma força formidável — a lógica implacável do computador.

3 ABC Dragon: Um jogo educativo para crianças.

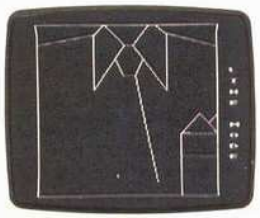
4 Zaxxon: Um dos primeiros jogos do tipo fliperama. A tela é o limite do piloto, à medida que ele avança dando voltas entre os mísseis e lutando para atingir seu objetivo.

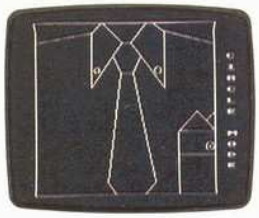
5 Frente Oriental: Talvez você possa ter sucesso na Rússia, onde Hitler falhou.

6 Simulação do vôo de uma aeronave. Você se esquiva ou sofre uma colisão.

O micro: um artista

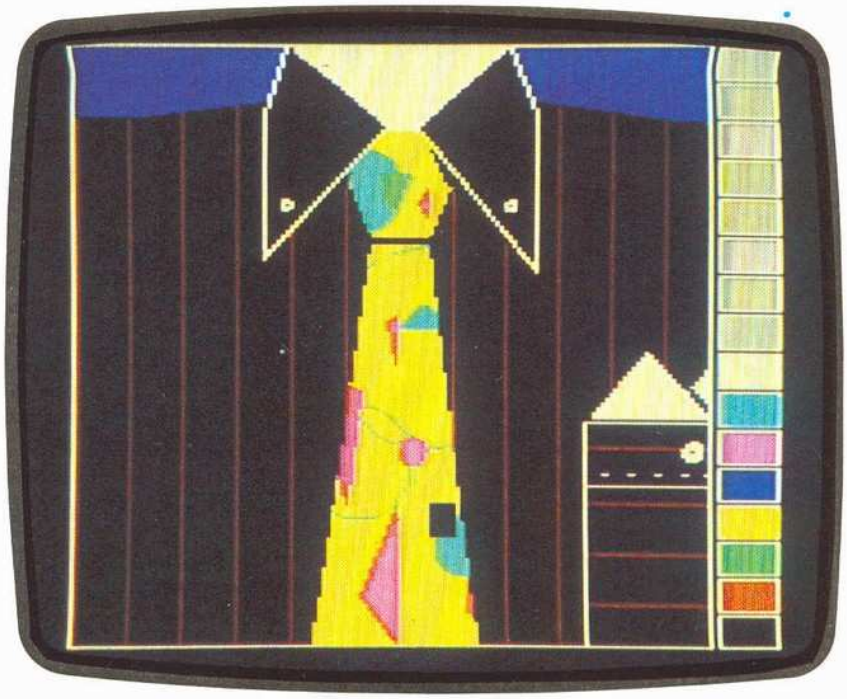
Simples ou sofisticadas, as figuras do vídeo são formadas por milhares de pontos. Memorizando a cor e a luminosidade de cada um deles, o computador é capaz de criar imagens.





Micropintura

Pintar figuras com o auxílio de um microcomputador é fácil, quando se dispõe de um software de gráficos. Esse programa possibilita o desenho de figuras complexas usando controles do teclado. O ponto de partida é, em geral, o posicionamento das linhas retas na tela. Círculos, triângulos, quadrados e outras formas predefinidas podem ser acrescentados. A cor é usada tanto para linhas como para partes da tela. As partes são pintadas selecionando-se a cor com base em uma escala disposta na tela, ou por meio do teclado. A área delimitada é automaticamente colorida. Em alguns programas, um "pincel" move-se pela tela como um cursor. A foto maior mostra o resultado desse processo de "construção". (O programa foi Graphkey, num computador inglês BBC modelo B.)



Você está lá em cima num avião, o motor em pane, um arranha-céu se aproxima, e a pista em que você desesperadamente procura descer está ocupada por um avião em chamas. Isso aparece à sua frente na tela do televisor. É apenas uma das formas de gráfico de computador.

Em programas educativos, como os destinados a alfabetizar ou ensinar aritmética, é evidente que, se não proporcionarem uma exposição visual interessante, a atenção das crianças não será mantida por muito tempo.

Outros usuários de microcomputadores estarão preocupados com números representando quantias recebidas ou gastas com o estoque de mercadorias e assim por diante. Mas esse tipo de informação é muito mais prontamente compreendido e interpretado quando mostrado pictoricamente. A habilidade do computador em "redesenhar" rapidamente uma figura, de forma a incorporar novas informações, propor opções e produzir cópias impressas quando necessário é também de imenso valor em muitas aplicações empresariais.

A tela do computador

Como um computador cria figuras? Para responder a essa questão, observe primeiramente a "tela" em que o computador trabalha. Um microcomputador produz imagens na sua tela de exposição, iluminando ou "acendendo" um ou vários pontos em determinadas posições na tela. Esses pontos são dispostos em linhas no sentido horizontal e em colunas no sentido vertical, de modo que a posição de qualquer ponto é dada por sua localização na linha e na coluna.

Imagens específicas (figuras ou impressões) são produzidas pela iluminação de certos pontos, enquanto os outros permanecem apagados. Isso se aplica não apenas às telas monocrômicas, mas também às exposições coloridas. Nesse caso, para que apareçam as figuras, os pontos adquirem cores apropriadas.

Para expor uma única letra ou número, o computador usa uma disposição retangular de pontos. Essa disposição é conhecida como "matriz de ponto". Num microcomputador típico, deverá consistir em um bloco de oito linhas com oito pontos cada.

O número de pontos na tela não é igual em todos os microcomputadores, mas uma disposição que poderia ser considerada típica consistiria em 192 linhas com 280 pontos cada — em outras palavras, 192 linhas e 280 colunas.

É evidente que, quanto mais pontos existirem na tela do micro, mais detalhadas serão as imagens. O grau de detalhe obtido quando se expõem gráficos é conhecido como "resolução". Diz-se que um computador que pode expor 192 linhas com 280 pontos cada tem uma resolução de 280 × 192. Quanto mais alta a resolução, isto é, quanto mais pontos forem

dispostos na tela, menor "granulação" terá a imagem.

Todos os micros têm uma densidade máxima de pontos que podem ser dispostos em suas telas de gráficos, mas alguns são também programados para usar disposições menos densas de pontos. Por exemplo, o micro inglês BBC tem uma resolução máxima de 640 × 256, isto é, uma densidade máxima de pontos dada por 256 linhas, cada uma com 640 pontos. Entretanto, pode também ser programado usando-se apenas 320 dessas colunas, uma resolução de 320 × 256, ou resoluções mais baixas, se necessário. Numa máquina capaz de proporcionar esse tipo de variação, a resolução deve ser definida no começo da parte do programa que produz os gráficos.

As linhas e curvas produzidas por um computador que usa um sistema de gráficos baseado em pontos não são propriamente contínuas, como seriam se fossem desenhadas com uma caneta. São caminhos de pontos iluminados, menos ou mais unidos, dependendo da resolução do sistema. Um sistema com resolução mais baixa poderá produzir apenas curvas desajeitadas e uma linha reta não será tampouco perfeitamente reta, uma vez que a posição dos pontos na tela está em linhas retas apenas em certas direções — como ao longo de linhas e colunas e sobre as diagonais da disposição de pontos.

Para traçar uma linha reta, o sistema de gráficos deve iluminar os pontos que estão mais próximos da trajetória da linha escolhida. O resultado pode ter um efeito de "degrau". Novamente, quanto mais alta a resolução do sistema, menos perceptíveis serão tais "degraus".

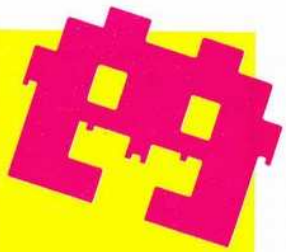
Para se ter continuidade na exposição dos gráficos na tela de televisor, a imagem deve ser continuamente "refeita" ou "redesenhada" — do contrário, seria visível apenas por um momento. Por essa razão, a imagem deve ser representada de alguma forma no computador, para ser consultada quando necessário. A representação da imagem é armazenada numa parte específica da memória do computador, conhecida como memória de tela. Em uma exposição monocrômica, cada ponto corresponde a um bit na memória de tela. Uma imagem é representada estabelecendo-se os bits correspondentes a pontos iluminados, 1, e os bits correspondentes a pontos apagados, 0. Assim, se um computador pode manter uma exposição monocrômica com resolução de 280 × 192, deve ter memória de tela de 280 × 192 ou 53.760 bits — ou 6,5 kilobytes, uma vez que 1 kilobyte é igual a 8,192 bits.

Com os gráficos em cor é necessário mais memória. Dois bits podem ser usados para representar quatro cores, por exemplo da seguinte forma:

bit 1	bit 2	cor
0	0	branco
0	1	vermelho
1	0	azul
1	1	preto

Para representar qualquer imagem de quatro cores é necessário designar dois bits para cada ponto na tela. Da mesma forma, para oito cores, três bits na memória são designados para cada ponto na tela, enquanto para exposições de dezesseis cores há quatro bits por ponto e assim por diante. Desse modo, um microcomputador que expõe cores com resolução de 160 × 256 deve ter uma memória de tela de 160 × 256 × 4, o que é equivalente a 160 kilobits (20 kilobytes).

A necessidade de uma memória de tela explica em parte por que alguns computadores são planejados para trabalhar com diferentes resoluções. Sem uma memória de tela suficiente, o computador não pode armazenar e portanto não expõe figuras de resolução mais alta.

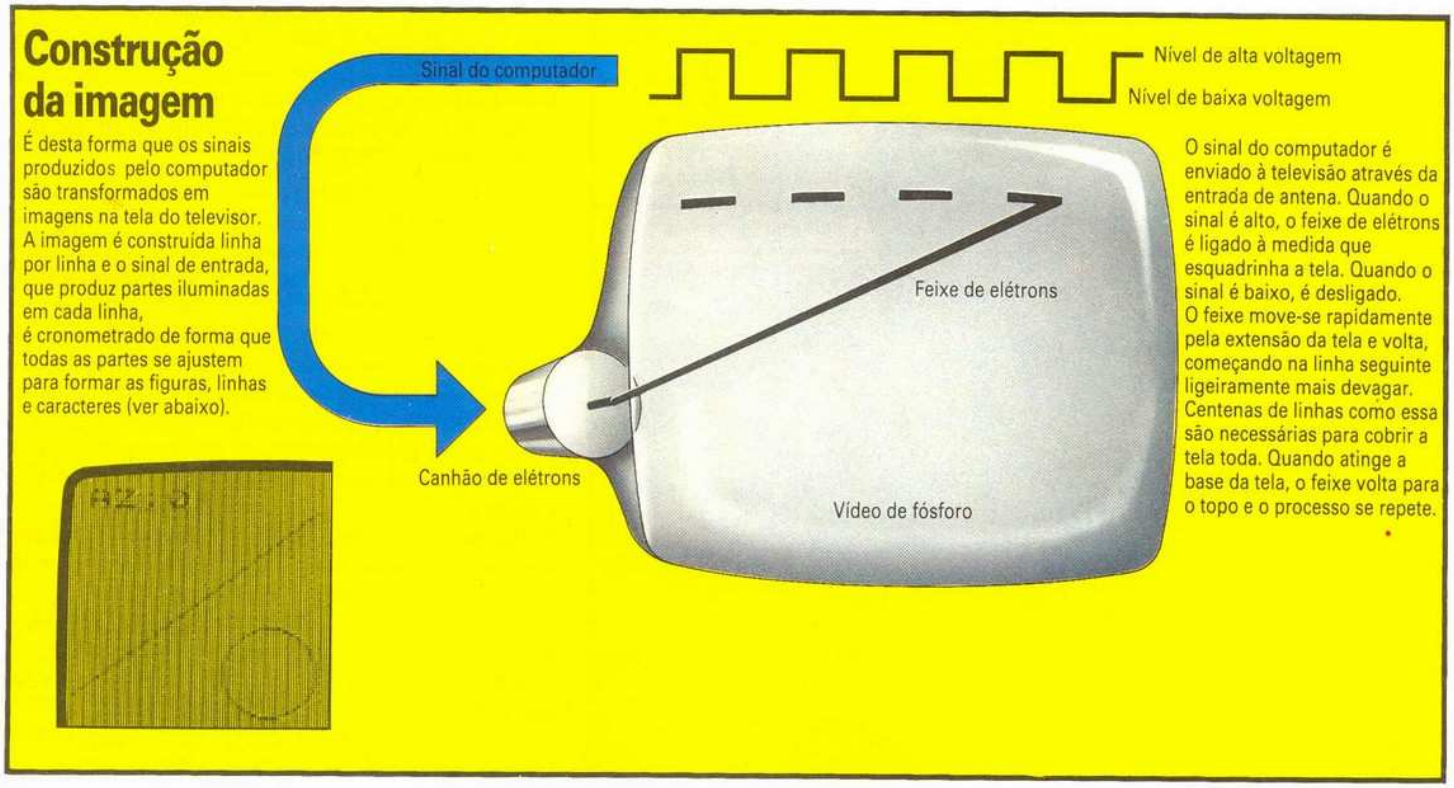


Criação de gráficos

Construir imagens no microcomputador é fácil. A maioria dos modelos tem controles especiais em BASIC, para facilitar o "desenho" ou definir objetos na tela, desde invasores alienígenas até figuras completas. Existem também programas especiais para criar figuras animadas na tela.

Construção da imagem

É desta forma que os sinais produzidos pelo computador são transformados em imagens na tela do televisor. A imagem é construída linha por linha e o sinal de entrada, que produz partes iluminadas em cada linha, é cronometrado de forma que todas as partes se ajustem para formar as figuras, linhas e caracteres (ver abaixo).



O sinal do computador é enviado à televisão através da entrada de antena. Quando o sinal é alto, o feixe de elétrons é ligado à medida que esquadrinha a tela. Quando o sinal é baixo, é desligado. O feixe move-se rapidamente pela extensão da tela e volta, começando na linha seguinte ligeiramente mais devagar. Centenas de linhas como essa são necessárias para cobrir a tela toda. Quando atinge a base da tela, o feixe volta para o topo e o processo se repete.

Pintando com números

Você pode criar imagens e gráficos com seu computador. Os números são os instrumentos, sua imaginação é o limite.

Cria-se um quadro aplicando centenas de pinceladas sobre uma tela. Mas como poderia um artista criar imagens com um computador?

Há três métodos principais para a criação de gráficos com computação e eles diferem quanto ao grau de controle da resolução ou granulação da imagem final. Todos os computadores utilizam gráficos em bloco, "pixels" ou gráficos de alta resolução.

Nos gráficos de alta resolução, o artista busca obter o controle sobre cada ponto de fósforo no monitor de televisão. O controle é limitado apenas pela capacidade de memória do computador, que determina exatamente como a memória da tela modela o vídeo de televisão. Em um computador de 32 K, cada ponto de fósforo tem um correspondente no modelo da tela no computador.

Nos gráficos em bloco, o artista ganha em facilidade o que pode perder em controle sobre cada ponto que compõe a tela. Formas básicas já se encontram prontas nos softwares para a construção de imagens. São controladas diretamente pelo teclado e em geral aparecem na parte frontal de cada tecla. Acionada uma tecla que altera a função das outras, um teclado semelhante ao de máquina de escrever se transforma em "paleta", que possibilita a elaboração de gráficos em bloco.

Cada figura é formada no interior de uma pequena matriz de pontos, de oito linhas por oito colunas. Alguns microcomputadores possibilitam até mesmo definir seus próprios caracteres em bloco. Um programa menor é usado para definir os novos caracteres e acrescentá-los aos do computador.

Os "pixels" situam-se entre os gráficos em bloco e os de alta resolução. Possibilitam ao artista o controle sobre a célula de imagem (picture cell, daí o nome "pixel"), que contém mais do que um ponto de fósforo, porém menos do que um bloco de oito pontos por oito. Cada "pixel" pode ser chamado se-

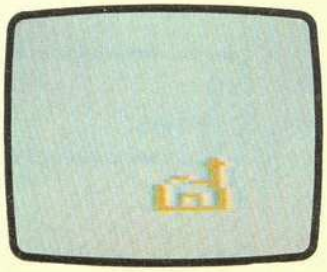
Os três tipos de resolução

Nos gráficos em bloco de baixa resolução, pouca capacidade de memória, como 1 Kbyte, é suficiente para armazenar todos os detalhes do que deve aparecer na tela. Se houver apenas quarenta blocos em cada linha e só 24 linhas na tela, o número total de blocos será de 960, quase 1 kilobyte. Assim, 1 kilobyte de memória para a tela pode armazenar até 8 bits de dados para cada bloco na tela.

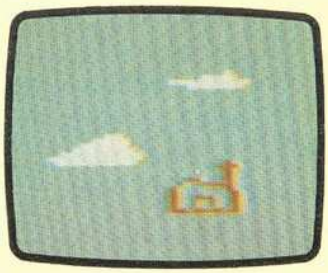
Em gráficos de média resolução, há necessidade de maior precisão de detalhes; assim torna-se necessária maior capacidade de memória. Oito kilobytes de RAM podem fornecer 65.536 pontos de luz na tela, que se apresentam iluminados ou não. Se a composição for em cores, parte da capacidade da memória será necessária para especificar a cor de cada ponto.

Uma figura em cores, de alta resolução, exige grande capacidade de memória. Composições de alta resolução na tela às vezes têm até 640 pontos em cada linha e 240 linhas na tela, num total de 153.600 pontos. Se a composição na tela for em uma cor, bastarão 19.200 bytes (153.600/8). Será necessária uma quantidade maior de memória se quisermos utilizar mais cores.

Gráficos em dimensão



Os gráficos em dimensão foram inicialmente desenvolvidos pela Texas Instruments, mas são hoje encontrados para vários microcomputadores, inclusive o Texas Instruments TI99/04A, o Commodore 64, os computadores da Atari e o Sord M5. (Até o momento esses computadores não estão disponíveis no mercado brasileiro.)



Em gráficos convencionais, as figuras são construídas em uma tela única, como aconteceria se você as pintasse numa folha de papel. Nos gráficos em dimensão, o artista do computador dispõe de vários "planos" ou camadas, cada qual podendo ter suas próprias figuras. Em alguns computadores com gráficos em dimensão, podem existir até 32 planos.

O modo mais fácil de ter noção do que seriam esses planos consiste em imaginá-los como folhas de plástico transparente: se a folha mais "próxima" ao observador mostrar a figura de uma árvore, enquanto a que estiver atrás apresentar a de uma nuvem, esta será vista passando atrás da árvore, como se estivesse sendo carregada pelo vento, no ar.

Desse modo, podem-se criar efeitos tridimensionais bem convincentes.

Uma vez "desenhado" um "objeto" qualquer (um pássaro, por exemplo), o programador pode deixar de lado os detalhes da construção da figura. Se quiser que se mova na tela, basta especificar a velocidade e a direção.

paradamente e localizado na posição desejada na tela.

Em gráficos de alta resolução, para computadores da linha Apple, é usado um comando BASIC para o desenho das linhas. O comando HPLOT x,y identifica o ponto na posição (x,y) da tela, enquanto o comando HPLOT x,y TO w,z faz com que seja traçada uma linha que vai do ponto (x,y) até o ponto (w,z). As linhas são geralmente numeradas no sentido da parte superior da tela para a inferior, e as colunas numeradas da esquerda para a direita. Assim, o ponto situado na linha zero da coluna zero encontra-se no canto superior do lado esquerdo da tela.

O programa abaixo pode ser processado nos modelos compatíveis com Apple (Unitron, Maxxi, Micro Engenho e outros). Lembre-se de passar o microcomputador para a modalidade gráfica. Digite exatamente como indica o programa, que poderá ser processado em qualquer computador com gráficos de alta resolução.

```
10 HGR
20 HPLOT 92,95 TO 57,125
30 HPLOT 57,125 TO 100,110
40 HPLOT 100,110 TO 143,125
50 HPLOT 143,125 TO 108,95
60 HPLOT 108,95 TO 100,50
70 HPLOT 92,95 TO 100,50
80 END
```

Quando você PROCESSAR esse programa, uma forma trinácria surgirá na tela, semelhante à da ilha da Sicília (e assim se denomina porque Trinácria é o antigo nome latino da ilha). O esquema do programa serve para desenhar qualquer forma constituída por uma rede de linhas unidas. Os números apresentados após o comando HPLOT indicam a posição, na linha e coluna, do ponto até o qual a figura deve estender-se. Pode-se desenhar qualquer coisa na tela usando unicamente esses dois comandos. Esse uso é limitado apenas pela resolução da tela do computador e pela persistência do usuário, pois mesmo as curvas devem ser representadas por pontos e os comandos HPLOT x,y e HPLOT x,y TO w,z possibilitam o controle desses pontos.

O programa apresentado a seguir fornece a imagem de um cone traçado por uma combinação de círculos. Foi desenvolvido de modo a ser processado pelo micro inglês Spectrum, da Sinclair.

```
10 FOR K = 2 T0 40
20 CIRCLE 40 + K,40 + K,K
30 NEXT K
```

Os dois programas ilustram o processo pelo qual um computador pode dar origem a composições gráficas na tela por meio de comados que utilizam números. Todavia, os procedimentos digitais são falhos sempre que se tenta imitar processos de fluxo contínuo. Por isso, há dispositivos especiais de hardware para desenho de imagens, que são conjugados aos microcomputadores e permitem ao artista livrar-se de fornecer os milhares de números necessários para imagens mais apuradas. O digitalizador é um desses dispositivos. O artista "executa" o desenho com uma "caneta" especial e o digitalizador traduz seu movimento para os números de colunas e linhas compatíveis com o computador.

A propósito...

ALTA RESOLUÇÃO

Em computadores da linha Apple (Unitron, Micro Engenho, Maxxi, Exato etc.), a composição da tela é de até 53.760 pontos com uma matriz de 192 × 280 pontos.

PROGRAMA COM CIRCLE

O comando CIRCLE não existe na maioria dos micros encontrados no Brasil. Para se ter o mesmo efeito deste comando, seria necessária a utilização de fórmulas trigonométricas junto aos comandos HPLOT e HPLOT TO.

Outras instruções podem ser utilizadas na elaboração de gráficos. Elas vão variar dependendo do modelo de seu computador e do BASIC que ele utiliza.

Processador portátil
O processamento de palavras está se tornando rapidamente uma das aplicações mais comuns do microcomputador. Novos equipamentos são projetados com características especialmente dirigidas para essa função, entre os quais as telas com 80 colunas (para apresentar as linhas do texto em sua completa extensão), a incorporação de unidades de disco (em alguns casos, um processador de palavras em disco vem incluído no preço) e teclas com funções programáveis, usadas para trabalhar com o texto. Alguns equipamentos, como o da foto, são portáteis — ideais, portanto, para jornalistas e executivos.

O texto perfeito

Com um software adequado, seu micro se transforma em um processador de palavras em condições de revisar, organizar e armazenar textos — e mesmo corrigir erros ortográficos.

O processamento de palavras é uma das tarefas mais úteis realizadas pelo microcomputador. Mas a expressão "processamento de palavras" não deixa claro para a maioria das pessoas do que se trata exatamente. "Como é possível processar palavras?" Costuma ser esta a reação dos que ouvem falar nisso pela primeira vez.

Até recentemente, vistosos anúncios ofereciam "processadoras de palavras para seu escritório". Mas deixavam de informar que o caro equipamento em questão consistia simplesmente em um microcomputador adaptado para rodar programas processadores de palavras. Processadores de palavras são menos versáteis que o microcomputador comum porque executam apenas uma atividade.

Talvez o tipo de programa que a expressão "processamento de palavras" indica fosse mais corretamente designado por "datilografia assessorada por computador". Pelo acréscimo de uma impressora, a maioria dos microcomputadores pode rodar programas de processamento de palavras ou de edição de textos. Assim, basta que o usuário do computador experimente o processamento de palavras para perceber sua grande utilidade.

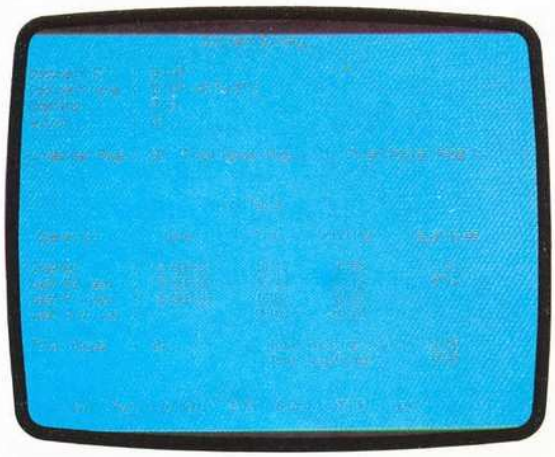
Quando usado como processador de palavras, o computador as apresenta na tela, à medida que são datilografadas no teclado, assim como a máquina de escrever as imprime em papel. Os computadores maiores podem apresentar 80 caracteres em linha na tela, representando a "página". Em computadores

menores, o usuário precisa ser mais paciente. Deve se habituar a uma tela de reduzidas dimensões, e no caso de alguns modelos não terá letras minúsculas à disposição. O usuário também tem de se lembrar que um equipamento menor armazena apenas uma extensão limitada de texto.

O programa proporciona vários recursos sofisticados. Todos os programas processadores de palavras detectam o final da linha à medida que este se aproxima e automaticamente transferem a última palavra por inteiro para o início da linha seguinte. Isso significa que o usuário não precisa ficar atento à mudança de linha, ao atingir a margem: datilografa o texto em seqüência ininterrupta, enquanto o programa cria, quando necessário, uma nova linha. Todavia, o início de parágrafo tem de ser indicado pressionando-se a tecla RETURN.

Em máquinas de escrever convencionais, a correção de erros é feita por procedimentos mecânicos, usando-se borracha ou encobrindo o erro com um corretor e reescrevendo o trecho incorreto. O resultado não é de boa qualidade. Se houver duas ou mais correções, as únicas opções serão enviar uma carta com má apresentação ou refazê-la. Com o processamento de palavras, o problema está solucionado. O cursor luminoso indicará sua posição na tela a cada momento. Você pode movê-lo no texto já feito, até atingir o ponto em que foi escrita a letra ou palavra errada e então eliminar o erro e datilografar corretamente.

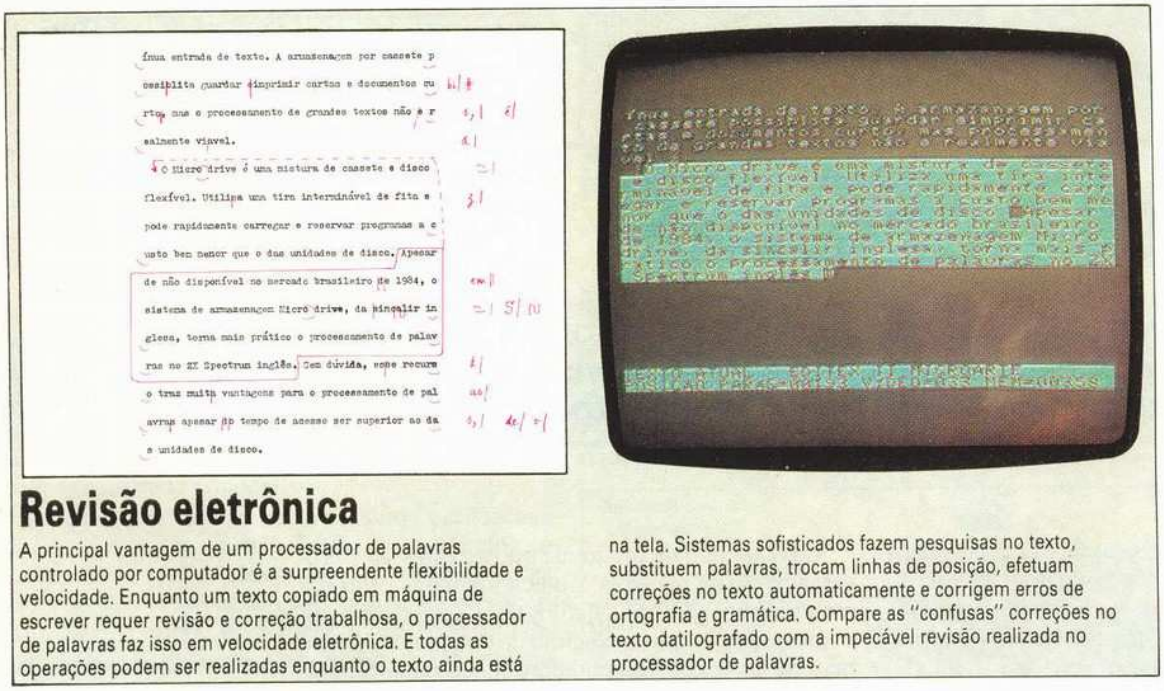
Tendo percebido a capacidade de "correção" do processador de palavras, os usuários serão motivados a dar mais atenção ao preparo do texto. Por exemplo, é possível usar o comando INSERT para acrescentar uma palavra, uma sentença inteira ou mesmo um parágrafo, tão facilmente como se se tratasse de uma única letra. Isso incentiva o usuário a reexaminar o que foi escrito na carta ou no documento. A instrução para eliminar trechos do texto é igualmente fácil. Um comando faz com que palavras e letras simplesmente desapareçam da tela e o texto remanescente se reestruture, recuperando a aparência perfeita da página. Jornalistas e escritores profissionais que já usam processadores de palavras confirmam a melhora de seu trabalho, tanto em qualidade quanto em quantidade.



Vendo o índice
Na foto, o índice de um sofisticado pacote de processamento de palavras. O índice surgirá na tela imediatamente após a inserção do software e indica as várias funções existentes no processador de palavras. Como exemplo poderíamos citar: tabulação e posicionamento de margem, espacejamento de linhas, contagem do número de palavras no documento, reordenação dos parágrafos e elaboração de índices.

Mesmo os menores microcomputadores proporcionam certa capacidade de processamento de palavras. O TK83 e o CP 200 funcionam com um simples programa de edição ou revisão de textos, que permite ao usuário escrever uma carta ou documento na tela e depois fazer as correções. Edição de texto é a expressão geralmente usada para programas limitados de processamento de palavras que trabalham com textos de uma ou duas páginas, mas não podem lidar com documentos mais extensos. A pequena capacidade de RAM do TK83 e de computadores de tamanho semelhante limita muito a extensão do texto a ser composto e elaborado na tela.

Um problema com o TK83 é o teclado tipo "touch", que impede a datilografia rápida. Embora o teclado do TK85 seja muito melhor nesse aspecto, ainda não é do tipo mecânico ao qual as datilógrafas estão acostumadas. Se você pensa em comprar um microcomputador para uso em processamento de



Revisão eletrônica

A principal vantagem de um processador de palavras controlado por computador é a surpreendente flexibilidade e velocidade. Enquanto um texto copiado em máquina de escrever requer revisão e correção trabalhosa, o processador de palavras faz isso em velocidade eletrônica. E todas as operações podem ser realizadas enquanto o texto ainda está na tela. Sistemas sofisticados fazem pesquisas no texto, substituem palavras, trocam linhas de posição, efetuam correções no texto automaticamente e corrigem erros de ortografia e gramática. Compare as "confusas" correções no texto datilografado com a impecável revisão realizada no processador de palavras.

palavras, deve examinar o teclado do equipamento, já que tem influência considerável na velocidade e comodidade de execução do trabalho (ver p. 36).

Após a aquisição da impressora, o principal problema quanto ao hardware vem a ser a existência de capacidade suficiente de memória para armazenar o texto. É possível utilizar um gravador cassete para a armazenagem. Mas esse tipo de equipamento limita a extensão do texto que pode ser escrito, pois a memória rapidamente se esgota. Isso não acontece com discos flexíveis, que vão automaticamente extraindo texto da memória, deixando-a livre para contínua entrada de texto. A armazenagem por cassete possibilita guardar e imprimir cartas e documentos curtos, mas o processamento de grandes textos não é realmente viável.

Apesar de não disponível no mercado brasileiro em 1984, o sistema de armazenagem Microdrive, da Sinclair inglesa, torna mais prático o processamento de palavras no ZX Spectrum inglês. O Microdrive é uma mistura de cassete e disco flexível. Utiliza uma tira interminável de fita e pode rapidamente carregar e reservar programas a custo bem menor que o das unidades de disco. Sem dúvida, este recurso traz muitas vantagens para o processamento de palavras, apesar do tempo de acesso ser superior ao das unidades de disco.

Eficiência e economia

O processamento de palavras é um método eficiente porque separa o ato de composição do ato de impressão. Tanto a escrita manual como a datilografia exigem que a palavra seja efetivamente escrita ao mesmo tempo que ocorre o processo de pensamento. Pelo processamento de palavras, nenhuma palavra aparece no papel até que a composição na tela alcance a forma desejada. Mas o surgimento de palavras no papel exige uma impressora — e uma impressora de baixo custo, adequada à emissão de listagens de programa, dificilmente poderá produzir cópias de boa qualidade (ver p. 74).

Os programas processadores de palavra estão agora disponíveis também em chips que podem ser conectados ao painel de circuitos do computador. São muito úteis na ausência de uma unidade de disco. Proporcionam a vantagem de rápido carregamento do programa, que se torna imediatamente disponível para uso. Se a memória RAM do computador tiver capacidade suficiente (32 Kbytes ou acima disso), você poderá elaborar documentos de até 5.000 palavras e fazer sucessivas revisões até chegar à forma satisfatória. Se quiser arquivar o texto revisado após a impressão, será necessário guardá-lo em fita cassete, pois o chip de processamento de palavras não pode armazenar o texto que você cria.

Alguns sofisticados processadores de palavras executam funções suplementares muito úteis; o dicionário automático, ou conferidor de ortografia, é um dos mais conhecidos. Para beneficiar-se desta comodidade, você necessitará de uma unidade de disco. O processador verifica as palavras escritas em um documento e as compara com aquelas armazenadas em seu arquivo, indica as que não reconhece e chama a atenção do usuário, que as corrigirá, se for o caso.

À medida que o uso do processador de palavras se amplia, é provável que em pouco tempo seja considerado um instrumento essencial tanto para os serviços de escritório quanto para os de correspondência.

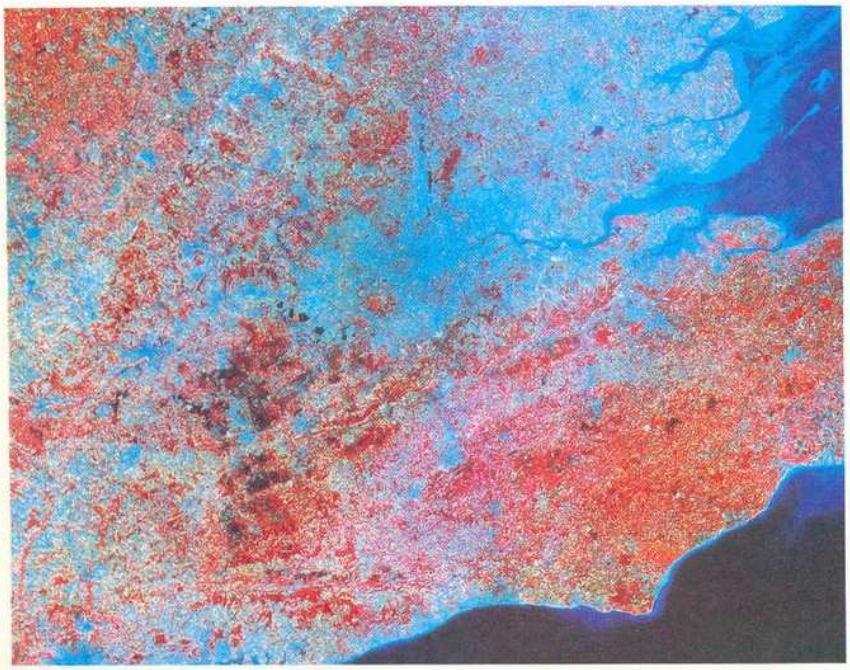
Aplicações básicas de um sistema de processamento de texto

CORRESPONDÊNCIAS SEMI-STANDARD	Cartas de transferência Propostas Certificados Convocações Cartas de cobrança etc.	Caracterizam-se por textos repetitivos (standard), onde deverão ser inseridos dados variáveis
CORRESPONDÊNCIAS COMPLEXAS	Contratos Fórmulas jurídicas Descrição de produtos Ofertas	Obtidas mediante a composição de vários parágrafos standard
CORRESPONDÊNCIAS LONGAS	Manuais Apostilas Relações Editais Relatórios Atas etc.	Comportam a datilografia de várias páginas, que devem ser corrigidas e logo após impressas
CORRESPONDÊNCIAS PERSONALIZADAS	Mala direta Circulares Cartas promocionais Cartas de aniversário Cartas convites etc.	Correspondências que deverão ser enviadas a uma grande lista de endereços
CORRESPONDÊNCIAS VARIÁVEIS	Memorandos Pró-memórias Requisições Comunicados Solicitações	Correspondências cujo teor será sempre diferente

Consulte o chip

Muitos profissionais poderão ficar livres da rotina graças aos sistemas especializados: computadores programados para analisar e responder a perguntas complexas e variadas.

Inteligência artificial, computadores que pensam e tomam decisões como seus criadores humanos, por enquanto, são uma fantasia da ficção científica. Ainda não se chegou ao conhecimento total de como funciona o cérebro humano e, apesar de alguns avanços nesse terreno, há poucos projetos de um computador "estilo 2001".



Uma visão do espaço
O satélite Landsat 4 foi lançado em julho de 1982 e está numa órbita que cobre toda a superfície do planeta. Ele retorna ao mesmo ponto de sua órbita a cada vinte dias. Todos os dados enviados pelo sensor vêm em forma digital. Na ilustração a informação foi processada fotograficamente, a fim de apresentar aspectos de Londres e do Sudeste da Inglaterra. A água limpa é representada em azul-escuro; a água rasa com sedimentos, em azul-claro; as cidades e os campos cultivados, em azul-acinzentado; os campos abertos, em marrom-avermelhado; o milho bom para colheita, em verde; e outras vegetações aparecem em vermelho-vivo.

Mas, no caso de uma tarefa específica, e se um computador precisa apenas "parecer" inteligente, atuando num campo muito restrito da atividade humana, então a simples simulação da inteligência torna-se algo bem mais fácil.

Esta é a teoria que está por trás dos sistemas especializados. A idéia se baseia no trabalho realizado por um especialista de determinada área. Por exemplo, um geólogo ou um cirurgião podem alimentar um computador com códigos e conhecimentos especializados. Depois de trabalhar as informações recebidas, o computador responde a pessoas não especializadas, sobre um campo específico.

Os sistemas especializados poderiam ter várias utilidades. Um programa foi desenvolvido para diagnosticar a causa de dor de estômago mediante perguntas aos pacientes sobre os sintomas. Outro programa se vale de nosso conhecimento de geologia para apontar os locais onde provavelmente seria achado molibdênio ou outros minerais. E outro deduz as prováveis estruturas para moléculas orgânicas, a partir de massas de dados experimentais não estruturados. Todas essas tarefas seriam normalmente realizadas por um profissional com muita experiência e devidamente treinado. Graças ao computador, essas pessoas podem agora dedicar seu tempo a atividades mais originais.

Mas os sistemas especializados significam muito mais que a simples substituição dos especialistas humanos. Uma vez que o conhecimento de um especialista está sendo usado pelo programa, o computador freqüentemente revela dados inesperados. Às vezes, o equipamento aponta uma relação entre itens de informação que os homens não conseguiram captar; e sugere ainda novos caminhos a serem explorados.

Assim, pode-se dizer que os sistemas especializados são, ou serão brevemente, um passo importante no que se refere à utilização do computador. Muitos computadores podem utilizar o mesmo programa, substituindo o conhecimento especializado de uma única pessoa pelo de um grande número de computadores igualmente especializados.

O único problema para os pesquisadores é escrever um programa que trabalhe adequadamente; um programa que seja exato como um computador e inteligente como um especialista humano.

A criação do programa

O primeiro passo é saber como os especialistas humanos se decidem sobre as evidências e sobre as perguntas que dizem respeito à sua especialidade. Quando comparado com o funcionamento de um computador, o pensamento humano não se mostra totalmente lógico e apóia-se muito na experiência. Quando um especialista humano depara com um novo problema, fatalmente o compara com situações vividas antes. A partir dessas comparações, algumas tentativas são feitas a fim de se obter o melhor resultado.

Mas, considerando o detalhado conhecimento de um especialista — por exemplo, um médico —, veremos que grande quantidade de regras tem de ser armazenada e associada de modo muito complexo. Além disso, modificações posteriores também serão necessárias para que o computador possa imitar o comportamento humano. Um médico nem sempre está seguro do que afirma e usa as expressões "estou quase certo de que..." ou "bastante confiante", ao expressar uma opinião. Baseado em alguns sintomas, o médico pode apenas ter 30% de certeza de seu diagnóstico.

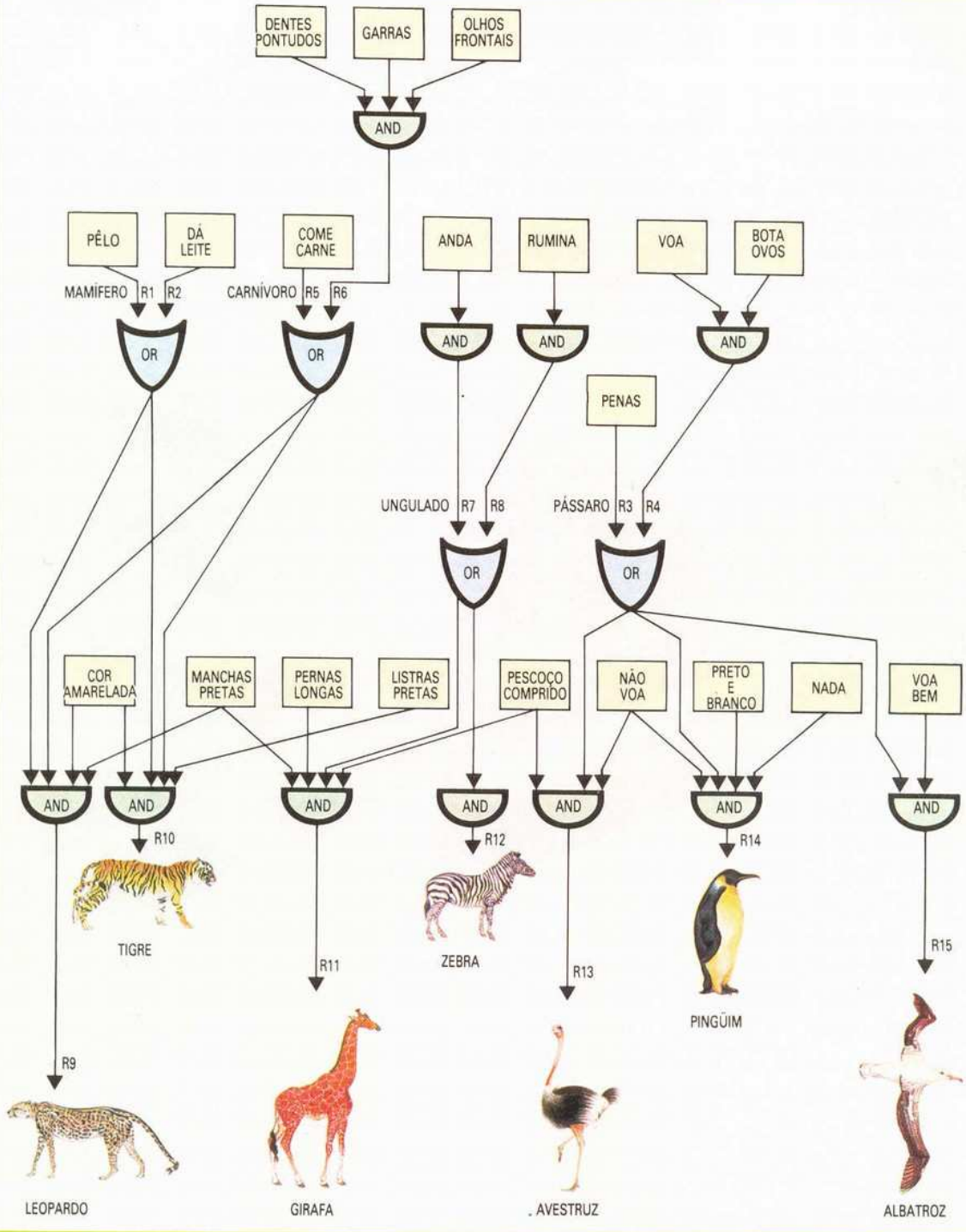
Assim, as regras num computador devem ter valores de probabilidade relativos às suas conclusões, indo de 100%, quando há apenas uma conclu-

são possível, até 1%, quando há outras cem conclusões igualmente prováveis.

A maioria dos sistemas especializados funciona por meio de um diálogo entre o entrevistado e o computador. O entrevistado deve detalhar os problemas a serem resolvidos pelo computador; e o modo mais simples de evitar problemas aqui é ter o computador fazendo uma série de perguntas de múltipla escolha ao usuário. Desse modo, o usuário não corre o risco de usar termos e frases que o computador não compreende. A seguir, o computador testa a informação de acordo com as normas depositadas no seu estoque de conhecimento. O modo como o computador chega ao circuito das normas depende das respostas que o usuário dá às suas perguntas. À medida que o programa avança, surge nova pergunta do computador.

O mais conhecido sistema especializado na Inglaterra, o programa Mickie de diagnóstico médico, funciona num microcomputador. E outros produtos comerciais a serem lançados transformarão o computador que se tem em casa em um especialista de vários assuntos.

Talvez ainda não possamos conversar com os computadores. Mas eles já nos dão respostas em que certamente podemos confiar.



Uma rede de animais

Esta tabela mostra como o "conhecimento" de um sistema especializado é associado, formando uma série de normas simples. Aqui, as normas são numeradas de R1 a R15 e têm o objetivo de identificar um animal por meio de informações a seu respeito. Por exemplo: suponha que você tenha a informação de que o animal possui dentes pontudos. O computador começa na primeira caixinha à esquerda da tabela e verifica onde ela pode ser colocada na rede, indo em direção a um dos animais que estão no fim da tabela. O AND, que aparece no círculo, significa que as três condições mostradas nas caixinhas têm de ser verdadeiras, antes que R6 seja aplicado. O computador não é capaz de progredir até que receba novas informações. Pergunta, então, se o animal tem garras e se tem os olhos na frente da cabeça (olhos frontais). Se ambas as respostas forem positivas, R6 permite que o computador prossiga. Mas R6 leva a um OR, o que significa que a operação só poderá continuar se R6 ou R5 for verdadeiro. Neste caso, R6 é verdadeiro e o computador segue em frente. Aqui existem duas possibilidades de AND, cada uma levando a diferentes conclusões. Para que R9 ou R10 se realize e a resposta LEOPARDO ou TIGRE seja dada, o computador deve perguntar se o animal tem listras pretas ou manchas escuras.
A informação traz, assim, uma resposta definitiva.

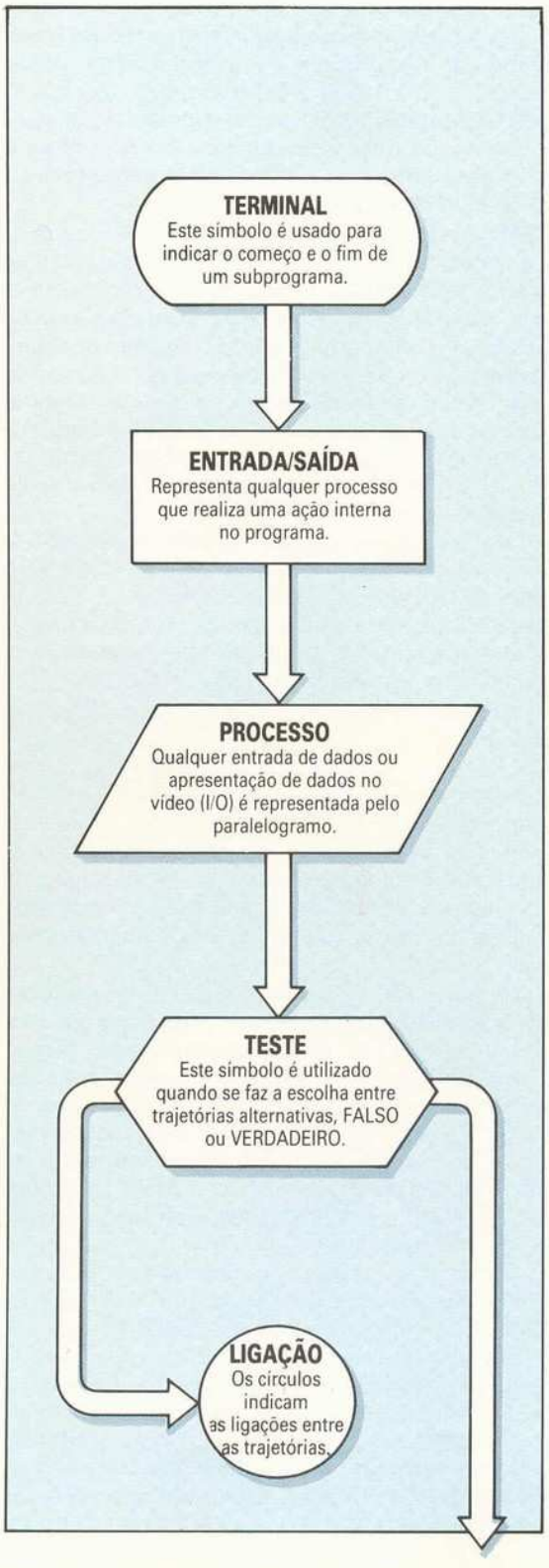
O mapa lógico

Os fluxogramas ajudam você a criar programas eficientes e com uma estrutura lógica bem elaborada.

O fluxo de informação
O objetivo do diagrama é indicar, de maneira concisa e simples, o fluxo de informação e controle em um programa de computador.

Os pontos de teste são os principais, quando o controle passa para um ponto que não é o seu consecutivo. Uma simples representação gráfica desta passagem de controle é de mais fácil compreensão do que um comando semelhante em forma escrita. Às vezes uma representação gráfica vale bem mais que milhares de palavras.

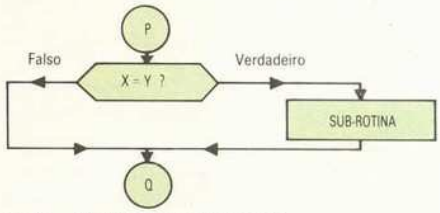
O símbolo de "teste" pode ser representado por um hexágono plano, como é mostrado aqui, ou por um losango.



Um problema pode ser representado de maneira ilustrativa por diagramas que mostram as fases de um processamento e a trajetória dos fluxos que o ligam. Esses fluxogramas são úteis para se compreender um problema e elaborar uma solução.

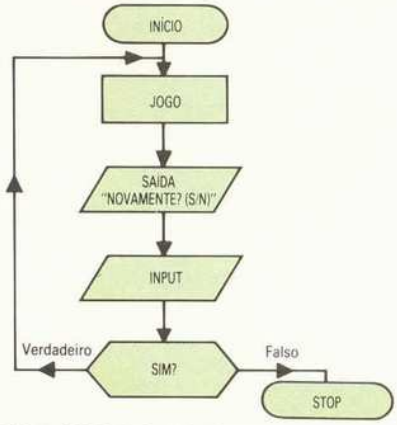
No fluxograma, cada símbolo representa um processo ou ação, e as linhas que ligam esses símbolos representam as possíveis trajetórias a serem feitas. Em um "fluxo de tráfego" de mão única, as setas indicam a direção, que no diagrama é normalmente de cima para baixo e da esquerda para a direita.

Todas as vezes que uma decisão deve ser tomada, um "símbolo de decisão" com a forma de um hexágono ou de um losango é usado. O controle flui em uma única trajetória, como anteriormente, mas pode sair em uma das direções, dependendo do resultado do teste em questão. Se o objetivo do teste é adotar ou não um procedimento, apenas uma das trajetórias de saída terá um "símbolo de procedimento". Aqui está um exemplo de um teste que visa a decidir se o desvio para a sub-rotina deve ser feito:




```
120 IF X = Y THEN GOSUB 300
```

O símbolo de decisão é também utilizado para indicar o teste que termina um loop. No exemplo abaixo, o controle retorna para o começo do programa, se houver uma resposta positiva para a pergunta "NOVAMENTE?"




```
90 REM** COMEÇO DO JOGO**
100
800
```

```
810 PRINT "NOVAMENTE? (S/N)";
820 INPUT R$
830 IF R$ = "S" THEN GOTO 100
840 END
```

Talvez queiramos tomar uma decisão que irá resultar em um de dois cursos de ação totalmente diferentes a serem seguidos. No exemplo abaixo, comparamos o placar do jogador com o placar máximo dos jogos anteriores:


```
1200 IF PLACJOGO > PLACMAX THEN GOTO 1230
1210 PRINT "MA SORTE, CONTINUE TENTANDO";
1220 GOTO 1250
1230 LET PLACMAX = PLACJOGO
1240 PRINT "PARABENS! NOVO RECORDE!";
1250 PRINT PLACMAX
```

Note que o valor de PLACMAX é impresso em ambos os eventos, e as duas trajetórias de fluxo possíveis reúnem-se no processo e tornam-se uma única entrada para esta operação de saída.

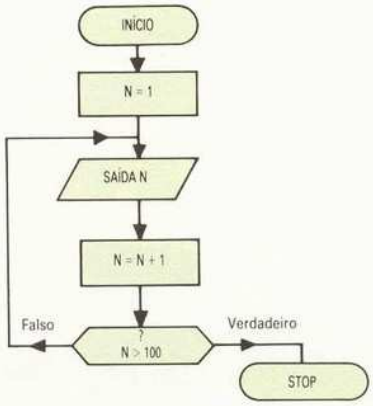
Todas as decisões são tomadas com base em testes semelhantes a este, que trazem um resultado negativo ou positivo, falso ou verdadeiro. Como se pode notar, este processo de decisão totalmente binário não admite respostas duvidosas. Você pode usar quantos símbolos de decisão quiser, mas é importante que duas trajetórias de saída sejam adequadamente marcadas.

Toda linguagem de programação tem um comando de decisão básico. Se o resultado positivo (verdadeiro) for alcançado, ele leva a uma declaração condicional; caso o resultado seja negativo (falso), ele transfere o controle para o comando seguinte. No caso de versões de BASIC que permitam somente o IF-THEN simples, temos de imitar a escolha condicional através de um comando GOTO, como na linha 1200 do último exemplo. O comando da linha 1210 só será executado se o resultado do teste na linha 1200 for falso.

Mas o que dizer do segundo uso de GOTO na linha 1220? Como se pode perceber, o uso do GOTO no fim do teste, para resolver o problema do destino da escolha condicional, forçou-nos a usar este método para reunir as duas trajetórias de controle possíveis, neste caso, na linha 1250.

O uso dos fluxogramas estimula a introdução dos GOTOs como forma de seguir toda a representação do programa. Geralmente, o uso de saltos incondicionais é muito perigoso. No caso de a versão do BASIC utilizada forçar essa solução, o fluxograma é um excelente meio de se avaliar o modo como o controle sai da sucessão normal do programa.

Um último exemplo mostra claramente como o uso do fluxograma nos permite representar exatamente as etapas necessárias para se realizar uma simples tarefa: imprimir todos os números entre 1 e 100.




```
10 LET N = 1
20 PRINT N
30 LET N = N + 1
40 IF N > 100 THEN END
50 GOTO 20
```

O uso de fluxogramas como o aqui mostrado tende a incentivar um processo, passo a passo, de programação escrita que, especialmente em grandes projetos, leva a um resultado bastante deselegante. O uso da repetição FOR-NEXT geralmente é indicado até mesmo para pessoas com um conhecimento superficial da linguagem BASIC. Por exemplo:

```
10 FOR N = 1 TO 100
20 PRINT N
30 NEXT N
40 END
```

O fluxograma é incapaz de representar esse pequeno programa BASIC, e a tentativa de segui-lo traria outros problemas. Entretanto, o fluxograma dá algumas informações sobre a estrutura de repetição FOR-NEXT, por isso tem seu valor. Ele demonstra de modo claro como são construídos esse e muitos outros comandos BASIC.

Os fluxogramas são úteis no estágio de planejamento e definição da programação, especialmente nas partes mais complicadas. Os programadores experientes tendem a usá-los menos que os iniciantes, e apenas recorrem aos fluxogramas para ilustrar e documentar um software escrito sem sua participação direta.

O fato de o fluxograma ser desenhado no papel ou estar apenas no subconsciente do programador não faz diferença. A diagramação do fluxo de informação e controle é essencial como forma de solucionar problemas.

Siga as pistas

O computador pode selecionar fatos e elaborar listas a partir das numerosas informações armazenadas em um banco de dados.

Um conjunto de dados mantidos e consultados pelo computador é conhecido pelo nome de banco de dados. Todos nós usamos vários bancos de dados não computadorizados em nosso dia-a-dia.

A lista telefônica é um exemplo de banco de dados não computadorizado. Entretanto, não é necessário catalogar ou armazenar dados em uma determinada ordem para termos um banco de dados. Em um computador, essa ordenação rígida dá origem a sérias limitações.

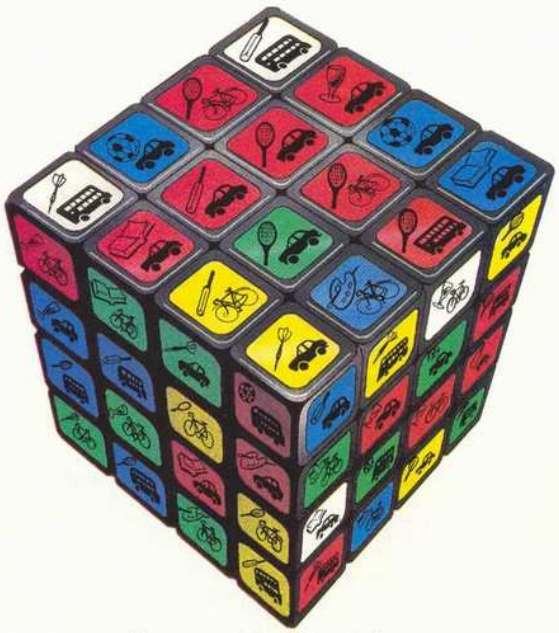
Programas de banco de dados são conjuntos de rotinas que permitem fazer seleções dos dados. Tais programas variam desde simples sistemas de fichário até linguagens completas. Geralmente, um banco de dados computadorizado é extenso e contém dados de muitos tipos. Mas isso não significa que você precisa ter um equipamento enorme. Qualquer computador pode controlar um eficiente banco de dados. As únicas limitações sérias estão no tamanho e velocidade dos recursos de armazenamento.

Poderíamos, por exemplo, fazer uma lista com dados sobre várias pessoas. Colocando essas informações em um fichário comum, teremos um conjunto semelhante a um fichário pessoal. Verificamos logo que, nesse caso, há muitas espécies de dados. Em cada categoria, muitos itens são palavras e alguns são números. Há possibilidades limitadas em algumas categorias: "sexo" (masculino e feminino) e "estado civil", que será um dos elementos do conjunto "solteiro, casado, divorciado, desquitado, viúvo".

Pode-se transformar certos itens em listas de palavras e números. Por exemplo: profissão, nome da empresa, endereço comercial, telefone comercial e nome do superior hierárquico serão convenientemente agrupados sob o título "vida profissional", e modelo e ano do carro ficarão incluídos na lista "carro".

Ampliando essa idéia, podemos conservar todos os endereços em forma de listas. É melhor do que mantê-los em um único item, pois talvez desejemos saber apenas em que cidade uma pessoa reside, e não em que rua.

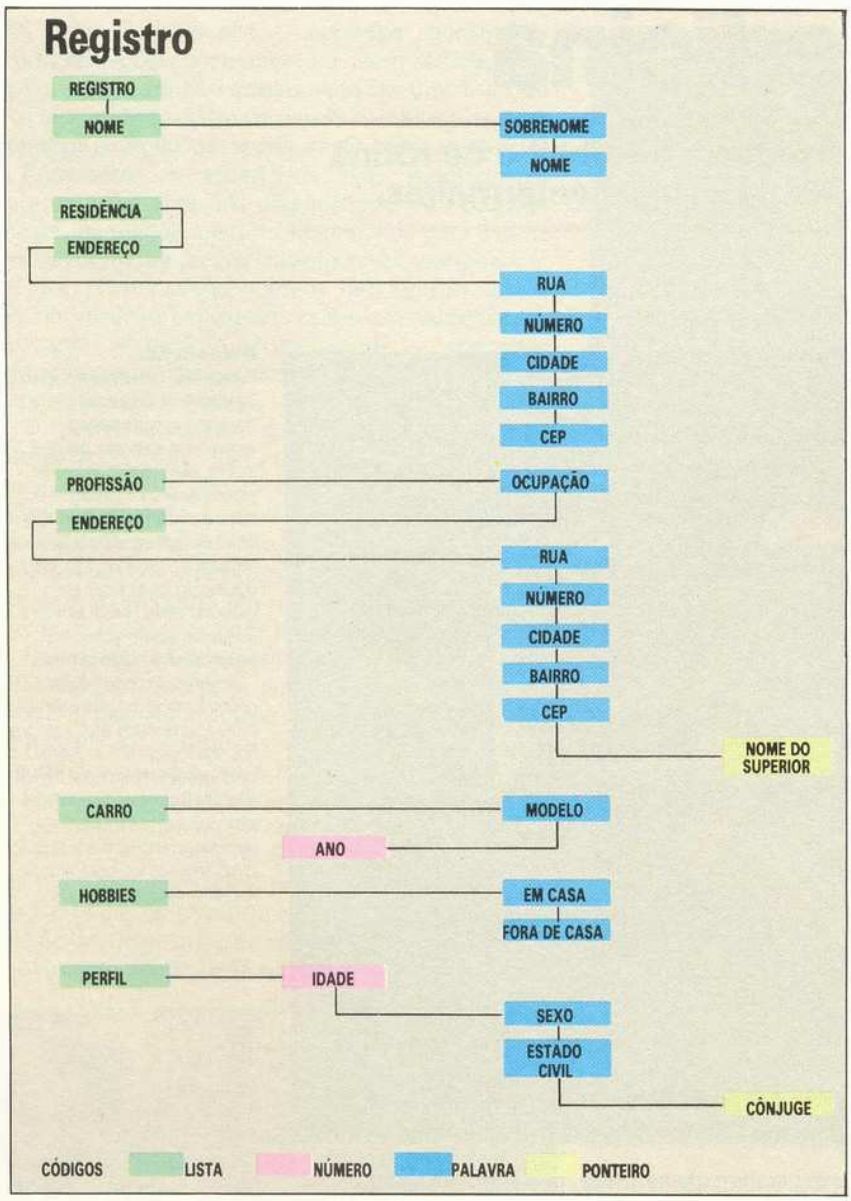
É possível também ampliar o item "estado civil" pelo acréscimo do nome do cônjuge, quando for necessário. O nome pode ser uma só palavra, mas,



Alguém se habilita?
Usamos o cubo mágico como uma analogia para um banco de dados rudimentar — isto é, contém todos os dados de que precisamos, mas ainda não colocados na ordem certa. Neste exemplo, estamos procurando um parceiro de tênis (o símbolo da raquete) que possua carro (o carro) e que esteja disponível no dia em questão (os quadrados vermelhos).

Fazendo as perguntas certas

Muitos programas de banco de dados comerciais usam códigos semelhantes a linguagens de programação. No exemplo acima o termo "ACCESS" indica ao programa que queremos consultar um arquivo previamente criado. O termo "USE" indica que vamos usar um subconjunto do arquivo "FRIENDS" e, "enquanto" ("while") essas condições forem mantidas, poderemos obter todos os itens que estão sob os títulos "CAR" ou "TENNIS". O resultado é uma lista de amigos, que dirigem carro e jogam tênis.



Informações organizadas
Um banco de dados organizado em hierarquia permite ao usuário passar de uma parte da informação a outra, tornando possível novas escolhas a cada vez. Nenhum conhecimento do conteúdo está aqui presumido.

como se refere a uma pessoa e o arquivo é sobre pessoas, seria conveniente que pudéssemos recorrer a outro registro.

Por estar em um lugar determinado do arquivo, cada registro tem um número. Assim, podemos usar o número do registro que descreve a pessoa, em vez de seu nome, para estabelecer uma referência cruzada.

Tal item de dados é chamado "ponteiro". Se usarmos essa técnica para referências ao superior da pessoa na empresa, o resultado será uma estrutura semelhante ao que se chama "registro".

A diferença entre um fichário e um banco de dados computadorizado está no fato de que o primeiro pode ter apenas uma ordem, geralmente alfabética.

O fichário será adequado se quisermos descobrir, por exemplo, qual empresa mantém como empregado uma determinada pessoa. Mas que faremos se quisermos saber quais os nomes de todos os empregados de uma empresa?

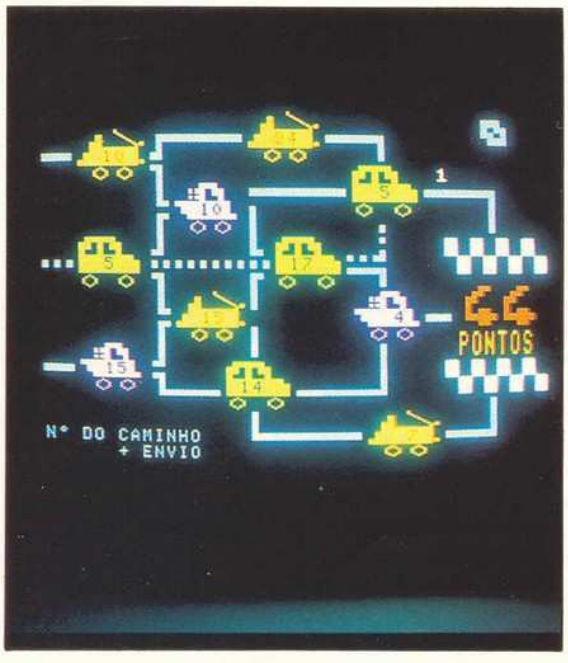
Pelo uso de um fichário, teríamos de examinar todos os cartões, retirando os que possuam o dado procurado. Esse procedimento não apenas consome muito tempo, mas também pode resultar em erros.

Com um sistema computadorizado, entretanto, podemos fazer com que a máquina verifique cada registro sucessivamente e imprima o nome de todas as pessoas que trabalham na empresa.

Outra possibilidade seria fazer o computador reorganizar o arquivo, colocando como item mais importante o campo de atividade da empresa. Isso resultará no mesmo banco de dados, com os mesmos dados, mas com uma estrutura completamente diferente. A reorganização colocará todas as ocorrências de determinada empresa em um grupo, e esta seção nos dará os nomes de todos os empregados. Em um fichário, há apenas um item básico para ordenação em geral: o nome, por exemplo. Porém, em um sistema computadorizado, a ordenação pode ser baseada em qualquer item.

Desse modo, pode-se acessar uma determinada informação do banco de dados com muita facilidade e rapidez.

A enciclopédia de hoje
Serviços como o videotexto são tentativas para tornar grandes bancos de dados acessíveis ao público em geral, através de serviços que cada um pode ter em sua própria casa. Esse tipo de sistema usa a televisão comum como tela de monitor, e um teclado ligado ao computador central por uma linha telefônica comum. O acesso ao banco de dados faz-se por meio de um "menu" de serviços disponíveis. As opções são apresentadas na tela, e o usuário chega até a "página" de dados desejada percorrendo uma determinada hierarquia. Dentro de pouco tempo, o fornecimento do código de seu cartão de crédito lhe possibilitará fazer compras sem sair de casa.



Gráficos em dimensão

A grande capacidade de memória permite a alguns micros produzir imagens coloridas e com rápidos movimentos.

Um dos aspectos mais notáveis dos microcomputadores está na sua capacidade de criar gráficos e composições com movimento, o que é conhecido como animação. Em muitos microcomputadores, o usuário traça pontos individuais, desenha linhas e círculos e altera as cores do fundo e do primeiro plano.

Para jogos de simulação e de ação rápida, precisamos ter condições de simular movimentos. O meio mais fácil de fazer isso é produzir uma série de figuras imóveis, uma após outra, e com rapidez suficiente para dar a ilusão de movimento. As imagens de televisão são obtidas de modo semelhante.

Velocidade de ação

Outro recurso utilizado para criar a ilusão de movimento consiste em imprimir um caractere, apagá-lo e imprimi-lo de novo em posição ligeiramente afastada da inicial. A cada passo, a distância do afastamento deve ser mínima, para obtermos um fluxo contínuo de movimento. Analogamente, o tempo para produzir a forma e apagá-la deve ser o menor possível.

O uso da linguagem BASIC para a produção de animação resulta em movimentos lentos. Um modo de superar esse inconveniente é recorrer à linguagem Assembly, um procedimento que exige prática, cuidado e atenção, para obtermos composições isentas de oscilações. Além disso, temos o problema adicional de controle dos caracteres individuais, à medida que estes se acumulam na tela.

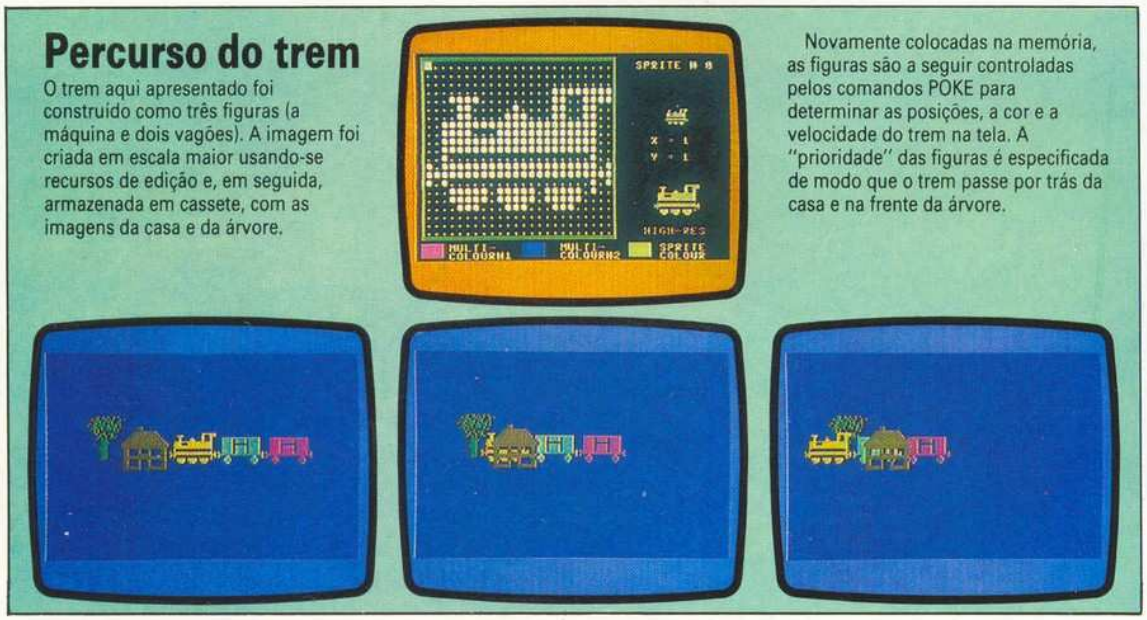
Muitos computadores, tais como o Commodore 64, o Sord M5, o Texas Instruments TI99/4A e os da linha Atari, superam o problema utilizando as mesmas técnicas empregadas nas máquinas de fliperama. A técnica é conhecida como ''sprite graphics''.

''Sprites'' são ''objetos'', ou figuras, que se movimentam na tela de modo independente uns dos outros. Isso é feito pela simples alteração dos conteúdos de duas posições de memória especificadas pelas coordenadas X e Y (as posições direita-esquerda e superior-inferior). De modo geral, a coordenada X pode variar de 0 a 255 e a coordenada Y de 0 a 191. Alguns equipamentos chegam a admitir a especificação da velocidade e da direção do movimento para cada figura, deixando ao computador o trabalho restante.

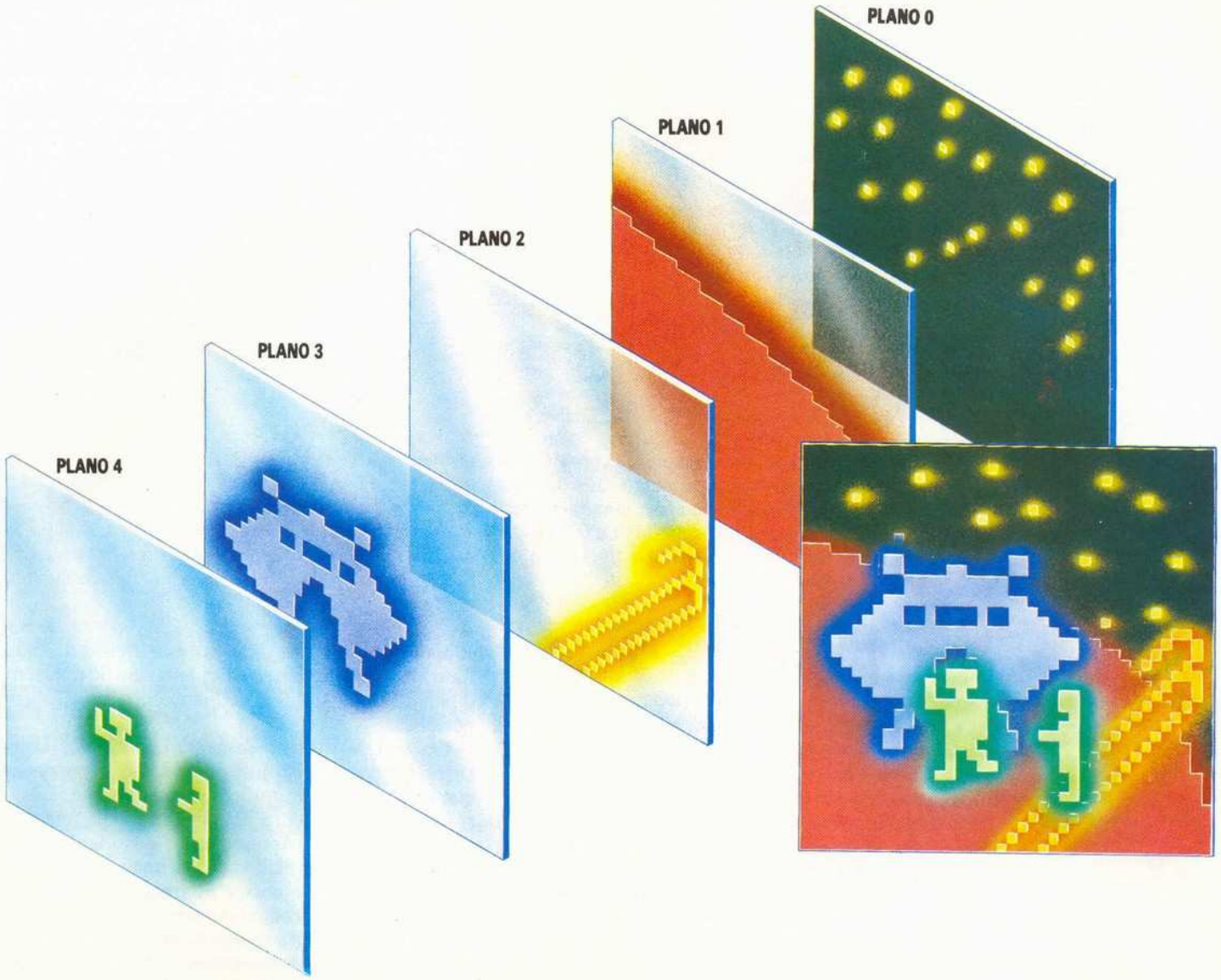
Para este tipo de gráfico, são necessários, em geral, chips ou circuitos especiais pertencentes ao hardware do computador. É possível comprar software feito para outros computadores com a finalidade de obter efeitos semelhantes, mas, quase sempre, os resultados são menos satisfatórios.

Percurso do trem

O trem aqui apresentado foi construído como três figuras (a máquina e dois vagões). A imagem foi criada em escala maior usando-se recursos de edição e, em seguida, armazenada em cassete, com as imagens da casa e da árvore.



Novamente colocadas na memória, as figuras são a seguir controladas pelos comandos POKE para determinar as posições, a cor e a velocidade do trem na tela. A "prioridade" das figuras é especificada de modo que o trem passe por trás da casa e na frente da árvore.



A velocidade de cada um dos elementos varia, pois eles se situam em ''planos'' diferentes. Desse modo, uma tela repleta de gráficos é constituída de uma série de planos sobrepostos, embora perceptíveis aos olhos apenas como um único plano na tela.

Efeito tridimensional é obtido pela passagem da figura na frente ou atrás de outra. Os elementos são numerados de 0 até o maior número disponível, que varia de acordo com o equipamento. Se dois elementos se sobrepõem, o que tem número menor aparece na tela. Desse modo, pela cuidadosa ordenação das figuras, obtém-se facilmente efeito tridimensional, como o de um trem passando na frente de uma árvore. Outra possibilidade é encobrir parcialmente o trem, quando este passa por uma casa situada em um plano de numeração mais baixa.

Pelo uso da variedade de cores que seu computador proporciona, cada figura pode ser colorida individualmente. Em alguns casos, pode-se ampliá-la ou reduzi-la com a mudança do conteúdo de uma posição de memória.

Evidentemente, em programas de jogos é que os ''sprites'' revelam melhor suas possibilidades, e um dos efeitos especialmente interessantes é o conhecido como ''detecção de colisão''. O equipamento é programado para, na ocorrência da sobreposição de dois ou mais elementos (por exemplo, quando um míssil atinge uma espaçonave inimiga), saltar para uma outra parte do programa, criando gráficos que simulam uma explosão (e aumentam a contagem de pontos do jogador).

Para utilizar essas figuras é preciso criá-las previamente, de modo muito semelhante ao usado na elaboração de um novo caractere. As letras do alfabeto, os números e símbolos gráficos especiais estão armazenados no interior do computador em um chip chamado gerador de caracteres.

Os caracteres, como mencionamos anteriormente no curso, são em geral construídos por uma matriz de oito pontos por oito, ou ''pixels''. O tamanho máximo dos elementos varia de um equipamento para outro, mas terá vários caracteres de largura e altura. No microcomputador Commodore 64, por exemplo, o tamanho máximo é de 24 pixels de largura por 21 de altura.

O melhor meio de criar uma figura é desenhar um

Em planos diferentes
Pelo uso de "sprite graphics", imagens complexas podem ser construídas através da colocação dos elementos componentes em planos diferentes, que se sobrepõem uns aos outros. Embora os gráficos sejam vistos na tela do computador apenas como um plano único, a grande vantagem desse sistema está em fazer com que objetos em planos diferentes se movam de modo independente. O programador determina uma ordem de prioridade, e assim, quando dois deles se sobrepõem na tela, aquele com prioridade maior é visto na frente do outro, com um efeito tridimensional. Computadores com "sprite graphics" também prevêem a interrupção do programa sempre que duas figuras entrarem em contato, criando o efeito de explosão, para aumentar os pontos do jogador.

Dimensões da figura
Existem "sprite graphics" para microcomputadores como o Commodore 64, o Sord M5, o TI99/4A e a linha dos Atari.

As dezesseis cores normalmente existentes aumentam para 256 no Atari, pois cada uma se apresenta em dezesseis graus de "luminosidade".

Nos jogos, é interessante saber quando duas figuras se chocam — por exemplo, duas espaçonaves — e, para isso, é fornecido um dispositivo para detecção de colisão.

O efeito tridimensional é produzido pela colocação de cada figura em um plano diferente; quanto mais planos forem possíveis, melhor.
O tamanho máximo de cada figura é expresso em pixels. Elas são controladas de modo a expandir-se ou a reduzir-se e podem ser movimentadas por toda a tela.

Para simplificação do processo de elaboração de figuras, existem à venda softwares especializados.

MICROCOMPUTADOR	NÚMERO DE CORES	DETECÇÃO DE COLISÃO	PLANOS	TAMANHO DA FIGURA	SOFTWARE
Commodore 64	16	Sim	8	24×21	Sim
Atari 800	256	Sim	16	16×16	Sim
Texas TI99/4A	16	Sim	28	32×32	Sim
Sord M5	16	Sim	32	8×8	Não

quadriculado, no tamanho máximo, e produzir a forma desejada pelo preenchimento dos blocos apropriados. Já sabemos que os computadores operam pelo uso exclusivo dos dígitos 1 e 0: os quadros em preto são representados por dígitos 1 e os brancos por dígitos 0.

O computador controla a maior parte dos dados sob a forma de bytes (um conjunto de 8 bits). O manual de seu computador explicará como o quadriculado de pontos que constitui uma figura deve ser subdividido em grupos de oito. Cada byte, por sua vez, precisa ser convertido em um único número decimal, que varia de 0 a 255, para poder ser usado em um programa BASIC. Isto é feito multiplicando-se o dígito binário (bit) da extrema esquerda por 128, o seguinte por 64 e assim por diante. Os resultados são somados em seguida.

O conjunto resultante de números decimais define de modo completo o formato da figura. Esses números são colocados em posições da memória por um programa em linguagem BASIC; o procedimento exato vai variar de acordo com o equipamento utilizado. É necessário, então, instruir o computador quanto às posições de memória em que são encontradas as especificações para cada uma das figuras desejadas.

Todo o resto é agora obtido usando-se comandos simples, que especificam a posição efetiva de cada figura na tela, alteram sua cor, ampliam ou reduzem o tamanho e descobrem quando ocorre uma sobreposição.

Existem pacotes de software chamados "utilitários", à venda para a maioria dos equipamentos que podem produzir gráficos com animação. Esses softwares tornam o processo de criação da imagem menos monótono. Apresentam o quadriculado na tela e possibilitam a elaboração da imagem pelo simples movimento do cursor luminoso pelo quadriculado. Todo o procedimento aritmético é controlado automaticamente e os resultados são então distribuídos em seus bytes correspondentes. Finalmente, o quadriculado desaparece e a figura sozinha surge na tela, pronta para utilização.

Nasce uma figura

O melhor modo de construir uma figura é começar com uma folha de papel quadriculado. O objeto é desenhado preenchendo-se diversos quadros.

Em outro quadriculado, os quadros preenchidos são representados como dígitos 1 e os não preenchidos como dígitos 0 — as duas unidades com as quais os computadores operam.
E, como a memória do computador é dividida em bytes (com 8 bits cada), todo o quadriculado é dividido em grupos de oito quadros.

Cada grupo deve ser convertido em um único número decimal. O bit da extrema esquerda será multiplicado por 128, o seguinte por 64 e assim sucessivamente. Os resultados somados devem fornecer uma resposta de 0 a 255, que é então utilizada no programa em BASIC.

Faça suas previsões

As "folhas eletrônicas" são úteis em muitas situações. Elas podem ser empregadas, por exemplo, para elaborar planejamentos, orçamentos ou estimativas de custo.

Calcula-se que gerentes de empresas gastam cerca de 30% de seu tempo preparando orçamentos — uma atividade que sempre exige respostas a muitas questões do tipo "o que aconteceria se...". Tradicionalmente é utilizada para esse trabalho uma folha de papel dupla, do tamanho destas duas páginas. Essa folha é dividida em doze colunas ou mais, cada qual com o nome ou número de um mês e com todas as despesas relacionadas de um lado. Em cada coluna, as despesas de várias categorias são especificadas. Somadas as colunas, obtém-se o total de despesas por mês e somam-se os resultados para determinar o total durante o ano, que é apresentado na coluna correspondente ao ano inteiro.

Surgem inconvenientes quando se planeja gastar muito acima — ou, pior ainda, muito abaixo — das despesas efetivas, sendo necessário, então, retornar e alterar diversas parcelas e calcular novamente os totais de linhas e de colunas, de acordo com os novos dados.

Utilizando-se programas de folha eletrônica, tem-se a possibilidade de refazer os cálculos da folha inteira, toda vez que um único elemento for modificado. Alterando-se, por exemplo, o custo de transportes em janeiro, altera-se o total de despesas nesse mês, bem como o total geral na coluna correspondente — tudo isso ao toque de uma tecla. Não é de admirar que os pacotes de folha eletrônica sejam o software de maior venda no mundo.

Assim como a maioria das unidades de software comercial, os programas de folha eletrônica são desenvolvidos em "overlays", o que significa que nem todos os programas estão efetivamente presentes no equipamento durante todo o processo. Se você imaginar o programa dividido em sub-rotinas (ver p. 77), a sub-rotina não necessária à operação em andamento não será chamada do armazenamento auxiliar (disco, ou fita). Quando isto ocorrer, o sistema operacional sobreporá a nova sub-rotina à anterior (daí o nome overlay: over = sobre, lay = colocar). Como você pode imaginar, esse método de ampliação da memória disponível é muito útil, mas também significa que muitas vezes será necessário esperar pela transferência dos dados — da memória auxiliar de armazenamento de dados para a principal.

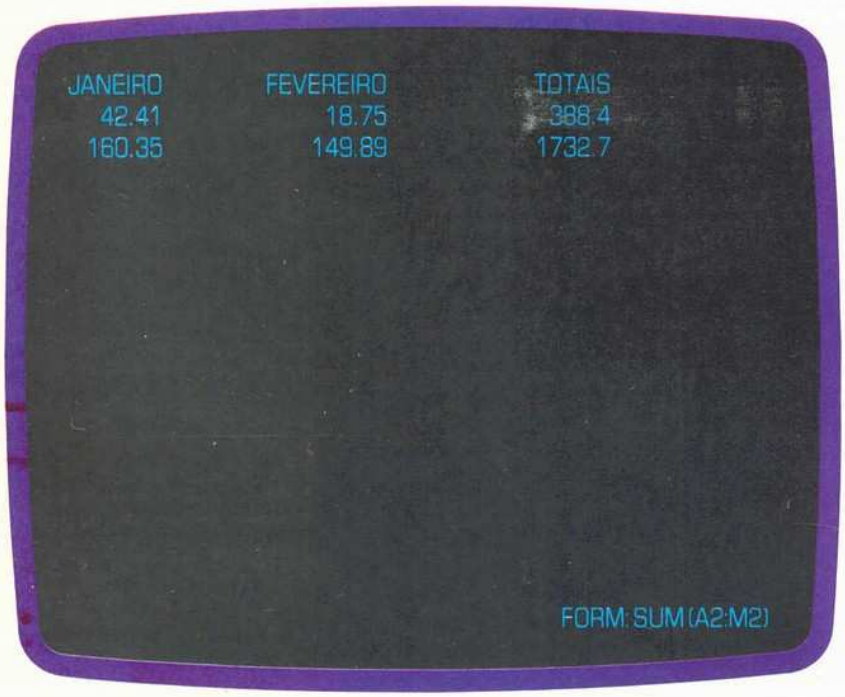
Os pacotes de folha eletrônica — que você facilmente reconhecerá pelo nome, com freqüência terminado em "calo" — existem em versões compatíveis com uma grande variedade de microcomputadores para uso doméstico e empresarial. O mais vendido é o Visicalc, originalmente escrito para o equipamento Apple II e colocado no mercado em meados de 1979. O mundo do software de microcomputação não é lento em acompanhar o sucesso de um de seus elementos e este caso não foi exceção. Imediatamente as folhas eletrônicas já estavam em toda parte, com variações compatíveis com todos os tipos de equipamento.

Para ser eficiente, a folha eletrônica deve ter duas qualidades: tamanho (não necessariamente o apresentado na tela, porque, como verificaremos mais adiante, você poderá ver apenas uma "janela" do todo) e boa variedade de comandos de controle e de formatação.

Isso significa que há sérias limitações nas máquinas que podem processar esse tipo de programa com o máximo de rendimento. Como regra prática, 32 Kbytes de RAM e uma tela de oitenta caracteres são requisitos mínimos para a utilização empresarial, embora uma tela de quarenta caracteres provavelmente seja suficiente para fins pessoais.

Quem utiliza o computador em casa verifica que muitos dos pacotes adequados a equipamentos com dispositivos para cassete, como o TK85 ou CP 200, embora naturalmente limitados em tamanho e capacidade, são bastante úteis.

Por ter capacidade de resolver questões do tipo "qual o impacto de...", os programas de folha eletrônica podem, obviamente, ser usados para montar modelos computadorizados simples. Na página 101 apresentamos um exemplo do trabalho de um



Campos de dados
A folha eletrônica está dividida em linhas e colunas e a intersecção de linha e coluna é chamada "campo" ou "célula". Cada célula pode conter um título (por exemplo, JANEIRO), um número (por exemplo, 149, 89) ou uma fórmula. O campo N2 contém a fórmula "SUM (A2:M2)", que representa a soma dos números da linha superior, de janeiro a dezembro. O resultado desse cálculo é exibido: 388,4. Observe que a folha foi dividida em duas janelas, de modo que os meses de março a dezembro não estão visíveis na tela.

A linha inferior
O cursor é o bloco retangular que em geral ocupa o campo N2. O caractere digitado aparecerá no campo em que o cursor está posicionado. Os conteúdos totais desse campo serão igualmente apresentados na linha de comando do programa que, neste caso, se encontra na parte inferior da página.

analista de sistemas e, caso você venha a usar um pacote de folha eletrônica, logo se tornará evidente a necessidade de um planejamento cuidadoso semelhante. Os programas de banco de dados, conforme já verificamos, consistem em quantidades maciças de dados, que são organizados de acordo com as exigências do usuário. Os programas processadores de palavras, outro software de grande vendagem, são projetados para permitir o deslocamento de palavras isoladas ou blocos inteiros de texto, ao serem processados. Porém, os programas de folha eletrônica são um pouco diferentes porque exigem efetivamente que o usuário realize um processo de planejamento.

Por exemplo, se você utilizar um programa de folha eletrônica para analisar suas despesas gerais, poderá agrupar prestações da casa ou hipoteca, impostos, seguros etc. e depois utilizar o resultado do cálculo na tabela maior, incluída na mesma folha. Você deve tomar o cuidado de somar primeiramente todas as despesas domésticas antes de transportar qualquer número para a tabela maior.

Cada posição individual na folha, chamada célula, é endereçada e localizada através de suas coordenadas X e Y. Horizontalmente são usadas as letras de A a Z, de AA a AZ e, em alguns casos, de BA a BM, o que permite dar conta da extensão total possível da folha — 65 colunas nas versões mais populares. Verticalmente, podem ser usados os números de 0 a 256. Cada célula pode ter um título (como "Vendas" ou "Lucros"), um valor, fornecido ou derivado de um cálculo, ou a fórmula desse cálculo, como B4+B6*B5. Muitas vezes, as fórmulas ultrapassam o tamanho do quadro visto na tela, e são, de modo geral, apresentadas em uma linha separada na parte superior da tela. Ao iniciar um novo programa, o tamanho da célula será preestabelecido, talvez em oito ou nove dígitos ou caracteres. Este é o chamado tamanho default. Geralmente, você terá a possibilidade de reduzir ou aumentar as células, para adaptá-las ao tipo de cálculo que vai realizar. Alguns pacotes permitirão que a coluna da extrema esquerda (títulos ou itens que você mesmo escolhe) seja maior que as demais. E não é necessário decidir de imediato o tamanho desejado. A maioria das versões admite ampliação ou redução, mesmo quando já houver dados nelas incluídos. Caso venha a reduzir para um tamanho menor que o do conteúdo, a parte não apresentada na tela não será apagada, apenas não será mostrada.

O último componente importante na folha é a linha de comando, que se apresenta na parte superior ou inferior da tela em resposta à tecla de "comando": /, por exemplo. Esses comandos são destinados ao uso em formatação e ao controle da disposição das unidades, e não dos dados, embora possam afetar o modo pelo qual estes aparecem. A maioria das versões populares desse software possibilita a realização de grande variedade de operações com banco de dados. Você pode, por exemplo, apagar, deslocar ou copiar linhas ou colunas inteiras; ou dividir a janela de modo que exiba conjuntamente as partes da folha que em geral estão longe demais para que a janela as inclua ao mesmo tempo; ou, ainda, deslocar essas folhas separadamente por toda a tela.

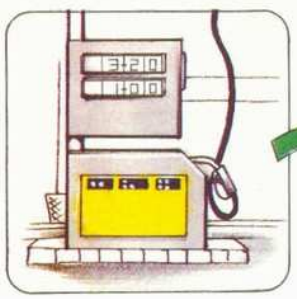
Geralmente, o deslocamento entre as células é realizado pela tecla de controle do cursor, mas há ainda outra tecla de comando que permite saltar para uma célula específica. Os procedimentos de armazenamento e retenção, limpeza e proteção são realizados por teclas de comando e, já que estamos falando nisso, salientamos novamente a importância de sempre guardar os próprios programas. A montagem de uma folha eletrônica toma um longo tempo, maior do que para fornecer os dados. Como regra geral, sempre retenha a folha, antes de começar a alimentá-la de informações. E, se cometer algum erro grave, você terá minimizado as perdas.

Os resultados são enviados à impressora por uma tecla de comando, e deve-se tomar cuidado em definir qual a parte da folha que se deseja imprimir, usando parâmetros. Assim como a tela é semelhante a uma janela, ou porção da folha inteira, assim também, evidentemente, será a página da impressora. Se você necessitar imprimir uma folha maior do que a da impressora, o procedimento recomendado é fazer duas impressões e, em seguida, colar as páginas de modo conveniente.

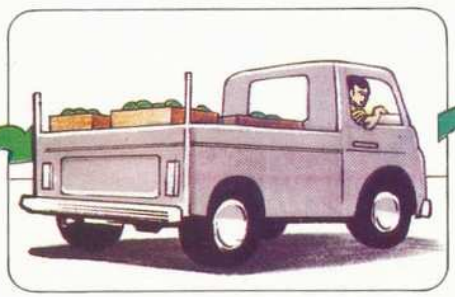
O uso de janelas, como observamos, permite que duas partes da folha sejam exibidas ao mesmo tempo. Uma folha com janela dividida também pode ser impressa. Isso é especialmente útil quando se fornecem dados, porque se pode fazer referência a itens anteriormente incluídos. A maioria dos pacotes de folha eletrônica permite ao usuário "manter" a parte superior ou as primeiras linhas, o que é conveniente pelas mesmas razões, pois estas costumam conter os títulos.

Até agora, consideramos a folha como uma tabela acessada unicamente de modo serial (um item após o outro, nas linhas e nas colunas); se quisermos, porém, abandonar a comodidade de efetuar as somas pela disposição em linhas e colunas, pode-se

E se...?
Se o conteúdo de um campo for alterado, o programa de folha eletrônica automaticamente refaz os cálculos dos outros campos que de algum modo dependem desse número. A rapidez e facilidade desse processo incentiva o usuário a testar suas estimativas e planejamentos, para verificar o que acontecerá, digamos, ao lucro global, caso certas condições se alterem. Veja, por exemplo, o vendedor de frutas...



Se o preço da gasolina aumentar X%...



Os custos mensais do transporte subirão Y%...

Isso acarretará o aumento do preço dos produtos por atacado...

Que será transferido ao consumidor, com preço final mais alto...

dar à folha qualquer outra disposição, se isso facilitar a linha de pensamento do usuário. Todavia, tenha em mente que essa modificação no interior do banco de dados exigirá um trabalho de análise de sistemas ainda mais completo.

As folhas eletrônicas com finalidade empresarial, como o Visicalc e o Supercalc, proporcionam ao usuário a possibilidade de transferir dados para os pacotes processadores de palavras ou de banco de dados, e existem muitos programas complementares que possibilitam a saída sob várias formas gráficas: gráficos em curvas ou barras, por exemplo.

Já tratamos de uma das utilizações da folha eletrônica: a elaboração de orçamentos e a análise de despesas domésticas. Outra aplicação muito interessante é a instalação de um sistema central de refrigeração, em que grande número de variáveis deve ser levado em consideração: energia a ser consumida, a quantidade e o tipo de equipamento, a área a ser refrigerada etc. De fato, qualquer tomada de decisões é consideravelmente auxiliada por um programa de folha eletrônica, pois o usuário é quase forçado a levar em conta qualquer possibilidade.

Talvez o aspecto mais notável do programa de folha eletrônica com finalidades empresariais, processado em equipamento adequado, seja a rapidez da operação. Esta é função direta da programação em código de máquina e não é surpreendente que a velocidade de um pacote desenvolvido em linguagem BASIC para alguns pequenos microcomputadores seja bem menor.

Será interessante examinar alguns problemas que poderíamos encontrar se tentássemos desenvolver tais programas em BASIC, apenas para dar uma idéia da complexidade da tarefa.

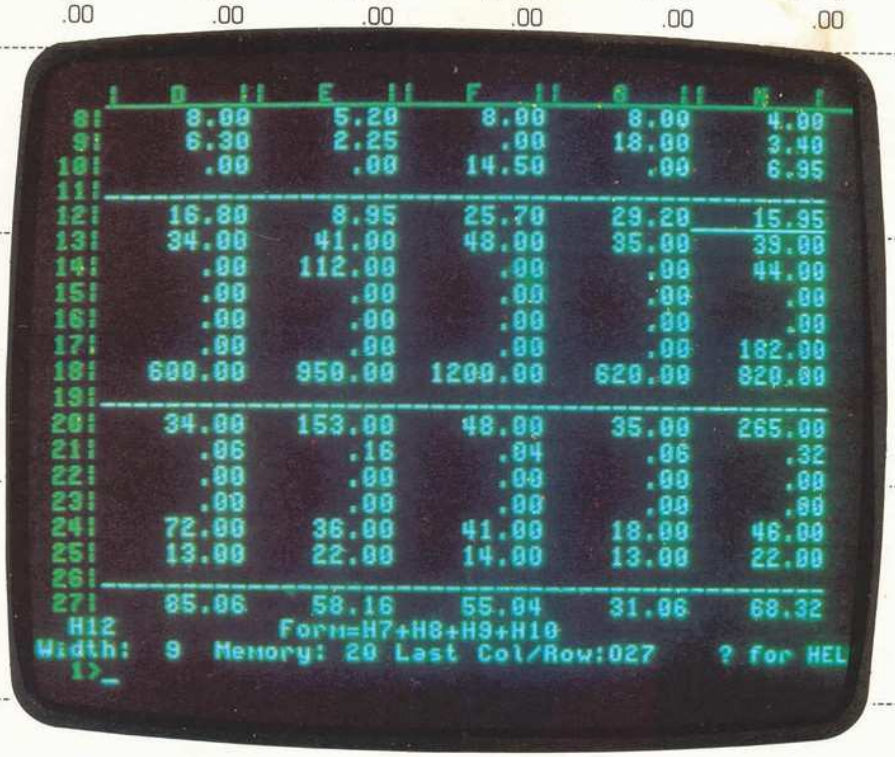
Para começar, cada célula deve ser definida de três modos: deve ser capaz de conter dados de "variáveis alfanuméricas", como "janeiro" ou "impostos"; deve ser capaz de conter dados numéricos para utilização em operações aritméticas; por exemplo, o total de impostos pagos em janeiro; e também deve poder conter fórmulas, que são basicamente linhas em código de programação, como "impostos anuais 12", para se obter a média mensal. Por isso, cada célula deve ter seu tamanho expandido ou reduzido, sem perder a menor parte de informação significativa; por esta razão, todas devem ser duplicadas: uma aparecerá na tela e outra, oculta, conterá todos os dados.

Como você vê, a simples manipulação dos dados é uma tarefa complexa, e lembre-se de que os mais sofisticados pacotes podem ter 16.000 células individuais! As técnicas usadas para desenvolver tal software são muito semelhantes às de desenvolvimento de interpretadores para linguagens, como BASIC e FORTH. Técnicas semelhantes são também utilizadas nos softwares de banco de dados.

Tudo isso leva à explicação do alto preço de softwares desenvolvidos com finalidades empresariais. Pacotes como o Visicalc ou o Supercalc podem ser considerados caros, mas deve-se ter em mente a economia que seu uso proporciona. O exemplo relativamente simples que examinamos anteriormente, da utilização desse tipo de software em substituição ao trabalho do gerente na elaboração de orçamentos, pode levar a uma economia de 15 a 20% ao ano. O custo do software será, então, insignificante, comparado a tal economia. Na verdade, o Visicalc talvez tenha sido o melhor amigo dos vendedores dos equipamentos da linha Apple.

Janelas do mundo
Podemos mover o cursor pela tela, para a direita ou a esquerda, para cima ou para baixo, pelo controle do teclado; assim também, em um programa de folha eletrônica movemos a tela por toda a folha. Isto permite que uma área muito maior do que a da tela seja examinada.

1: TÍTULOS	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO
2: ALUGUEL	180.00	180.00	180.00	192.00	192.00	192.00	192.00	192.00
3: IMPOSTO	42.00	42.00	42.00	51.00	51.00	51.00	51.00	51.00
4: SEGURO	.00	.00	.00	.00	.00	.00	.00	.00
5: TOTAL								
6: DESPESA	222.00							243.00
7: TRANSP.	2.00							.00
8: REFEI.	6.00							.00
9: ROUPAS	14.00							.00
10: LIVROS	.00							.00
11: TOTAL								
12: ESCOLA	22.00							.00
13: COMBUS.	42.00							92.00
14: MANUT.	.00							.00
15: OFICINA	26.00							18.00
16: IMPOSTO	.00							.00
17: SEGURO	.00							.00
18: TRU	100.00							90.00
19: TOTAL								
20: CARRO	68.00							110.00
21: PEDÁGIO	.00							.00
22: AVIÃO	190.00							74.00
23: HOTEL	86.00							340.00
24: REFEI.	22.00							190.00
25: DIVER.	12.00							38.00
26: TOTAL								
27: LAZER	378.00							752.00



Quando o herói é você

Os jogos eletrônicos de aventura estão cada vez mais difundidos. Neles você não é só um espectador, mas um participante ativo.

Aventura: uma palavra que faz recordar um livro, um filme, uma viagem ou, talvez, uma experiência pessoal. Para muitas pessoas, usuárias de computadores, aventura também lembra um determinado tipo de jogo eletrônico.

Para caracterizar melhor o que são as aventuras de computadores, vamos compará-las a um livro. Quando você lê uma história de aventuras, pode empolgar-se com os perigos, os mistérios e uma série de episódios interessantes — mas, na verdade, tudo isso é "vivido" por outra pessoa. Numa aventura de computador, você não fica de fora da ação, mas participa dela! Como o herói ou heróis da história, você se liga nos acontecimentos, vive diretamente as experiências.

Num livro, o leitor não pode influenciar o curso do enredo. A ordem e o desfecho da trama são sempre iguais, não importa quantas leituras sejam feitas. Numa aventura de computador, suas decisões, julgamentos e ações é que determinam o desdobramento e a modificação de uma trama. Pode haver qualquer número de variações para a ordem dos acontecimentos, além de muitos finais diferentes, alguns alegres, outros tristes. O que existe de emocionante nos jogos de aventura é o fato de você participar da ação, apesar de estar sentado confortavelmente em sua casa.

Cada aventura ocorre num ambiente diferente, que pode ser um mundo subterrâneo, um labirinto, uma cidade fantasma, um outro planeta, uma terra encantada; também pode acontecer em qualquer época: passado, presente ou futuro.

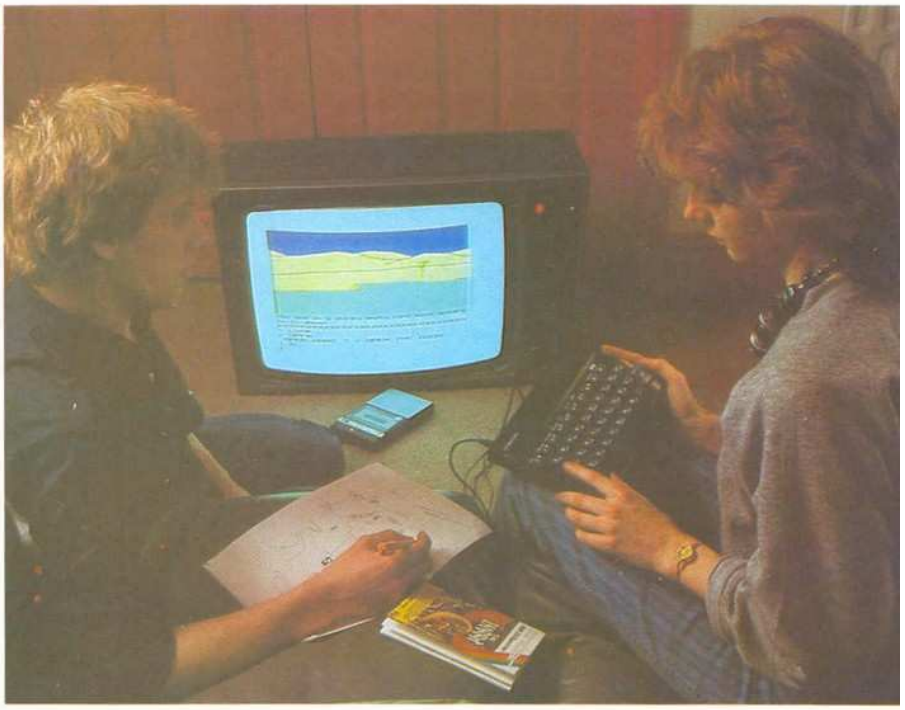
Geralmente, a aventura tem um tema constante que inclui um objetivo final. Por exemplo, você deve fugir de um planeta estranho, ou encontrar e destruir um feiticeiro malvado, salvar a princesa, pegar um tesouro, descobrir o motivo de um crime e prender o culpado.

Incidentes interessantes como esses fazem com que a aventura se torne um quebra-cabeça agradável e o enigma seja uma parte constante do jogo. Com freqüência, tem de ser resolvido antes que a confusão aumente. Há também as situações imprevistas; por exemplo, depois de atravessar um desfiladeiro para agarrar a figura estranha que o observa há muito tempo, você descobre que era apenas um enorme espelho que refletia sua própria imagem.

No decorrer do jogo surgem pistas ocasionais que ajudam mas não são essenciais para a solução do mistério — a descoberta de uma passagem secreta pode levar a um lugar muito perigoso. É uma questão de vida ou morte — você está totalmente perdido, sem comida e sem água.

Esses percalços podem ser decifrados usando o bom senso, o que não exige nenhuma experiência ou conhecimento especializado. Entretanto, o aventureiro deve ficar atento, pois sempre surge uma nova pista. Os elementos casuais, exceto em doses pequenas, não aparecem num programa de aventura bem elaborado.

No decorrer de um jogo, é provável que você encontre objetos, mensagens ou personagens aparentemente insignificantes. Entretanto, tenha em mente que tudo é relevante numa história de aventura; para cada detalhe há um propósito, mesmo que seja apenas tirar você da pista certa.



O que significam várias garrafas quebradas? O que pensar ao ouvir uma voz cavernosa que diz BAH? Como alguém pode utilizar um monte de lama suja? Como um tapete pode ser pregado no chão se não existe chão? Esses enigmas são componentes essenciais do enredo da aventura. Ao encontrar um objeto pela primeira vez, não importa que seja estranho ou inútil, dificilmente você descobrirá logo sua importância; mas com certeza precisará dele até o final da história.

Muitas aventuras têm um pequeno labirinto onde cada sala ou lugar é descrito em termos idênticos. O único modo seguro de traçar uma trilha nesses labirintos é fazer como João e Maria, deixando obje-

Você vive o papel
Jogos de aventura como os de masmorras e abismos existem há muito tempo; mas o microcomputador levou-os a uma camada maior da população. Na Europa e nos Estados Unidos, os jogos de aventura tornaram-se tão populares como os fliperamas. Apesar de poderem ser jogadas por uma só pessoa, as aventuras são um divertimento para toda a família.
O personagem fictício ou histórico é assumido pelo jogador, que viaja em busca de um tesouro ou de outro objetivo.

Leitura do mapa
Num jogo de aventura, o jogador ou o personagem que ele representa tem de agir dentro de um vasto terreno ou outro ambiente — uma rede de passagens subterrâneas é algo bastante comum. Alguns fabricantes fornecem sugestões num manual de instruções para os que não têm muita experiência nesse tipo de jogo, mas o melhor meio de obter as soluções é a própria experiência na aventura. Esses jogos podem levar semanas para terminar; sendo assim, é importante traçar um resumo dos lugares e obstáculos que você já ultrapassou.

tos pelo caminho que percorrer. Este método tornou-se tão conhecido que alguns programas já acrescentaram outros problemas, tal como a presença de um bandido que persegue o jogador e vai apagando os vestígios.

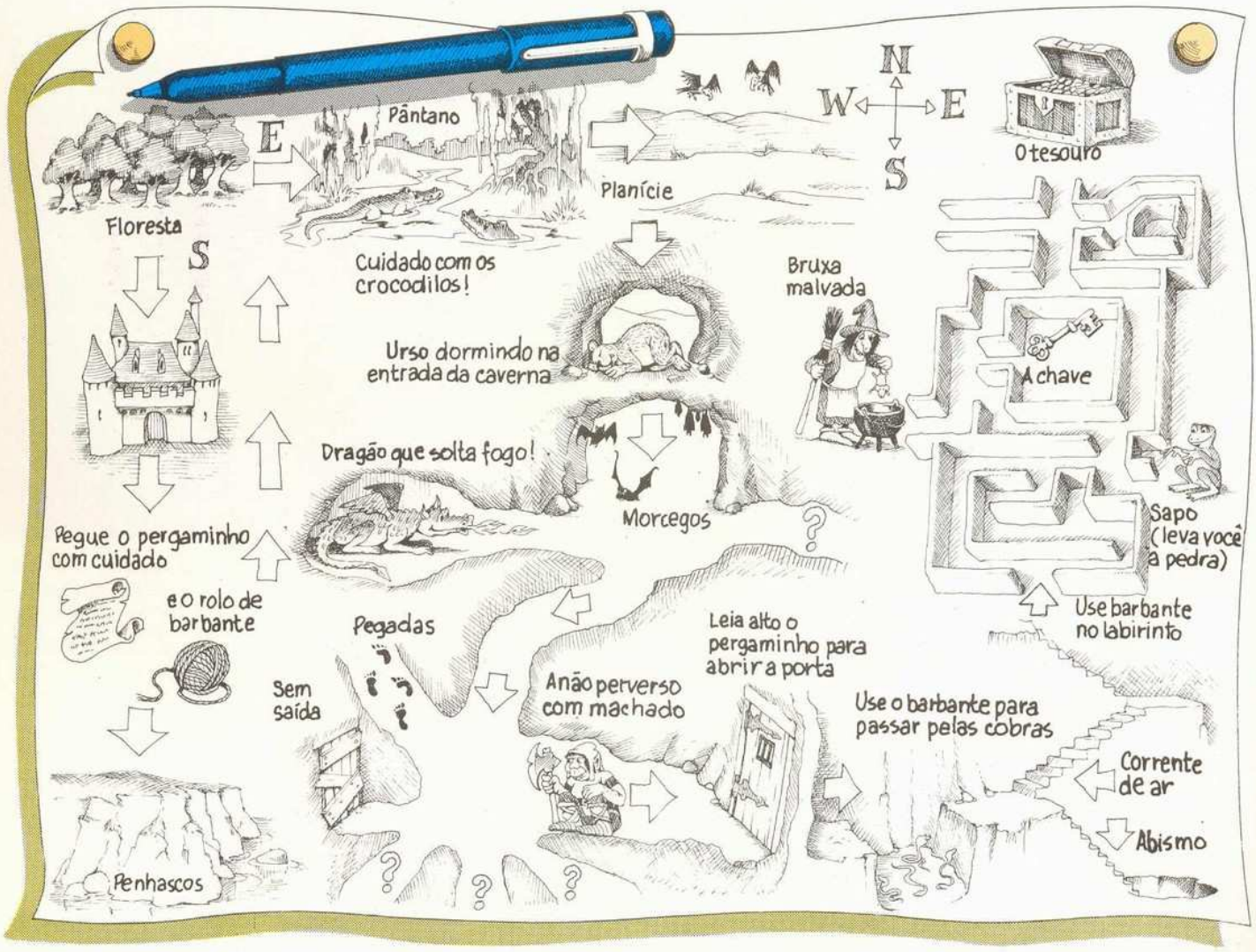
Em algumas aventuras, apesar de você ter de solucionar os enigmas e alcançar todos os objetivos para completar o jogo com sucesso, a ordem em que os mistérios são desvendados e as metas alcançadas não tem grande importância. Isto contrasta com aquele tipo de aventura onde há apenas uma trilha para a conclusão triunfante.

Uma boa aventura deve durar horas, talvez semanas de jogo, até que todos os segredos sejam revelados. Deve, ainda, possibilitar que você pare de jogar no ponto que quiser, no cassete ou no disco, e continue mais tarde. Isso também é bastante útil quando você atinge uma parte perigosa da aventura; por exemplo, indo corajosamente ao encontro de um exército, mesmo tendo apenas uma lanterna e um cantil. O jogador aventureiro mais prudente poupará o tempo de jogo antes de iniciar um diálogo com os soldados do exército. Então, caso o exército o convide para jantar, o jogo será recomeçado e o aventureiro pode procurar um novo curso de ação; isto, se ele não ficar bastante aborrecido por ser a refeição escolhida pelo inimigo.

A morte do aventureiro não significa necessariamente o fim do jogo. Alguns programas permitem que o perdedor "ressuscite", em geral por meio de um sopro de fumaça amarela. Ao mesmo tempo provoca a perda de alguns pontos e posições conquistadas e coloca-o numa situação indesejável, digamos, num mundo inferior ou no meio de "lugar nenhum".

Como você participa e se relaciona com o programa? Ele pode estabelecer uma comunicação direta com você, ou utilizar um "boneco", um personagem controlado por comandos. O computador atua como intérprete e narrador de seus desejos. O jogador registra os comandos no teclado e as respostas do computador aparecem na tela.

Algumas aventuras mostram apenas textos na tela do computador, outras só apresentam gráficos, e há as que combinam os dois. Efeitos sonoros são utilizados nas aventuras elaboradas por meio de gráfico. Os jogos de aventura que só trazem um texto são como um livro de histórias não ilustrado; as palavras descrevem locais, objetos e acontecimentos. Os gráficos, na maioria dos jogos combinados, servem



para complementar as descrições textuais, e geralmente são constituídos de cenas estáticas de locais e objetos. Variam de uma simples linha desenhada até figuras bem detalhadas. Nas aventuras baseadas em gráficos, estes normalmente são usados em mapas estilizados, figuras de camadas da terra ou em representações de interiores de edifícios. Os personagens e os objetos aparecem sob a forma de símbolos ou figuras em miniatura. Neste caso, o jogador restringe-se a um pequeno número de comandos que, na maior parte, se realizam por meio de pressões no teclado do computador, a fim de mover e controlar o personagem.

O texto numa aventura geralmente abrange três elementos: onde você está, o que você pode ver e aonde pode ir. Por exemplo, o texto que aparece na tela pode ser: ''Você se encontra numa floresta escura. O céu está encoberto por densa folhagem. Existe uma trilha que leva a leste e a oeste. Mais à frente, para o norte, há uma armadilha perigosa. Pode-se ver uma espada no chão: em volta dela, uma cobra verde''. Você recebe uma descrição do que está mais próximo, algumas direções que pode seguir e os objetos que aparecem no caminho.

Os comandos quase sempre consistem em duas palavras: um verbo seguido de um substantivo, embora as aventuras mais sofisticadas compreendam frases inteiras. Alguns verbos são: PEGAR, SOLTAR, EMPURRAR, PUXAR, JOGAR, ACENDER, MATAR, COMER e BEBER, IR PARA O NORTE. Cada um deles pode especificar um movimento, apesar de algumas aventuras utilizarem apenas a abreviatura desses comandos — por exemplo, N para NORTE.

EXAMINAR é um verbo essencial — geralmente significa que o aventureiro deve buscar mais informações. EXAMINAR COBRA na ilustração (à esquerda) pode resultar numa mensagem como ''É uma cobra de jardim'', ou talvez ''A cobra percebe seu movimento na direção dela e vai atacá-lo''. Alguns verbos não precisam de substantivo.

A palavra INVENTORY (listagem) é empregada para informar o que você está carregando. Alguns objetos podem ser colocados dentro de outros — água em garrafa e machado numa sacola; enquanto outros devem ser usados diretamente no corpo — um anel, talvez, ou uma capa.

A palavra SCORE é utilizada para que o jogador saiba o quanto ele já progrediu em relação aos objetivos que deseja alcançar. A palavra HELP pode fazer com que você supere mais facilmente uma dificuldade, e, ao mesmo tempo, funciona como um incentivo para que continue tentando. Por vezes, como resposta a um comando, o aventureiro pode receber um ''Não faça isso ainda!'', o que mostra que a combinação de um verbo com um substantivo dará algum resultado, mas não naquele momento nem, talvez, naquele local. Grande parte do desafio está em descobrir os verbos e os substantivos que são importantes para a aventura. Palavras e combinações que a aventura não reconhece recebem uma resposta do tipo ''Eu não compreendi você''.

A maior parte dos produtores de aventuras fornece folhas que contêm dicas para aqueles que se confundem no jogo e também incluem algumas sugestões sutis.

Algumas aventuras são muito longas para a memória do computador. Para que isso não represente um problema, elas são muitas vezes fornecidas em discos, onde o programa principal é carregado na memória no início, e partes do texto ou dos gráficos são colocadas ou tiradas do disco, de acordo com a necessidade.

Mas, graças às técnicas de condensação de grandes textos em memórias restritas e ao uso da programação em código de máquina, uma aventura longa pode ser armazenada na memória do computador, e, conseqüentemente, gravada numa fita cassete ou num disco flexível.

Existem aventuras para quase todos os tipos de micro. E qualquer uma delas proporcionará uma experiência agradável e será até mesmo o começo de um passatempo habitual. Por isso teste seu raciocínio e divirta-se. Feliz aventura!

Caça ao tesouro

O Hobbit é uma aventura gráfica para o Sinclair Spectrum e leva você à história de J. R. R. Tolkien. Uma cópia dessa história acompanha a fita cassete. Você é Bilbo, e faz uma viagem ao centro da Terra, onde encontra muitos dos personagens e acontecimentos do livro, enquanto procura o dragão e o tesouro.

Na pista certa
Deadline é uma variação dos temas de aventura. Você faz o papel de um detetive que vai solucionar o mistério de um assassinato. Seu arquivo contém o resultado de uma autópsia, algumas pílulas encontradas perto do cadáver e outras anotações sobre o caso. O jogo não ocorre num só dia (o espaço de algumas semanas é normal), mas cada nova ação, tal como movimentar-se de um local para outro, ou a entrevista com um novo suspeito, faz você perder minutos reais do seu prazo limite de doze horas.

Tradução alternativa

Computadores "pensam" em código de máquina. Programadores preferem linguagens como o BASIC ou PASCAL. E os compiladores e interpretadores fazem a tradução.

Os computadores não tinham teclado no início de seu desenvolvimento. As instruções dos programas eram fornecidas pelo ajuste de cada um dos oito interruptores "para cima" ou "para baixo", a fim de representar uma única operação. Essas combinações de interruptores constituíam um código de máquina.

A substituição dos interruptores por um teclado semelhante ao de máquina de escrever e das combinações de posição dos interruptores por palavras da linguagem comum foi uma conseqüência lógica, da qual resultaram as linguagens de alto nível, como BASIC, que substituíram os códigos de máquina de baixo nível.

Como processadores, todavia, os computadores não se modificaram; ao contrário, continuaram a funcionar no antigo sistema de interruptores (e ainda o fazem); desse modo, os programadores tinham de desenvolver programas na notação original de baixo nível, a fim de traduzir os de alto nível em combinações que os processadores poderiam operar. Esses programas de baixo nível tornaram-se conhecidos como interpretadores e compiladores, conforme seu método de tradução.

Em computação (como em tudo o mais), qualquer aumento em capacidade ou velocidade tem um custo — em dinheiro, tempo ou liberdade de ação. O mesmo acontece com os interpretadores e compiladores. Juntos, proporcionam ao programador todos os recursos para a tradução de programas. Os interpretadores oferecem vantagens em alguns aspectos e os compiladores em outros, mas também apresentam desvantagens.

Os interpretadores, geralmente incorporados ao microcomputador, são um método barato de tradução de programas de linguagem de alto nível para uma linguagem que o computador possa compreender. Não utilizam muita memória, o que deixa mais espaço para os programas.

A maioria dos micros possui um interpretador em linguagem BASIC: você dá entrada em um programa em BASIC, digita o comando RUN e o programa funciona ou se interrompe pela apresentação de uma mensagem de erro do sistema, semelhante a:

```
SYNTAX ERROR ON LINE 123
```

Com isto, você deve digitar a tecla LIST, detectar o erro, corrigi-lo e o programa funciona ou se interrompe, e assim por diante. Observe que alguns dos mais sofisticados interpretadores BASIC verificam realmente erros sintáticos à medida que cada linha é fornecida.

Talvez isso tenha ocorrido centenas de vezes sem que você imaginasse o funcionamento do interpretador. Sua principal qualidade está no fato de ser um dispositivo oculto, que permite ao usuário operar o programa sem se preocupar com sua localização na memória, ou como executá-lo: o programa encontra-se ao alcance de seus dedos e você pode imediatamente teclar RUN, LIST ou EDIT.

O interpretador é de uso fácil e não muito sofisticado: quando você digita o comando RUN, o interpretador deve localizar o programa em linguagem BASIC na memória, traduzi-lo e executá-lo, linha por linha. Se seu programa contiver o seguinte loop:

```
400 LET N = 0
500 PRINT N
600 LET N = N + 1
700 IF N < 100 THEN GOTO 500
```

o interpretador deverá traduzir e executar cem vezes as linhas de 500 a 700, como se jamais as tivesse encontrado antes.

Os compiladores são diferentes: caros, difíceis de ser escritos, ocupam e utilizam muita capacidade de memória. Quase sempre funcionam com software em disco; desse modo, o usuário necessita de equipamento complementar dispendioso.

A vantagem que proporcionam está na flexibilidade, alta capacidade e velocidade; diante das quatro linhas de programa em linguagem BASIC acima, o compilador as traduzirá, todas de uma vez, e em seguida executará o código resultante cem vezes.

Isto permite boa economia de tempo. Admitamos que você tenha um compilador em linguagem BASIC e queira processar um programa em BASIC.

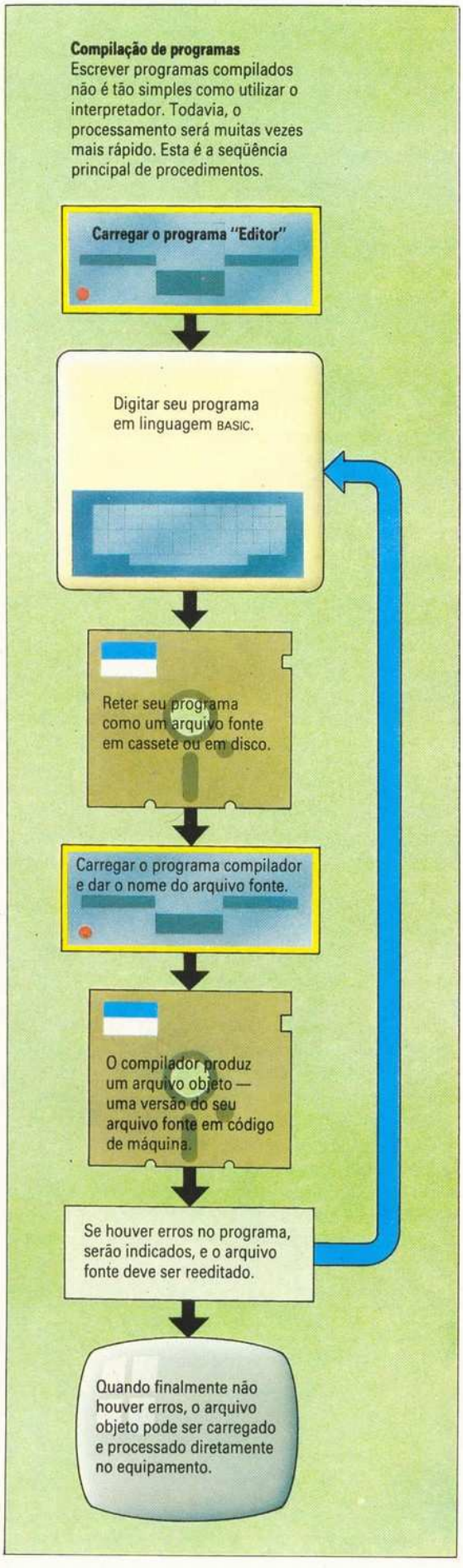
Primeiramente, você deve carregar um programa de criação de arquivos (chamado Editor), que lhe permite digitar o programa e retê-lo em disco, como um "arquivo fonte".

Os arquivos devem receber denominações que permitam sua localização (exatamente como os arquivos de um escritório); assim, o Editor solicita a digitação do nome do arquivo fonte. Os nomes de arquivo têm em geral duas partes: a primeira consiste em um nome qualquer que você escolhe — digamos, MEUPROG — e a segunda, em um código de três letras, que indica o conteúdo do arquivo; este código é a "extensão". Um arquivo em BASIC possui o código BAS como extensão. Seu arquivo fonte está agora no disco, sob o nome MEUPROG.BAS. Agora a digitação de

```
COMPILE MEUPROG.BAS
```

induz o computador a carregar e processar o compilador, utilizando um arquivo fonte em BASIC chamado MEUPROG.BAS.

Você aguarda alguns segundos, de acordo com a extensão de seu programa, enquanto o compilador o traduz em um "arquivo objeto", que é retido em



Compilação de programas
Escrever programas compilados não é tão simples como utilizar o interpretador. Todavia, o processamento será muitas vezes mais rápido. Esta é a seqüência principal de procedimentos.

disco, sob o nome de MEUPROG.OBJ — a extensão OBJ indica que este é o arquivo objeto, uma tradução para o código de máquina de um arquivo fonte.

Enquanto traduz o arquivo, o compilador verifica possíveis erros sintáticos. Ao encontrar algum, emite uma mensagem semelhante às seguintes:

```
100 REED X:IF X=3(N+2) LET P=Q
    1              2     3
FATAL ERROR:-
1) //REED// UNRECOGNISED COMMAND
2) //(// ILLEGAL OPERATOR HERE
3) ??"THEN" OR "GOTO" EXPECTED HERE
```

Você receberá este tipo de mensagem para cada linha que contiver erro. Ou seja, a indicação de erro é muito mais abrangente do que em um interpretador BASIC. Então você deve carregar e processar novamente o Editor, chamar outra vez o arquivo fonte retido no disco, efetuar as alterações e tentar nova compilação. Se não mais restarem erros, digite

```
RUN MEUPROG
```

e ele funcionará ou não como o esperado. Nesta fase, corrigidos os erros sintáticos, você pode, ainda, querer alterar o programa; deverá então carregar e processar o Editor, modificar o arquivo fonte, recompilá-lo, e assim sucessivamente.

O bom desempenho do compilador não é percebido de modo claro no estágio de desenvolvimento do programa, embora a informação de erros tenha muita utilidade. O valor dos compiladores se evidencia quando você está com um programa sem mais correções a fazer e digita o comando RUN.

Os programas compilados são rápidos — cinco a cinqüenta vezes mais que programas interpretados, conforme a eficiência do compilador; mas a maior velocidade de execução de programas compilados obtém-se à custa da menor velocidade no desenvolvimento do programa.

A comparação entre compiladores e interpretadores pelo contraste de seqüências habituais de comandos para o usuário, como as acima, não faz justiça aos compiladores, já que são projetados visando principalmente a equipamentos menos especializados e de maior porte, nos quais os usuários podem ter necessidade de escrever e processar programas em muitas linguagens diferentes.

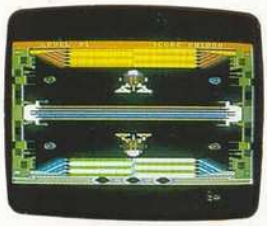
A linguagem COBOL (para escrever programas comerciais, folhas de pagamento e estoques), por exemplo, foi inventada tendo em vista a compilação, enquanto o BASIC efetivamente foi desenvolvido para ser interpretado. Se quiser comparar dois veículos diferentes, você deverá testá-los tanto em estrada de terra como na asfaltada.

Uma vez elaborado e compilado o programa, você não precisará do arquivo fonte, a não ser para referência. Assim, o programa fonte pode ser comentado e desenvolvido objetivando a facilidade de leitura, enquanto o arquivo objeto é muito menor, ocupando menos espaço em disco e memória.

O fato de um arquivo objeto criado pelo compilador consistir em código de máquina ilegível é, surpreendentemente, uma vantagem. Se você for um vendedor de software, não venderá o arquivo fonte, apenas o arquivo objeto, mais difícil de plagiar, copiar ou alterar.

Lento
Em jogos de aventuras, a velocidade não é fundamental e a maior parte do programa consiste na manipulação de texto. Por isso, o programa pode ser desenvolvido em BASIC e interpretado à medida que é processado.

Mais rápido
Muitos programas contábeis são difíceis de desenvolver em código de máquina pois implicam grande quantidade de procedimento. Todavia, linguagens interpretadas seriam muito lentas; desse modo, esses programas geralmente são desenvolvidos em BASIC e então compilados.



Rapidíssimo
Para jogos de ação rápida, tipo fliperama, que envolvem a manipulação de gráficos, mesmo programas compilados não seriam suficientemente rápidos. Tais pacotes devem ser escritos diretamente em código de máquina — uma tarefa lenta e cansativa.

Piratas à vista

A pirataria na indústria do software é um sério problema. E muitas são as tentativas dos fornecedores para solucioná-lo.

Gravador "escravo"
Cassetes de software, assim como cassetes de música, são duplicados usando-se uma copiadora de alta velocidade. Este aparelho consiste em um deck principal, onde é colocado o original, e algumas unidades "escravas", que gravam simultaneamente. Copiar os dois lados de um programa em cassete é uma questão de segundos, ao passo que discos têm de ser copiados individualmente, usando-se unidades de disco normais.

Cem por um
Assim como é ilegal reproduzir gravações musicais em cassete, a reprodução de programas é chamada de pirataria de software. Infelizmente, ela é difícil de ser evitada, detectada ou legalmente processada. Alguns fornecedores afirmam que para cada exemplar de programa comprado legalmente são feitas cerca de cem cópias ilegais.

Pirataria em software é, por definição, a atividade de reproduzir programas sem a devida autorização. Assim como acontece com as gravações musicais, ela se processa em diversos níveis e modos. Em um nível mais elementar, comete-se pirataria toda vez que o usuário de um microcomputador faz a cópia de um programa emprestado por um amigo. O fato de alguns programas (principalmente os escritos em linguagem de máquina) não poderem ser reproduzidos através de comandos normais de BASIC não constitui empecilho, uma vez que basta conectar dois gravadores cassete e copiar o programa de um para o outro, sem haver necessidade do uso de computador.

Fornecedores de jogos eletrônicos afirmam que para cada título de programa vendido são feitas aproximadamente cem cópias ilegais. Embora alguns possam suportar esse prejuízo, a verdade é que há muita gente que vive de royalties de programas e nem por isso ganha uma fortuna.

Existem controvérsias em torno de comerciantes que alugam ou emprestam software por um período de experiência, pois se acredita que estes esquemas facilitam as coisas para os que copiam programas. Comerciantes menos escrupulosos vão além, distribuindo cópias piratas de programas conhecidos aos compradores de microcomputadores, a fim de aumentar o valor real destes aparelhos.

Há ainda os casos em que distribuidores reproduzem programas em larga escala, vendendo-os a outros comerciantes. Estes produtos equivalem a reproduções clandestinas de gravações de rocks famosos, o que não chega a ser muito arriscado quando os fornecedores originais dos pacotes são de outro país.

A pirataria passa a ser mais sofisticada e mais difícil de se detectar quando, por exemplo, alguém pega um programa já existente, faz nele algumas modificações e comercializa-o como sendo seu próprio. A nova versão às vezes apresenta um desempenho melhor ou algumas funções adicionais, mas, por outro lado, pode apenas ter sofrido modificações nos caracteres de comandos do programa original e no layout da informação que aparece na tela, a fim de que o pacote não seja reconhecido de imediato. Esta prática é mais usada com programas de aplicação comercial do que com jogos eletrônicos.

Se este processo constitui ou não um caso de pirataria de software, da mesma forma que uma simples cópia, é um ponto bastante discutível, e por essa razão tanta gente escapa ilesa. A lei pouco defende os editores de software e as disposições sobre direitos autorais não protegem os programas contra pirataria ou plágios.

Aparentemente, os direitos autorais só dizem respeito a material impresso (com exceção da música); portanto, os programas de computação armazenados em RAM, disco, ou cassete não são resguardados. Como acontece com a maioria das questões de ordem legal, é necessário estabelecer precedentes, o que custa tempo e dinheiro.

Quando uma empresa copia a idéia de um programa conhecido e lança sua própria versão, surge uma questão bastante controvertida. Neste caso, eles não copiam os códigos do programa, mas tomam nota detalhadamente de como ele aparece no vídeo e reage aos dados fornecidos pelo usuário; então, a partir dessas anotações, escrevem um programa com o mesmo efeito. Um exemplo característico é o do PacMan, um jogo eletrônico que foi lançado em máquinas operadas por fichas, posteriormente distribuído pela Atari para ser usado em microcomputadores e em aparelhos de videogames, e que acabou sendo lançado, com pequenas alterações, por uma série de editores de software.

Todos estes jogos, com pequenas variações, apresentavam a conhecida criaturinha devorando pontos em seu caminho através de um labirinto. Após alguns meses, a Atari conseguiu eliminar a maior parte desses concorrentes por meio de ações judiciais e, em casos de operações menores, pela simples ameaça de processá-los.

Geralmente, autores e fornecedores de software recorrem a certos meios extrajudiciais a fim de proteger seus interesses. Alguns argumentam que vendendo seus programas a um preço bastante baixo, desestimularão aqueles que fazem cópias clandestinas. No caso de programas mais sofisticados, um manual bem produzido e uma embalagem atraente já proporcionam certo grau de proteção.

Um outro meio de proteger um software caro é o "registro do usuário": só recebe ajuda e apoio por telefone o usuário que devolver o cartão que acompanha o manual do proprietário.

Os assim chamados meios de proteção "hard" consistem quase sempre em um dispositivo do tamanho de uma caixa de fósforos chamado "dongle", que deve ser ligado a uma das interfaces do computador, a fim de permitir que o programa funcione. Os circuitos do dongle têm um código eletrônico, geralmente uma combinação dos dígitos 1 e 0, gravados em ROM. A intervalos freqüentes, o programa solicita o dongle, e, se não receber de volta o código correto, recusa-se a continuar. O código pode ser diferente para cada dongle, significando que cada cópia do pacote deverá ser combinada com o dongle que a acompanha.

Neste caso, só se consegue fazer cópias ilegais falsificando o dongle ou reescrevendo o código do programa para remover as seções que o solicitam. Sem dúvida, este procedimento está além da capacidade da maioria dos programadores amadores, o que se não evita a pirataria, ao menos a dificulta.

Intensas pesquisas vêm sendo desenvolvidas para obtenção de métodos que ofereçam essa mesma proteção, sem o auxílio de hardware adicional. Um recurso, semelhante à linha-d'água do papel, consiste em um código magnético sobreposto no cassete ou disco, "atrás" da gravação do programa propriamente dito, o qual não se transfere para a cópia, de modo que o programa só funciona no cassete ou disco originais.

A única proteção "hard" economicamente viável para os fornecedores de jogos eletrônicos é o cartucho ROM, que em geral alcança preços mais elevados, pois não necessita de longo tempo para carregamento, como no caso dos cassetes. Contudo, nem mesmo os cartuchos são inexpugnáveis: atualmente existem dispositivos que copiam um cartucho para um cassete ou para um outro cartucho, que pode ser programado ou reprogramado pelo usuário.

A pirataria em software é uma guerra de "bandidos e mocinhos", em que protagonistas tentam superar uns aos outros em engenhosidade. Nesta brincadeira, ninguém é eliminado — na melhor das hipóteses, pode-se dificultá-la o bastante para que se torne apenas uma atividade marginal.

"Dongles"
São pequenos dispositivos usados para proteger certos programas contra reproduções ilícitas. Os programas não funcionam enquanto o dongle certo não for ligado a uma interface do computador. A parte eletrônica é geralmente envolta em resina sólida, dificultando qualquer acesso aos circuitos.

Colocando em ordem

Quando se projeta um programa que envolve a armazenagem ou a manipulação de informação, é importante ficar atento à estrutura, a fim de que os dados não precisem ser duplicados.

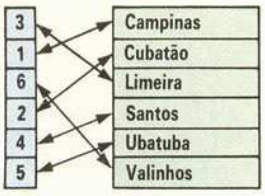
Trajeto reduzido
Uma firma tem interesses em seis cidades, e precisa enviar um caminhão para todas elas mensalmente. O computador da firma organiza o trajeto mais eficiente:

Cidade$(1)	"Campinas"
Cidade$(2)	"Cubatão"
Cidade$(3)	"Limeira"
Cidade$(4)	"Santos"
Cidade$(5)	"Ubatuba"
Cidade$(6)	"Valinhos"

Os nomes das cidades estão num arquivo seqüencial na fita, em ordem alfabética. São lidos do arquivo nessa ordem, através da matriz Cidade$( ). O computador estabeleceu que a melhor ordem é:

Limeira	Cidade$(3)
Campinas	Cidade$(1)
Valinhos	Cidade$(6)
Cubatão	Cidade$(2)
Santos	Cidade$(4)
Ubatuba	Cidade$(5)

Em vez de armazenar os nomes das cidades novamente nesta ordem, utilizando mais memória, o computador deposita apenas os números de posição na matriz Cidade$( ). Assim:



A lista de números é um índice para a matriz Cidade$( ). Outros índices para a matriz poderiam ser criados, cada um dando uma ordem diferente para os nomes. A vantagem de se utilizar índices deste modo é que o arquivo original não precisa ser separado ou duplicado; basta separar e duplicar o índice.

Qualquer pessoa que tenha utilizado um sistema de índice por cartões (o catálogo de uma biblioteca, por exemplo) sabe o quanto ele pode ser útil. Se alguma vez você deixou que esse arquivo caísse no chão e saísse da ordem, notou que os cartões perderam toda a utilidade. Um catálogo de biblioteca contém muitas informações, mas, a menos que estas estejam ordenadamente arrumadas, o valor desse sistema de informação é nulo.

A essência de um sistema de dados não é a informação em si, mas sim sua organização. Considere, por exemplo, este item fictício numa lista de endereços comerciais:

Silva, J., rua Alta 15, Centro

Tomada isoladamente, essa informação é limitada. No entanto, se você sabe que ela procede de certa seção da lista de endereços, a informação ganha novo significado, relativo à estrutura do local onde se insere.

A mais simples estrutura de dados é denominada arquivo: um grupo de dados com características comuns. O nome do arquivo revela algo sobre a informação nele contida, e o fato de todas as informações estarem reunidas sob o mesmo nome torna mais fácil sua compreensão. O arquivo pode ser visto como uma grande unidade de informação, ou como grupos específicos de pequenas unidades. Um livro é um arquivo; lê-se uma autobiografia geralmente como um conjunto, ao passo que, num livro de culinária, as receitas formam uma coleção de dados individuais.

Se o arquivo é grande, a busca de uma informação específica significa partir do primeiro item e observar todos os seguintes até que se encontre a informação desejada. A isso atribui-se o nome de busca seqüencial ou acesso em série. Um arquivo organizado desse modo denomina-se arquivo seqüencial de informação. Um programa de televisão ou a fala humana constituem arquivos seqüenciais de informação.

Os arquivos seqüenciais são comuns por poderem ser alterados facilmente, e por espelharem, de certo modo, os métodos do pensamento humano. Mas muitas vezes revelam-se vagarosos e não abrangentes. Por isso, é comum sua divisão interna em subarquivos, que podem ser encontrados facilmente sem que para isso se precise pesquisar todo o arquivo. Os livros eram simples arquivos seqüenciais até que a criação de capítulos, páginas numeradas e índices transformou-os totalmente. Os capítulos funcionam como um subarquivo do livro, e as páginas são subarquivos do capítulo.

Um arquivo que não exige pesquisa seqüencial é chamado arquivo de acesso direto. Um conjunto de músicas numa fita magnética pode ser considerado um arquivo seqüencial, já o mesmo conjunto em forma de disco transforma-se em um arquivo de acesso direto. Para se encontrar determinada música na fita é preciso tocá-la desde o começo e avançá-la gradualmente, enquanto, num LP, qualquer faixa pode ser encontrada se experimentarmos colocar o braço do toca-discos em todas elas.

O acesso direto depende de saber-se onde as coisas se localizam. Num livro, o índice faz esse trabalho para você. Saber onde as coisas estão significa trabalho — e custa dinheiro. A criação do índice de um livro é uma tarefa extra para o editor, e o tipo de informação contida no livro pode não justificar esta despesa: os romances, por exemplo, não são indexados, como em geral ocorre com livros de poesias.

Os computadores processam grandes quantidades de informação em alta velocidade, usando várias estruturas de dados. Estes devem ser armazenados permanentemente em fita magnética ou disco, de modo estruturado — em geral são utilizados arquivos seqüenciais —, e de forma bem diferente na memória central do computador.

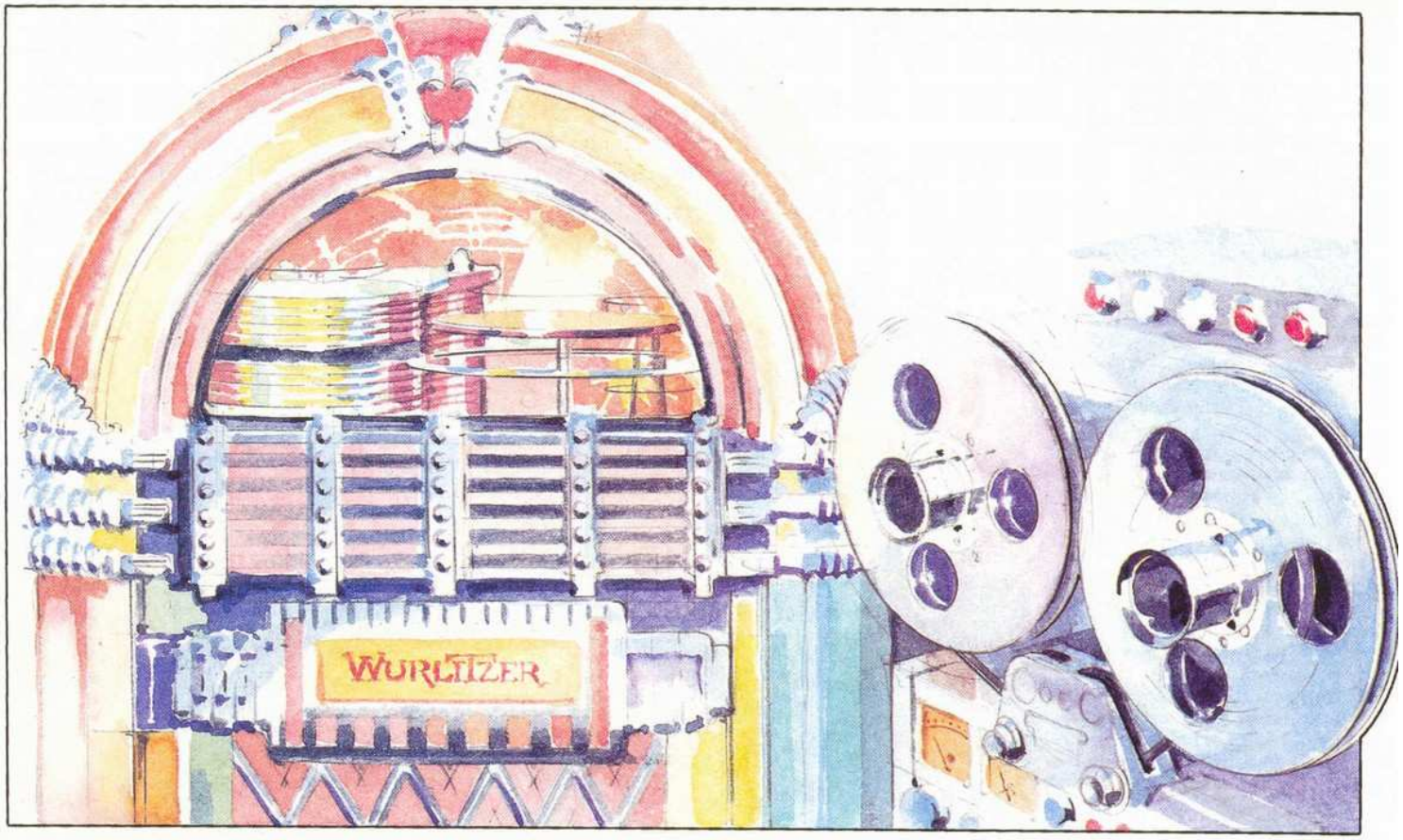
Suponhamos que uma padaria tem o endereço de todos os seus clientes num arquivo seqüencial em fita, e deseja que o computador imprima roteiros para os motoristas que entregam pão. O arquivo na fita será mais ou menos assim:

Almeida rua Alta 22
Arantes & Cia. av. Itabira 108
Duarte alameda da Saudade 49
Irmãos Wilson rua Alta 7
Soares estrada Paulistana 65
Vieira av. Atlântica 31

Quando o arquivo é lido da fita para a memória central, cada nome e endereço será depositado numa posição numerada e todas as posições reunidas formam um bloco de memória com seu próprio nome. Deste modo, o arquivo em memória fica assim:

NOME DE BLOCO: Estabelecimentos
1) Almeida rua Alta 22
2) Arantes & Cia. av. Itabira 108
3) Duarte alameda da Saudade 49
4) Irmãos Wilson rua Alta 7
5) Soares estrada Paulistana 65
6) Vieira av. Atlântica 31

Agora os itens de dados podem ser encontrados individualmente, bastando apenas que o bloco e a posição sejam nomeados. Estabelecimentos(4), por exemplo, contém Irmãos Wilson rua Alta 7. Essa estrutura, chamada matriz (ver p. 194), é mais comumente utilizada por computadores para processamento interno de dados. Assim como um livro traz



uma informação em cada página. Essa estrutura simples modifica o modo como vemos um bloco de dados anônimos. Não é necessário que os computadores conheçam o conteúdo de cada item; precisam apenas localizá-lo e saber o que fazer com ele.

Os dados na matriz Estabelecimentos( ) estão em ordem alfabética, mas não é certo que esta seja a ordem mais econômica para que se visitem os locais. Suponhamos que o computador descubra que o melhor programa de entrega é:

1) Irmãos Wilson rua Alta 7
2) Almeida rua Alta 22
3) Duarte alameda da Saudade 49
4) Arantes & Cia. av. Itabira 108
5) Vieira av. Atlântica 31
6) Soares estrada Paulistana 65

Essa ordem deve ser armazenada em outra matriz, mas isso significa que a mesma informação é armazenada na memória duas vezes. As pessoas que possuem micros sabem que a RAM é limitada, e talvez seja inconveniente ou impossível duplicarem-se dados desse modo. Portanto, um outro método deverá ser utilizado.

Caso os dados reais sejam substituídos por seus números de posição na matriz Estabelecimentos( ), o programa de entrega fica assim:

NOME DE BLOCO: Entregas
1) 4
2) 1
3) 3
4) 2
5) 6
6) 5

As instruções para o motorista, portanto, são estas: "Primeiramente vá ao estabelecimento cujos detalhes estão armazenados em Estabelecimentos(4), a seguir vá para Estabelecimentos(1), depois para Estabelecimentos(3), e assim por diante. A única informação significativa no programa é a ordem em que os locais devem ser visitados; assim, isso é armazenado na nova matriz, Entregas.

Entregas( ) constitui agora um índice para a matriz Estabelecimentos( ) com o objetivo de entregas. Quando imprime esses dados, o computador usa os números da matriz Entregas( ) para imprimir na ordem apropriada os nomes e os endereços da matriz Estabelecimentos( ).

Nesse exercício simples, a informação — os nomes e os endereços — foi manipulada mas não modificada pelas diferentes estruturas de dados impostas. Uma estrutura de dados não deve modificar o conteúdo dos dados, mas sim dar significado a eles.

Do mesmo modo que podemos reorganizar a matriz Estabelecimentos( ) criando um novo índice de acordo com a matriz Entregas( ), é possível produzir outros índices para diferentes fins. Quando estudamos bancos de dados (ver p. 124), percebemos que certa informação poderia ser selecionada utilizando-se ponteiros incluídos em cada registro individual. É o que ocorrerá se inserirmos em cada registro do arquivo Estabelecimentos( ) um ponteiro que mostre sua localização no programa de entregas. Poderíamos até mesmo aumentar o registro para incluir, por exemplo, um ponteiro que mostrasse os pedidos permanentes. O departamento de produção teria condições de percorrer o arquivo, extraindo somente a informação relacionada com o número de pães solicitado por cada cliente.

Na pista certa
Um toca-discos automático contém duzentas músicas em cem discos arquivados. Para se escolher qualquer música, pressionam-se três teclas. O tempo médio entre a escolha e o início da música é de 15 segundos. Já para que uma música seja encontrada em uma fita, contendo as mesmas duzentas músicas, o tempo médio será de 1.500 segundos ou 25 minutos. O toca-discos é um aparelho de acesso direto, rápido, bastante especializado e caro. O toca-fitas é um aparelho de acesso seqüencial: lento, muito menos especializado, mas razoavelmente barato. O toca-fitas cassete ligado a um microcomputador é um aparelho de acesso seqüencial, enquanto uma unidade de disco flexível é um aparelho de acesso direto, mesmo quando utilizado para armazenar arquivos seqüenciais.

Inimigo eletrônico

Alguns dos mais potentes microcomputadores já criados não se encontram em escritórios ou fábricas de alta tecnologia, mas em fliperamas, bares e restaurantes.

Invasores alienígenas
"Você precisa atacá-lo, antes que atinja o ponto zero, se não ele vai se multiplicar... Atenção! A nave-mãe está lançando uma esquadrilha... é melhor você usar a bomba Smart..."
Não, não se trata de um diálogo de filme de ficção científica, é apenas uma conversa ouvida por acaso em um fliperama.

Em 1971, um jovem chamado Nolan Bushnell gastou tempo e esforço tentando convencer proprietários de bares e restaurantes nas cercanias de Sunnyvale, Califórnia, onde vivia, a experimentar um novo tipo de jogo por ele inventado. Funcionava por meio de moedas, o que significava dinheiro em caixa para o arrendatário, mas baseava-se em uma idéia totalmente nova.

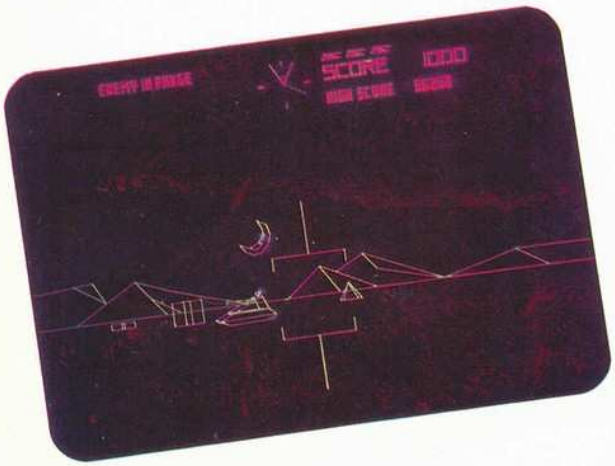
Finalmente, o dono de um bar concordou em experimentar. Dois dias depois, telefonava para o inventor, reclamando que o jogo havia quebrado. Bushnell logo descobriu que o defeito era bem simples — o compartimento para moedas estava repleto, e a fenda para colocá-las, obstruída. Resolveu o problema instalando uma caixa maior para as moedas.

Pong — uma variante do pingue-pongue — foi o precursor de todos os emocionantes e originais jogos eletrônicos, que funcionam baseados em microcomputação e se encontram agora por todo o mundo. Talvez seja interessante notar que, apesar de ter sido o primeiro a sugerir que jogos como pingue-pongue podem ser simulados por um computador, Bushnell ainda não tinha atingido o ponto essencial da questão: seu jogo requeria duas pessoas, para competir uma com a outra, e não permitia que um jogador sozinho experimentasse sua habilidade e destreza em confronto com a máquina. Longo tempo se passou até o surgimento da geração seguinte de jogos eletrônicos. Só em 1977 é que uma companhia japonesa, a Taito, entrou em cena com o jogo Space Invaders, que alcançou enorme sucesso.

Na terminologia moderna, Space Invaders é conhecido como um jogo no qual um "alienígena ataca usando escudos". O jogador move sua base de bombardeio ao longo da parte inferior da tela, ora protegendo-se atrás de escudos, ao sentir-se ameaçado, ora atirando contra uma fila de alienígenas estilizados, que avançam continuamente e respondem ao fogo inimigo a intervalos irregulares. O ritmo do avanço alienígena é previsível e seguido de um ruído eletrônico bitonal adequado, que acompanha sua

lenta aceleração. Descrito assim, esse jogo realmente parece bem simples, mas o efeito conjunto é fascinante e irresistível.

Os primeiros jogos Space Invaders utilizavam uma tela de televisão monocromática e gráficos de sondagem exploratória. Apresentavam poucas inovações em hardware e software, mas, quando surgiram, ocasionaram uma revolução social como há muito não se via. O hardware que sustentava Space Invaders e todos os seus similares diferia muito pouco do utilizado pelos micros da época. Os fabricantes desses microcomputadores logo perceberam que havia grande mercado inexplorado entre as classes de maior poder aquisitivo. Esse mercado teve seu foco rapidamente transferido do ensino de programação para o uso do computador como um meio de entretenimento, ao que se seguiu a elaboração de numerosos projetos de máquinas. Ainda hoje são vendidos muitos equipamentos devido ao fato de executarem jogos, apesar de os mais recentes jogos eletrônicos ultrapassarem a capacidade de quase todos os mais avançados microcomputadores. Não é raro encontrar até 1 milhão de bytes de memória em um jogo eletrônico atual. Seus recursos são semelhantes



aos encontrados em terminais gráficos utilizados em computadores de grande porte.

Muitas das conquistas no campo dos microcomputadores podem remontar aos jogos eletrônicos. A mudança para processadores de 16 e até 32 bits, por exemplo, deu-se como resultado da necessidade de endereçar mais de 64 Kbytes e de obter velocidades maiores de processamento. Os fabricantes de jogos eletrônicos estavam à frente desse movimento e entre os primeiros clientes de microprocessadores de 16 bits, como o Motorola 68000 e o Intel 8086. Sua necessidade de processamento rápido e memória de maior capacidade foi anterior à dos usuários de microcomputadores de escritórios.

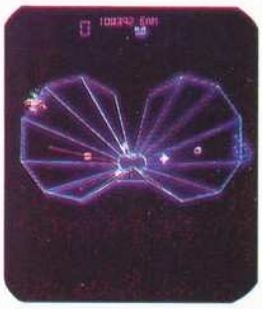
Derivaram daí vantagens para os usuários de microcomputadores, mesmo para os de menor poder aquisitivo. Os gráficos sprite, por exemplo, foram desenvolvidos para os jogos eletrônicos e, posteriormente, se tornaram acessíveis aos microcomputadores. Os eficientes chips para gráficos e produção de som, como o Video Interface Chip e o Sound Interface Device da Commodore, tiveram a mesma origem, assim como os três chips empregados pela Atari com a mesma finalidade em seus microcomputadores das séries 400 e 800.

A Atari, cujo êxito está fundado no jogo original de Nolan Bushnell, é um exemplo particularmente adequado do intercâmbio entre jogos eletrônicos e microcomputadores. Essa empresa há muito tinha descoberto no micro um meio de lazer antes de tudo original (um reflexo, talvez, do maior interesse de sua proprietária, a Warner Bros) e, além de sua série de micros, também oferece um eficiente computador para jogos, o VCS (Video Cartridge System), que leva para casa, praticamente sem alteração, muitos dos jogos encontrados em equipamentos de fliperamas. No Brasil, a Atari, através da Gradiente/Polyvox leva grande vantagem, é claro, por se tratar de importante produtora de equipamentos eletrônicos. Entretanto, outras empresas dedicadas ao lazer e entretenimento, como Philips e Sharp, estão também envolvidas na mesma atividade. Adquiriram os direitos de produção de muitos jogos eletrônicos para o mercado doméstico, que podem ser utilizados em suas consoles Odissey e Intellivision, respectivamente.

Todo o mercado, seja apenas para jogos eletrônicos, ou software de jogos para microcomputadores, está rapidamente se tornando uma indústria autônoma. Em termos de estratégia de marketing, lembra a indústria fonográfica, apresentando também as listas dos vinte jogos mais vendidos no momento. Os dois fenômenos são similares em outros aspectos: os softwares de jogos são objeto de troca entre crianças em idade escolar, e as atividades de pirataria de software estão proliferando dia a dia (ver p.192).

Mencionamos antes que o grau de desenvolvimento do hardware encontrado nos jogos eletrônicos coloca-os numa classe diferente da dos micro-

computadores. O mais recente destes desenvolvimentos requer o uso de videodiscos para proporcionar um cenário de fundo no monitor de televisão, contra o qual os jogos são realizados. Ainda não se sabe, porém, se esses refinamentos chegarão a alcançar o mercado doméstico. A tecnologia certamente está a preços acessíveis ao consumidor de produtos eletrônicos, embora o software de jogos produzido para consoles e microcomputadores não tenha realmente atingido o padrão dos jogos eletrônicos em sua forma original.



Campos de batalha

Jogos eletrônicos, como Space Invaders, em que o jogador pode mover seu sinal apenas em uma linha fixa, têm sido igualados em popularidade aos jogos de labirinto e perseguição, como PacMan. Ambos utilizam gráficos sprite e, por isso, têm qualidade visual inferior. Ultimamente, os projetistas de jogos têm produzido representações bem mais abstratas, como Battlezone, jogo de tanque e missil contra um jogador do futuro, ou Tempest, que utiliza bela composição gráfica produzindo uma ilusão de expressiva profundidade.

Cápsula espacial

Astron Belt é um jogo característico da nova geração de jogos eletrônicos, que utiliza discos laser (ver abaixo) para proporcionar um fundo que muda, no jogo em andamento. Por serem discos de acesso aleatório, é possível a passagem direta de uma cena a outra — de um confronto com uma nave alienígena, para uma explosão, por exemplo. Para o jogador, é um grande passo em direção ao realismo. As imagens no disco laser podem ser filmes da vida real ou desenhos animados feitos pelo computador. O Astron Belt também oferece som estéreo e um assento trepidante, que reage a baixas freqüências no circuito sonoro

Quando tratamos dos efeitos de simulação de vôo (ver p. 201), observamos que todos os jogos eletrônicos são simulações de um tipo ou de outro, quer de uma situação da vida real, como o jogo de pingue-pongue ou uma corrida de carros, quer de uma fantasia, como Space Invaders, PacMan ou Frogger. Durante os anos que se seguiram à invenção dos jogos eletrônicos, estes dois grandes ramos têm se expandido em correntes paralelas de desenvolvimento, apesar de os jogos de fantasia/espaço serem muito mais comuns.

A partir desses dois jogos originais — Pong e Space Invaders — muitas alterações foram sendo introduzidas.

O primeiro jogo de bola e bastão a colocar o homem contra a máquina foi o Breakout e suas variantes, no qual o jogador atira uma bola contra um muro de tijolos. Cada tijolo desaparece da tela assim que é atingido, e o objetivo do jogo é removê-los todos, sem perder a bola. Como resultante deste jogo, temos simulações de golfe, snooker/bilhar e pebolim. De modo geral, quanto maior a atenção do programador em reproduzir as forças físicas (por exemplo, a gravidade, a resistência do ar, o atrito, o choque), melhor será o jogo.

Mas tais critérios não se aplicam aos jogos de fantasia. Aqui, o jogador está na verdade competindo com a pessoa que programou a máquina para o jogo em questão, inteiramente sob suas regras e no universo por ela criado. O primeiro estágio após Space Invaders foi introduzir diferentes espécies de alienígenas, atacando de formas diversas e a intervalos irregulares. Em seguida, foi necessário acrescentar movimento aos sinais executados pelo jogador na tela, o que deu origem a jogos como Defender, considerado pelos conhecedores como o melhor de todos os jogos deste tipo.

Jogos de linhas fixas, como o Space Invaders inicial, continuaram a se desenvolver, originando outros como Missile Command e seus derivados, nos quais o objetivo do jogador é defender sua base dos ataques de mísseis balísticos intercontinentais. Jogos de labirinto/perseguição, como o Pac-Man, evoluíram dos primeiros jogos de corrida, em que o objetivo era dirigir um elemento em forma de carro em um trajeto, sempre que possível no tempo permitido, sem colidir com as muretas. Não havia nenhum elemento de antagonismo nesses jogos — mesmo considerando as manchas de óleo ocasionais que apareciam como por mágica — e, assim, o passo seguinte foi transformar o que era antes "uma competição contra o tempo" em uma perseguição. A pista inicial transformou-se então em um labirinto, e os sinais, por sua vez, em frutas, lâmpadas ou algo parecido.

Jogos de corridas de automóveis tornaram-se representações pseudotridimensionais do trajeto, visto do carro, ou atrás deste, com a estrada sempre se modificando e surgindo à frente do jogador. Um método muito semelhante é utilizado nos jogos de simulação de vôo mais realísticos.

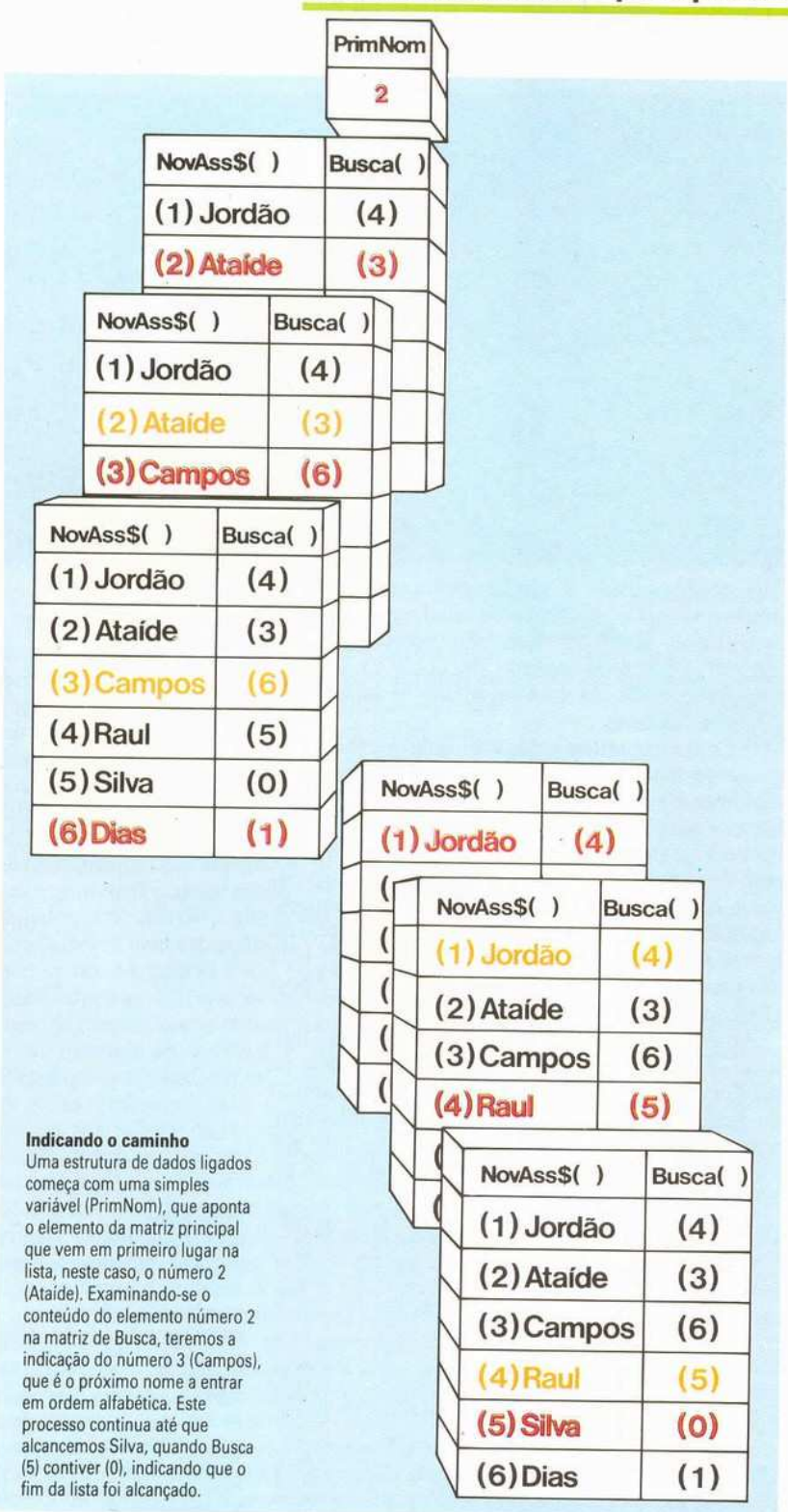
Por fim, vêm os tradicionais jogos de tabuleiro, como dama, xadrez e gamão. Seu uso restringe-se aos microcomputadores porque, em geral, é preciso um tempo muito longo para jogá-los e a representação gráfica está em segundo plano no programa, em relação aos próprios algoritmos dos jogos.

A única diferença técnica real está no método de geração de gráficos. Os primeiros jogos e a maior parte dos atuais apresentam gráficos de sondagem exploratória, mas alguns, em especial Asteroids, usam métodos de sondagem vetorial, por meio dos quais somente as imagens na tela, não as áreas escuras, são sondadas pelo feixe de elétrons.

Assim, a próxima vez em que você passar por um fliperama, lotado de jogos complicados, ou apoiar-se em um deles num bar, lembre-se de que o computador que está no interior da máquina é, provavelmente, muito mais complexo e potente que aqueles usados em casa ou nos pequenos escritórios, utiliza muitas de suas técnicas e é programado por alguns dos mais talentosos profissionais do mundo de hoje.

Nomes encadeados

A indexação é uma forma de estruturar muitos dados, tais como nomes e endereços. A lista de ligação ou encadeamento é outra alternativa que apresenta vantagens distintas.



Indicando o caminho
Uma estrutura de dados ligados começa com uma simples variável (PrimNom), que aponta o elemento da matriz principal que vem em primeiro lugar na lista, neste caso, o número 2 (Ataíde). Examinando-se o conteúdo do elemento número 2 na matriz de Busca, teremos a indicação do número 3 (Campos), que é o próximo nome a entrar em ordem alfabética. Este processo continua até que alcancemos Silva, quando Busca (5) contiver (0), indicando que o fim da lista foi alcançado.

Na memória de um computador só existem dados, byte após byte, armazenados em milhares de padrões de voltagem. O significado é dado aos bytes através da estrutura de dados que o processador central impõe. Estas várias estruturas de dados decidem se algum byte específico é interpretado como parte de uma instrução, ou como dígitos pertencentes a um número maior, ou como um código de caractere.

Do ponto de vista do usuário, alguns tipos de estrutura de dados são ligados nos computadores por fios. Linguagens de programação geralmente exigem que os dados sejam estruturados segundo um limitado número de maneiras. O BASIC classifica dados como numéricos e alfanuméricos, e fornece variáveis e matrizes que possibilitam sua manipulação. Outras linguagens em geral utilizam esta e outras estruturas de dados, cuja extensão e variedade determinam o poder da linguagem.

As estruturas de dados em BASIC — variáveis e matrizes — constituem tudo que precisamos para simular outras maneiras de se examinarem dados.

A matriz indexada é uma estrutura de dados útil, facilmente utilizada em BASIC. Tem suas limitações, entretanto, particularmente quando os dados podem ser alterados de modo imprevisível.

Suponhamos que uma companhia telefônica tenha um arquivo dos novos assinantes, que serão incluídos na próxima lista telefônica. Até este momento, os nomes e endereços estão arquivados em ordem alfabética, para serem encontrados com maior facilidade, mas o arquivo cresce constantemente, e novos dados chegam a todo instante. Certo dia, o arquivo NovAss$( ) pode estar desta forma, quando colocado na matriz:

NovAss$( )	Índice( )
(1) Jordão	(2)
(2) Ataíde	(3)
(3) Campos	(6)
(4) Raul	(1)
(5) Silva	(4)
(6) Dias	(5)

A matriz Índice( ) mostra a ordem em que NovAss$( ) deve ser lido para que os novos itens sejam colocados em ordem alfabética. Desta forma, o primeiro item a ser colocado alfabeticamente é NovAss$(2), Ataíde. O segundo item é NovAss$(3), Campos. Neste exemplo, apenas os nomes estão sendo mostrados, mas, na verdade, uma lista abrange nomes, iniciais e endereços — cerca de sessenta caracteres normalmente. A movimentação de sessenta caracteres na memória é bastante lenta (já

que a seleção requer a movimentação de muitos dados) e desperdiça memória; sendo assim, é mais eficaz manter NovAss$( ) em seu lugar. Agora, um novo nome, Barros, deve ser acrescido à lista:

NovAss$( )	Índice( )
(1) Jordão	(2)
(2) Ataíde	(7)
(3) Campos	(3)
(4) Raul	(6)
(5) Silva	(1)
(6) Dias	(4)
(7) Barros	(5)

Note que o conteúdo do Índice( ) acima do novo acréscimo continua o mesmo, e o conteúdo abaixo dele permanece na mesma ordem anterior, mas todos pularam uma casa abaixo na matriz. Assim, o acréscimo no índice requer: encontrar a posição do novo elemento, mover cada elemento entre ele e o do fim do índice e escrever a nova entrada. É preferível fazer isso, em vez de fazer a mesma coisa com os dados reais, NovAss$( ), mas esse processo ainda é relativamente lento, se o índice for muito extenso.

Suponha, agora, que estruturamos os dados de outra forma. Deixemos NovAss$( ) não classificada, já que sua manipulação é lenta e cara. Vamos, então, estabelecer uma matriz paralela chamada Busca( ), cujos conteúdos são numerados de acordo com as posições equivalentes na NovAss$( ):

PrimNom(2)

NovAss$( )	Busca( )	Índice( )
(1) Jordão	(4)	(2)
(2) Ataíde	(3)	(3)
(3) Campos	(6)	(6)
(4) Raul	(5)	(1)
(5) Silva	(0)	(4)
(6) Dias	(1)	(5)

A primeira diferença é que uma simples variável chamada PrimNom é necessária: ela indica NovAss$(2) que, alfabeticamente, é o primeiro elemento de NovAss$( ). A outra diferença é que o número (0) foi utilizado em Busca(5), o que indica que o NovAss$(5) é, alfabeticamente, o último da matriz.

A outra diferença diz respeito ao conteúdo do Índice( ) e de Busca( ). O Índice( ) deve ser lido: "o primeiro elemento está em NovAss$(2), o segundo está em NovAss$(3), o terceiro em NovAss$(6)" etc. Ao passo que a leitura do PrimNom é feita da seguinte maneira: "o primeiro elemento está em NovAss$(2); e Busca(2) diz que o próximo elemento está em NovAss$(3); Busca(3) diz que o elemento seguinte está em NovAss$(6), e assim por diante. Busca(5) diz que NovAss$(5) é o último elemento. O Índice( ) dá uma posição absoluta para os elementos do arquivo, enquanto Busca( ) fornece apenas posições relativas; qualquer item em Busca( ) pode lhe informar apenas onde encontrar o elemento seguinte e não dá nenhuma informação sobre a posição absoluta. O número de Índice(4) indica o quarto item que está alfabeticamente organizado no arquivo, enquanto o número em Busca(4) indica apenas o item que segue o NovAss$(4) no arquivo classificado. O Busca( ) auxilia a estrutura de dados chamada Lista de Ligação. A leitura de uma Lista de Ligação é como a procura de um tesouro: no início você recebe informações sobre o caminho a seguir; quando chega a certo local, encontra uma indicação do próximo etc. A leitura de uma matriz organizada em Índice é como participar de uma corrida de automóveis. No início, você recebe informações a respeito da rota e a ordem do percurso.

A grande vantagem da estrutura de Lista é a flexibilidade que apresenta. Considere a Lista após o acréscimo do elemento Barros:

PrimNom (2)

NovAss$( )	Busca( )
(1) Jordão	(4)
(2) Ataíde	(7)
(3) Campos	(6)
(4) Raul	(5)
(5) Silva	(0)
(6) Dias	(1)
(7) Barros	(3)

A matriz Busca( ) foi modificada em dois lugares:

i) Busca(2), que anteriormente indicava o NovAss$(3) que continha o próximo elemento em ordem alfabética depois do NovAss$(2), agora indica o NovAss$(7), já que ele passa a representar o elemento seguinte ao NovAss$(2).

ii) Busca(7), que antes não era usado, agora indica o NovAss$(3) como o próximo item depois do NovAss$(7), que aparece em ordem alfabética.

Isso ilustra o processo geral de inserção ou acréscimo em uma Lista de Ligação: encontrar o elemento da lista que deveria vir antes do novo elemento, e fazer com que ele aponte o novo elemento; a seguir, fazer com que o novo elemento indique aquele que perdeu a colocação. Estas operações constituem tudo que é necessário para um acréscimo à Lista de Ligação, e apenas as operações básicas serão afetadas pelo tamanho da Lista. A inserção de um elemento na Lista é como pôr um elo numa corrente — decida onde colocar o elo, rompa a corrente, junte o novo elo ao anterior e ao seu consecutivo. As Listas de Ligação são às vezes chamadas de Listas de Encadeamento. Os números em Busca( ) — os elos — são chamados ponteiros.

Um fator importante das Listas é sua característica de seqüência: é impossível encontrar um elemento na Lista a não ser começando pelo início da Lista e pesquisando cada elemento até encontrar aquele desejado. A Lista é aqui complementada pelo uso de matrizes, que são projetadas com o objetivo de serem estruturas de Acesso Direto, mas a Lista transformou-as efetivamente em Arquivos Seqüenciais. Em outras linguagens, como LISP ou PASCAL, o recurso da Lista já vem incorporado.

As Listas são estruturas úteis quando se quer lidar com dados dinâmicos (dados que mudam com freqüência) e quando se trata de uma linguagem natural (como no reconhecimento da fala) ou de uma artificial (compilação de programas), onde os próprios dados formam uma lista de elementos.

Um livro de figuras

O Lisa, da Apple, reproduz em sua tela os objetos usualmente dispostos sobre uma mesa de escritório. Mas muitas de suas funções estão passando para os micros domésticos.

O Lisa, produzido pela Apple Computer, é uma máquina de uso exclusivamente comercial. Seu preço, mesmo sem impressora, é elevado. Você, então, obviamente, estará indagando o que essa máquina tem a ver com o MICROCOMPUTADOR — CURSO BÁSICO. A resposta é simples: decidimos dar tamanha atenção ao Lisa porque se trata de um produto pioneiro e várias de suas funções começam a ser incorporadas aos micros não-comerciais. A própria Apple já lançou um equipamento menor e mais barato com as mesmas características — o Macintosh. E os concorrentes se preparam para competir com a capacidade de desempenho desse produto.

A novidade em torno do Lisa não é o seu hardware e sim o software padronizado que o acompanha. O desenvolvimento de um software sofisticado vem sendo uma tendência generalizada de todos os fabricantes de micros. Hoje são despendidos menos homens-hora para projetar e construir uma nova máquina do que para escrever um complexo jogo eletrônico ou um programa de uso comercial. O software tornou-se o elemento mais importante de qualquer sistema de computador e também o mais caro. Usuários de microcomputadores constatam que é possível gastar, por ano, com jogos, programas aplicativos e utilitários o mesmo dinheiro que pagaram pela máquina.

Mesmo assim, analisaremos primeiro o hardware do Lisa, cujo design foi fundamentalmente ditado pelas exigências do software. A memória-padrão do Lisa é de 1 megabyte de RAM (isto é, mil vezes a memória-padrão de um Sinclair ZX81). Uma memória tão grande exige que o microprocessador gaste muito tempo com "administração de memória" — movimentando os dados e controlando os lugares em que eles estão armazenados. O processador é um Motorola 68000, capaz de processar 16 bits de informação de cada vez, ao passo que as CPUs da maioria dos microcomputadores usuais processam 8. Pelos padrões dos microcomputadores, é um processador muito rápido, com um conjunto de instruções realmente avançado. Para armazenamento permanente, o sistema Lisa inclui duas unidades de disco flexível e uma unidade de disco rígido independente e com poucas funções externas. O disco rígido é necessário tanto pela sua capacidade (5 megabytes) quanto pela velocidade — o Lisa faz uso de um grande número de programas que freqüentemente precisam ser trocados entre a RAM e o disco. Discos rígidos serão estudados com maior profundidade neste curso, pois já estão aparecendo unidades de baixo custo no mercado de microcomputadores.

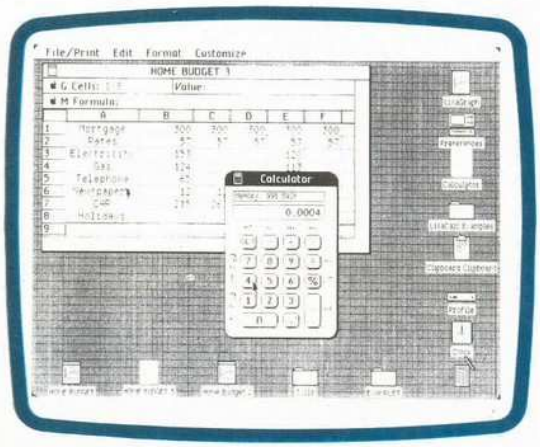
Outra característica marcante do hardware do Lisa é o display monocromático embutido no mesmo módulo da CPU, que tem uma resolução de 720 x 364 pixels. Ele proporciona uma variedade de tipos diferentes para elaboração de textos, além dos gráficos descritos mais adiante neste artigo. O Lisa dispõe de chips e circuitos especiais com a função específica de dirigir esse display e mover rapidamente as imagens.

Fácil para o usuário
O Lisa foi projetado para ser usado pelos que trabalham em escritório e não têm experiência com computadores. O manejo do "mouse" permite que o teclado seja usado com menos freqüência do que em outros sistemas.

Ligado a uma impressora apropriada — do tipo matriz de pontos, de alta qualidade e velocidade — é possível reproduzir no papel qualquer coisa exibida na tela. Se a impressora não for compatível com o alto nível de resolução da tela, o Lisa produzirá apenas uma imagem impressa, de qualidade razoável.

O teclado do Lisa é separado da unidade principal e tem um bom layout. Contudo, é menos usado que o de outras máquinas, porque o Lisa possui um "mouse". Um mouse é uma das muitas alternativas para introduzir informações na tela sem usar o teclado — entre outros métodos, incluem-se joysticks, canetas ópticas e unidades de reconhecimento de voz. Do tamanho de um maço de cigarros, o mouse, ligado ao computador por um cabo de conexão, é uma caixa que movimentamos manualmente sobre a

A vantagem do "mouse"
Toda função é representada por um símbolo chamado "ícone" (figura). Para executá-la, o mouse é movido até o cursor ficar sobre a figura escolhida e aí o botão SELECT do mouse é apertado. Isto "abre" a aplicação, que é exibida com detalhes na tela do Lisa.



superfície de uma mesa. Movendo o mouse, impulsionamos um "cursor", ou ponteiro, que "anda" pela tela. Deste modo, apontando o cursor para a informação ou comando que necessitarmos, apertamos o botão SELECT existente no mouse e a informação será selecionada de imediato ou o comando executado. O teclado só é usado quando novos dados em forma de texto ou de números tiverem de ser introduzidos no computador.

Neste ponto, podemos analisar o software do Lisa e novamente ressaltar que, embora as aplicações usuais dessa máquina se concentrem totalmente na área comercial, os princípios sob os quais elas operam acabarão, sem dúvida, sendo transmitidos para as aplicações dos microcomputadores pessoais (menores) e domésticos.

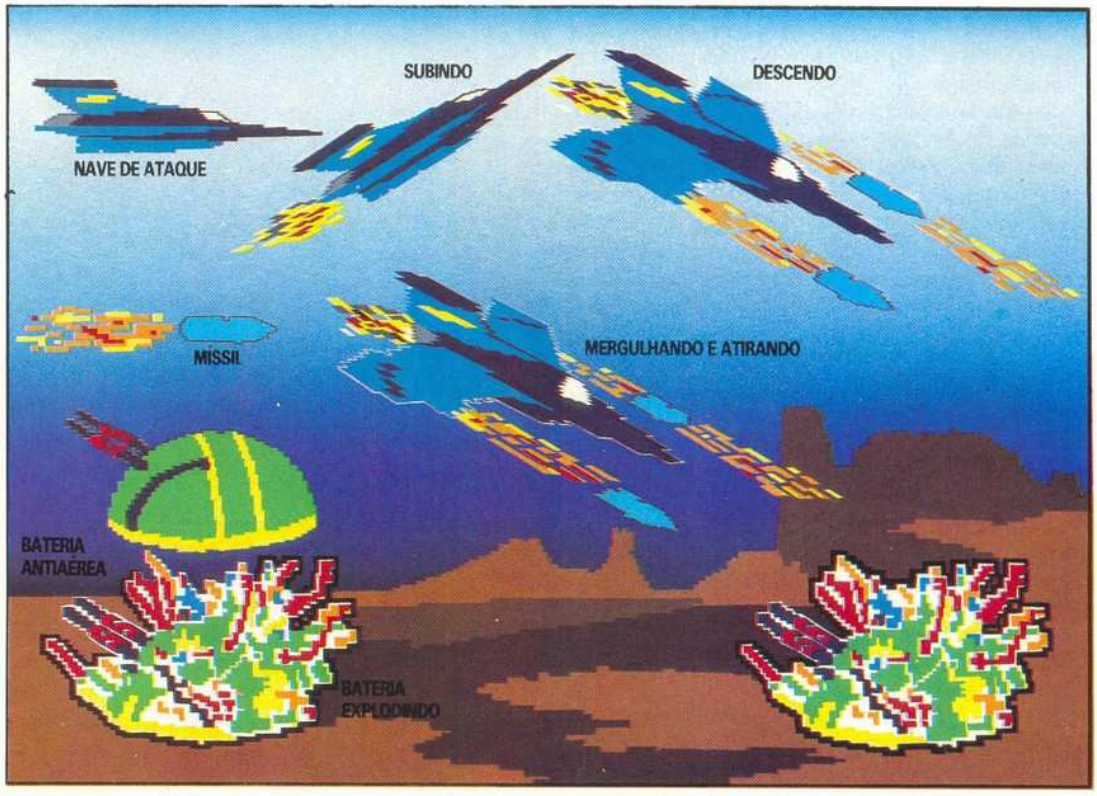
Quando você liga a máquina, a imagem que aparece no Lisa representa o tampo de uma mesa de trabalho com diversos objetos dispostos sobre ela. De fato, quase tudo que você faz no Lisa aparece representado na tela como se você estivesse trabalhando sem o auxílio do computador. Esta é a principal razão de os principiantes acharem tão mais fácil enfrentar o Lisa do que o hardware e o software convencionais. Cada um dos objetos arrumados sobre a mesa é chamado de "ícone" (figura) e representa uma função específica, geralmente assinalada abaixo.

Tomemos como exemplo a figura do relógio. Movendo o cursor sobre o pequeno relógio por intermédio do mouse e apertando o botão SELECT, aparecerá um relógio maior na tela, junto com a data. Se você não quiser que o relógio grande fique atravancando a mesa, ele pode ser simplesmente "fechado" para o tamanho original. Da mesma forma, selecionando a figura de uma calculadora, você "abrirá" uma calculadora maior, que pode ser usada para fazer cálculos aritméticos. Se você não estiver satisfeito com a disposição das figuras na mesa, pode mudá-las de lugar movimentando o mouse com o botão SELECT pressionado. Um dos efeitos mais divertidos, e que ilustra a que ponto o Lisa é modelado de acordo com o nosso modo de trabalhar, é a figura do cesto de lixo. Quando você não precisa mais de alguma parte do trabalho, joga-a no lixo usando o mouse. Com este procedimento, torna-se muito difícil apagar alguma informação acidentalmente. Você pode até examinar o conteúdo do cesto de lixo e recuperar o que for necessário, contanto que o Lisa não tenha sido desligado nesse meio tempo.

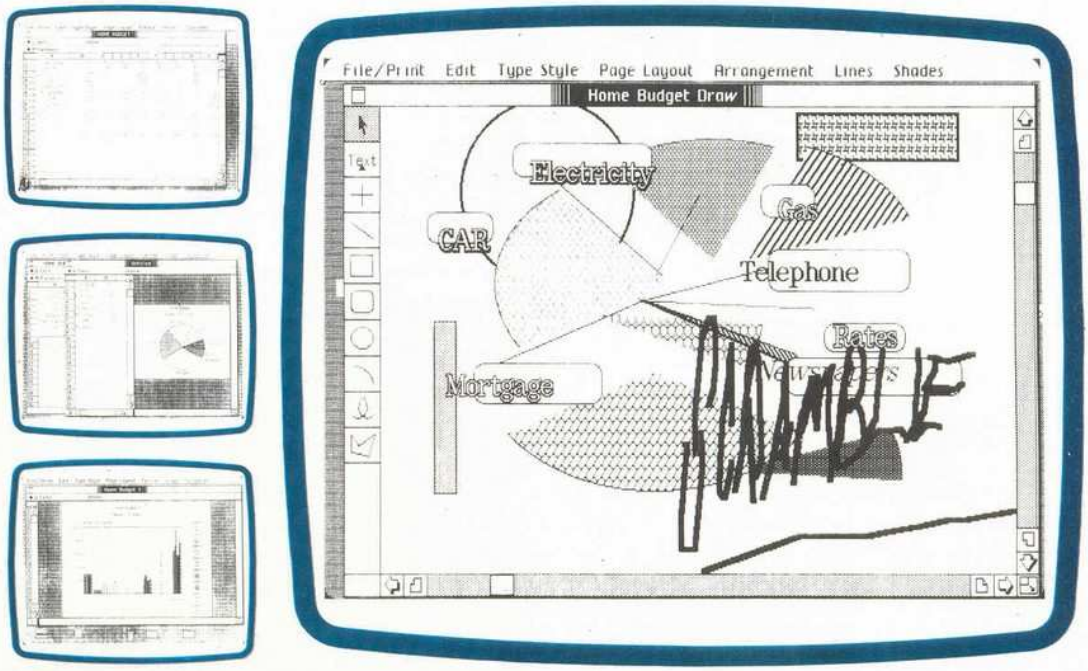
A maior parte do trabalho no Lisa é feita com a utilização de seis sistemas de aplicações: LisaWrite, um processador de palavras; LisaCalc, uma folha eletrônica; LisaGraph, um sistema para desenhar gráficos; LisaList, um gerenciador de banco de da-

Programação rápida
Para criar um jogo como "Defensor" em programação convencional, você desenharia numerosos modelos para o layout da tela, e então escreveria um programa a partir do zero, para controlar o jogo. Usando a programação de objeto orientado do Lisa, você pode concentrar-se em cada elemento individualmente.

Começando pela nave de ataque, você estabelece que ela andará sempre da esquerda para a direita, que quando o joystick for movimentado para a frente ou para trás, a nave andará respectivamente para cima e para baixo; quando o botão FIRE for acionado, um missil será lançado. A seguir, você define a forma do míssil e determina que ele continue na mesma direção até entrar em contato com outro objeto, o que fará com que ele desapareça. A bateria antiaérea é definida por uma forma simples, que se transforma numa explosão, se atingida pelo missil. Passe as três definições para linguagem adequada, ponha-as na forma de instruções, e jogue.



Passando informação adiante
Uma das características do Lisa é a capacidade de fazer a informação passar de uma aplicação para outra, usando a função COPY. Ela armazena, temporariamente, informação pronta para ser afixada em outra "janela". Por exemplo, comecemos por analisar alguns dados usando LisaCalc (a folha eletrônica).
Os números resultantes são copiados no LisaGraph, que com eles produz automaticamente um gráfico em forma de barras ou círculo.
Finalmente, a imagem é transferida para o LisaDraw, onde é acabada com títulos, setas, diagramas etc.
O resultado final pode então ser impresso.



dos/listas; LisaProject, um auxiliar para planejamento de projetos; e LisaDraw, um instrumento sofisticado para criar qualquer tipo de imagem gráfica. As figuras ou ícones para estas aplicações são simples blocos de papel. Para fazer um cálculo na folha eletrônica, por exemplo, o cursor é posicionado sobre o bloco de papel riscado com linhas e colunas chamado LisaCalc. Apertando o botão SELECT, uma folha deste bloco é "arrancada" e colocada em outro lugar da mesa, podendo receber um título como "planejamento de vendas", por exemplo.

No caso de o usuário precisar ter várias aplicações de folhas eletrônicas na mesa ao mesmo tempo, ele deverá resselecionar a mesma figura. Num sistema de computador normal, você teria de passar pelo processo de carregar o programa de folha eletrônica e então especificar o arquivo de dados com o qual quer trabalhar. No Lisa, entretanto, programa e dados são inseparáveis. Este é um outro exemplo de programação de objeto orientado, que vimos no artigo que trata do Programa de Criação de Fliperama (Pinball Construction Set) (ver p. 241).

Outra importante característica do "ambiente operacional" (como é chamado) do Lisa é sua capacidade de apresentar "janelas". Quando uma aplicação é selecionada, ela aparece como uma grande folha de papel sobre a mesa. O tamanho dessa folha pode ser especificado com a utilização do mouse. Caso haja mais de uma aplicação "aberta" ao mesmo tempo, as folhas ficam sobrepostas, e a que estiver sendo usada no momento ficará no alto da pilha, exatamente como se elas estivessem sobre a mesa. Pode acontecer que a aplicação na qual você esteja trabalhando, o processador de palavras, por exemplo, necessite de maior espaço do que o da folha que você especificou. Neste caso, a folha passa a atuar como uma janela sobre a aplicação e pode ser movimentada de um lado para outro, mostrando qualquer parte do documento completo. O princípio das janelas foi amplamente explicado quando examinamos folhas eletrônicas (ver p. 158).

É possível passar informação de uma aplicação para outra utilizando-se os ícones ou figuras, e esta é outra importante função do Lisa. Digamos que você esteja fazendo uma análise do seu volume de vendas mensal, usando LisaCalc. Por meio da função COPY, que é selecionada de um menu de funções especiais listadas no alto da tela, você faz uma cópia temporária dos resultados da folha eletrônica. Então, escolhendo um pedaço de papel LisaGraph, esses resultados são introduzidos na seção INPUT DATA da aplicação para traçar gráficos, bastando para isso recorrer à opção PASTE do menu. Caso seja solicitado, o LisaGraph produzirá um gráfico em forma de círculo (ou de barras, ou de linhas), totalmente marcado e graduado. Agora, usando de novo as opções COPY e PASTE, que se encontram no alto da tela, pode-se copiar esta imagem num pedaço de papel LisaDraw.

Esta última aplicação lhe permite completar o gráfico com setas e diagramas, ou substituir os tipos dos títulos e legendas por uma variedade de modelos diferentes. O resultado final pode ser impresso e copiado para um retroprojetor de slides ou usado como arte final para um relatório ou artigo de revista.

Como dissemos, os princípios da programação de objetos orientados e dos ambientes operacionais no estilo do Lisa serão brevemente transferidos até para as máquinas mais baratas, principalmente se elas se tornarem mais avançadas quanto à velocidade do processador e ao tamanho da memória RAM. Procure imaginar: se houvesse janelas múltiplas na tela do seu microcomputador, você poderia escrever um programa em uma delas e observar o output em outra. Em seguida, você poderia "chamar" os auxiliares de programação, bastando apontar para uma figura em forma de caixa de ferramentas, e mudar as sub-rotinas de lugar na sua listagem simplesmente movendo o mouse sobre sua mesa de trabalho. Esperemos que tal capacidade de desempenho do software não esteja muito distante.

Comportamento simulado

A simulação é uma técnica em computador que permite experimentar situações que em outras condições seriam muito perigosas ou dispendiosas. Simulações em microcomputadores podem ser muito instrutivas.

Uma das mais importantes utilizações do computador situa-se na área da simulação. Trata-se de um método de planejamento em que se examina um modelo da situação a ser analisada, simulando-a no computador. O modelo fornece uma visão simplificada da situação específica, retendo os aspectos importantes do problema e descartando os detalhes de menor influência nos resultados finais.

Para analisar uma situação, tomemos um exemplo de dois copos, um com vinho branco e o outro com vinho tinto. Coloca-se uma colher de vinho tinto no vinho branco e, depois de misturados, uma colher desse líquido é posta no copo de vinho tinto. Qual o copo que fica com maior "impureza"?

Há várias maneiras de resolver esse problema, mas uma das mais simples é utilizar um modelo. Por exemplo, podemos estabelecer um modelo em que o volume de vinho em cada copo seja exatamente de uma colher. Será fácil perceber que ambos os copos ficarão com o mesmo grau de impureza (uma mistura igual de tinto e branco). Se ampliarmos esse modelo para maior quantidade de vinho, chegaremos à mesma conclusão.

Há três formas principais de modelos. Um modelo que podemos chamar de pictórico ou bem expressivo — por exemplo, uma fotografia ou até mesmo um mapa — mostra os arranjos e relações espaciais entre os elementos da figura. Outra forma de modelo é aquele cujos componentes se comportam, um em relação ao outro, de modo semelhante aos elementos reais que ele representa no problema — por exemplo, os problemas solucionados por computadores analógicos (ver p. 238). O terceiro tipo é o modelo que usa símbolos abstratos e relações matemáticas para representar uma situação. Os computadores digitais usam este último tipo na simulação.

Destacam-se quatro situações de problemas que podem ser solucionadas por simulação no computador. A primeira é a que apresenta alto risco e impede experiências reais; por exemplo, a determinação do nível seguro de radiatividade numa usina nuclear.

Na segunda situação, da qual um bom exemplo é a economia de um país, seria quase impossível encontrar solução matemática simples para o conjunto de equações que compõem o problema. Será melhor dispor as equações na forma de um modelo no computador e observar os efeitos de diferentes ações e eventos sobre ele.

O terceiro caso se dá quando o problema a ser analisado envolve tanta despesa que alguns ajustes devem ser feitos no modelo, para evitar um erro na versão final. No projeto de um novo aeroporto em Londres, por exemplo, um longo trabalho de simulação foi realizado para solucionar questões de planejamento e engenharia. Essa simulação fez pesquisas sobre o barulho e outras condições relacionadas ao meio ambiente, assim como o fluxo de pessoas e o tráfego em torno da área escolhida.

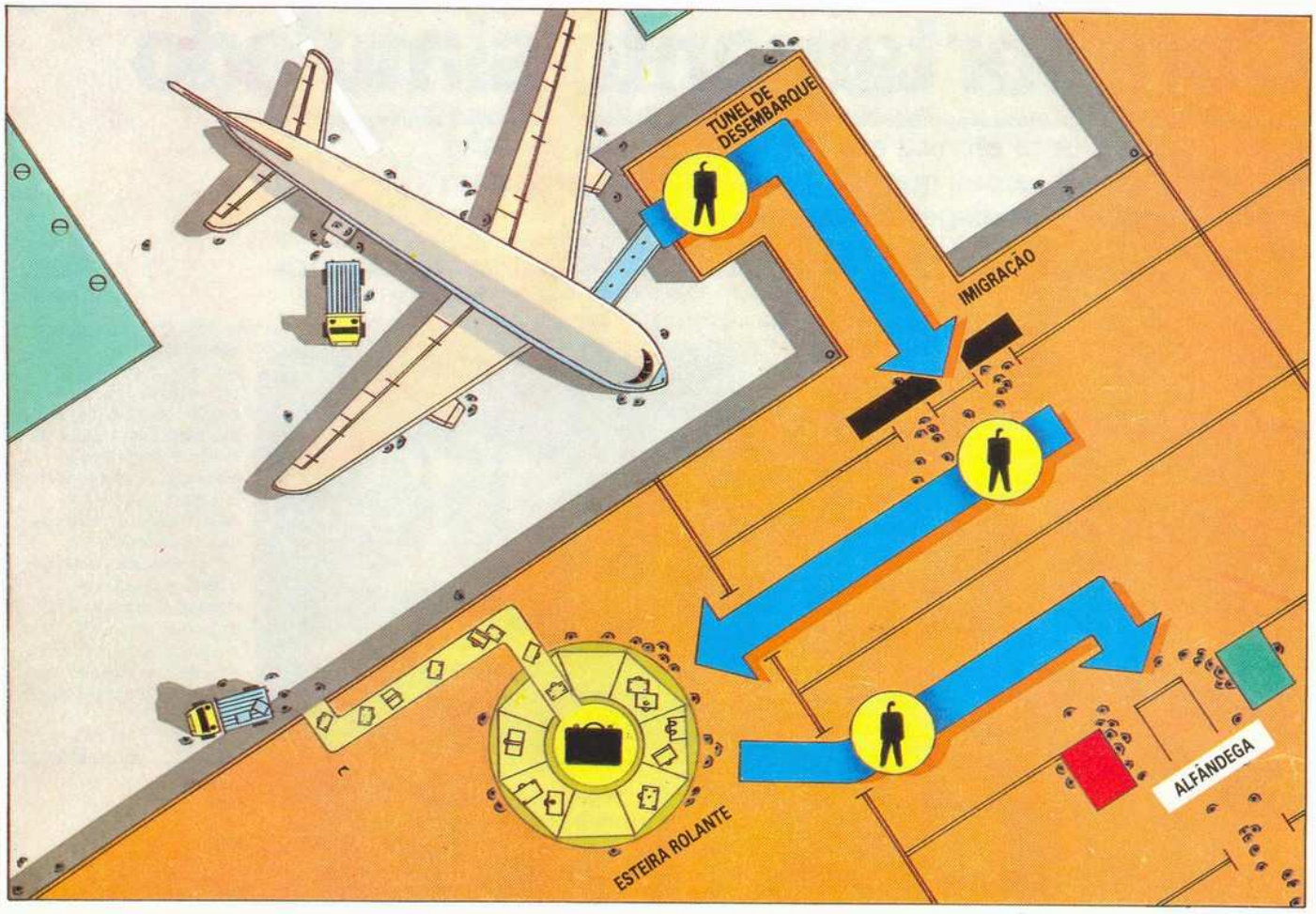
Sob controle
Aplicação importante da simulação dá-se na área escolar: os estudantes são exercitados em modelos de sistemas reais, como a simulação de uma usina de energia nuclear feita no Atari. O usuário controla os vários sistemas de refrigeração, para evitar que o reator esquente em demasia. O manual explica as funções dos vários mecanismos da usina e há uma demonstração do modo pelo qual o programa opera.

A quarta situação, em que o emprego de um modelo assume grande importância, ocorre quando um problema se restringe ao campo teórico e é impossível a experiência *in loco*. Os astrofísicos, por exemplo, especulam sobre a formação das estrelas utilizando modelos para avaliar uma teoria cosmológica — digamos, a teoria do "Big Bang" — em relação a uma outra hipótese.

Em toda simulação, a primeira coisa a fazer é organizar um modelo e decidir quais são os elementos importantes e como estão interligados.

Os sistemas e seus modelos dividem-se em dois grupos: sistemas fechados ou determinísticos, e sistemas estocásticos ou de conclusões abertas. Em um orçamento doméstico, as despesas podem ser distribuídas de várias formas, mas o livro de contas deve fazer o balanço no final — trata-se, neste caso, de um sistema determinístico. Entretanto, se a família gastar dinheiro de forma aleatória, sem considerar a quantia disponível em seu balanço, o sistema é chamado estocástico.

Por meio de simulações teóricas não se pode ter certeza absoluta de que o modelo escolhido é o mais correto. Imaginava-se que a Terra era o centro do universo, até Copérnico apresentar um modelo matemático bem mais simples, tendo o Sol no centro. Atualmente, os astrônomos, ao observarem as galáxias, constatam que todas estão se afastando de nós a velocidades crescentes, o que sugere, mais uma vez, que estamos no centro do universo. Mas, se adotarmos um modelo cósmico com a Terra no centro, cometeremos um equívoco, como mostra o seguinte



Um problema Jumbo
A simulação em computador foi usada no projeto de um novo terminal do aeroporto de Heathrow, em Londres. O programa simulava, primeiro, o tráfego de aviões, com o número dos passageiros e bagagens. Depois simulava os "processos" a que os passageiros seriam submetidos. O modelo permitiu, assim, adequar as dimensões do terminal projetado.

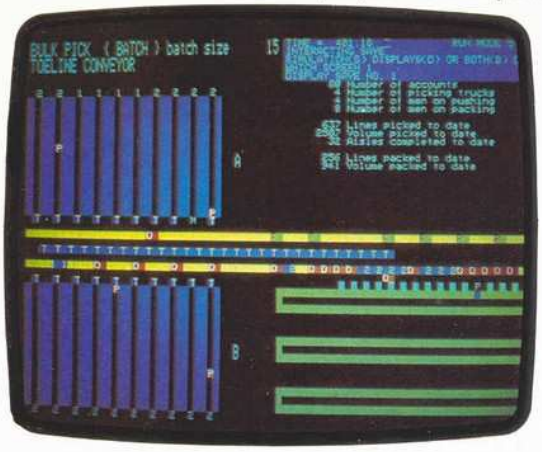
modelo alternativo. Se você respingar uma bexiga de plástico com pontos de tinta para representar as estrelas e, a seguir, inflar a bexiga, vai notar, evidentemente, que os pontos de tinta se afastam uns dos outros, e nenhum deles está no centro do balãozinho.

Entretanto, há muitas vantagens no uso de modelos. Eles ajudam a formular melhores teorias, abreviam o tempo das análises, permitem novas tentativas e modificações e, o mais importante, são bem mais baratos que uma experiência real. O uso de modelos também permite chegar a novas descobertas. O laser, por exemplo, foi inventado quando se pesquisava um aspecto de um modelo matemático para o qual não se havia dado anteriormente atenção.

A BL Systems, subsidiária da Leyland inglesa, pôs à venda um sistema de simulação de múltipla finalidade, que foi originalmente desenvolvido para projeto de linhas de produção e depósitos automatizados, em duas cidades. O sistema é chamado "Veja por quê", e utiliza saídas gráficas para mostrar resultados em vez de tabelas comuns de estatística. O sistema tem tido boa aceitação e foi usado pela Administração de Aeroportos Ingleses para modelar um novo terminal em Heathrow.

Um problema típico que pode ser examinado por meio da simulação é a ordem de chegada de aviões. Num aeroporto, se o vento mudar de repente, e apenas uma das pistas de pouso estiver disponível, os aviões terão inevitavelmente de formar fila. Eles dispõem de reserva limitada de combustível, e leva algum tempo até cada um poder aterrissar. Assim, as regras de enfileiramento serão programadas no sistema. Nessas simulações, geradores de números randômicos (ver p. 209) são usados para criar eventos inesperados, como pousos casuais.

A simulação é uma área importante de aplicação para os computadores digitais e resultou em novas linguagens, especialmente escritas para a simulação de projetos (por exemplo, GASP, SIMSCRIPS e GPSS). À medida que as atividades humanas se tornam mais complexas, o uso de modelos para simular problemas tem-se revelado cada vez mais importante.



Veja por quê
A BL Systems, uma divisão da Leyland inglesa, desenvolveu um pacote de modelos para microcomputador, com projetos de novas linhas de produção e instalações. Esse pacote é chamado "Veja por quê". Ele se caracteriza pela saída gráfica na forma de diagramas esquemáticos. Os números ao lado de cada etapa do processo indicam a progressão do trabalho e salientam qualquer problema.

A ordem da jogada

A capacidade de classificar dados é fundamental para a maioria dos programas e há muitos meios de consegui-lo.

Bubble Sort (Classificação Bolha)
O diagrama ilustra "a classificação bolha" para um leque limitado a nove cartas (D representa a carta 10). A parte ordenada começa da extremidade direita, a cada passagem. Os números 1 e 2 sob os conjuntos indicam as duas cartas que estão sendo comparadas.

```
Início da classificação
2 8 9 3 D 5 R 6 7 Início da
passagem 1
1 2
8 2 9 3 D 5 R 6 7
  1 2
8 9 2 3 D 5 R 6 7
    1 2
8 9 3 2 D 5 R 6 7
      1 2
8 9 3 D 2 5 R 6 7
        1 2
8 9 3 D 5 2 R 6 7
          1 2
8 9 3 D 5 R 2 6 7
            1 2
8 9 3 D 5 R 6 2 7
              1 2
8 9 3 D 5 R 6 7 2 Fim pass. 1
9 8 D 5 R 6 7 3 2 Fim pass. 2
9 D 8 R 6 7 5 3 2 Fim pass. 3
D 9 R 8 7 6 5 3 2 Fim pass. 4
D R 9 8 7 6 5 3 2 Fim pass. 5
R D 9 8 7 6 5 3 2 Fim pass. 6
Encerramento da classificação
```

Classificação por inserção
Por este método, a parte ordenada avança a partir da extremidade esquerda. As cartas são deslocadas diretamente para sua posição correta na lista à medida que são examinadas.

```
Início da classificação
2 8 9 3 D 5 R 6 7
2 1
8 2 9 3 D 5 R 6 7
2   1
9 8 2 3 D 5 R 6 7
    2 1
9 8 3 2 D 5 R 6 7
2     1
D 9 8 3 2 5 R 6 7
      2   1
D 9 8 5 3 2 R 6 7
2         1
R D 9 8 5 3 2 6 7
          2     1
R D 9 8 6 5 3 2 7
          2       1
R D 9 8 7 6 5 3 2
Encerramento da classificação
```

Embora uma das operações mais corriqueiras do computador, a classificação (sort) é uma tarefa em que ele se mostra — segundo seus próprios padrões — de pouquíssima eficiência. De acordo com pesquisas operacionais, cerca de 30 a 40% de todo o trabalho de computação é empregado em classificação e, se acrescentarmos as tarefas a ela vinculadas, de juntar os dados e buscar itens específicos, esse índice provavelmente ultrapassará os 50%.

Métodos aperfeiçoados de classificação são de entendimento extremamente difícil, mas torna-se bastante fácil compreender os mais simples, quando recorremos ao exemplo da ordenação de um baralho.

Coloque treze cartas do mesmo naipe em uma mesa, dispondo-as em uma linha, sem seguir uma ordem específica, de modo que o ás e o 2 não fiquem na extremidade direita da linha. As cartas devem ser dispostas em ordem decrescente (rei, rainha, valete... ás), a partir da esquerda. Esta é, para nós, uma tarefa banal. Entretanto, se for exigido que apenas uma carta seja deslocada por vez, que nenhuma carta seja colocada sobre outra e que todas ocupem o mínimo espaço possível, não será fácil determinar um método eficiente para sua execução. Nesta analogia, as cartas equivalem às unidades de dados, a superfície máxima ocupada corresponde à memória necessária do computador, e você representa o programa. Como resolver o problema?

1) Coloque uma moeda debaixo da carta situada na extremidade esquerda, para representar um marcador de posição e fazê-lo lembrar-se do ponto em que você se encontra no processo de ordenação ou classificação. Compare a carta marcada com a carta da direita. Estão em ordem decrescente? Se não estiverem, inverta a posição, deixando a moeda no mesmo lugar e seguindo a regra de mover apenas uma carta por vez e não colocar uma carta sobre outra. Observe o que você deve fazer para inverter a ordem.

2) Se as duas cartas estiverem na ordem correta, desloque a moeda um espaço para a direita e repita o procedimento 1. Você agora se encontra em um loop que se encerrará quando a moeda alcançar a extremidade direita da linha. O alcance dessa posição denomina-se fazer uma "passagem" pelas cartas.

3) No final da primeira passagem, examine as cartas. O ás, que é a carta mais baixa do naipe, atravessou a série até atingir a extremidade direita e está portanto na posição correta. Se você fizer outras passagens pela série de cartas, como foi dito nos procedimentos 1 e 2, a carta 2 terá alcançado sua posição correta. Esse procedimento repetido em sucessivas passagens resultará, por fim, na colocação de todo o naipe em ordem decrescente.

Você deve ter percebido vários inconvenientes nesse método. É muito enfadonho e toma tempo, pois a simples troca de posições entre duas cartas exige três operações; e, acima de tudo, muitas das comparações realizadas entre várias cartas são desnecessárias. Por exemplo, após uma passagem, o ás se encontra em seu lugar correto; assim, não adianta deslocar a moeda para a posição 13 (na qual, de qualquer forma, nenhuma comparação é possível). Na segunda passagem, visto que a carta da direita está no lugar correto, não há necessidade de deslocar o indicador para a posição 12. De modo geral, cada passagem terminará em uma posição à esquerda do ponto de encerramento da passagem anterior.

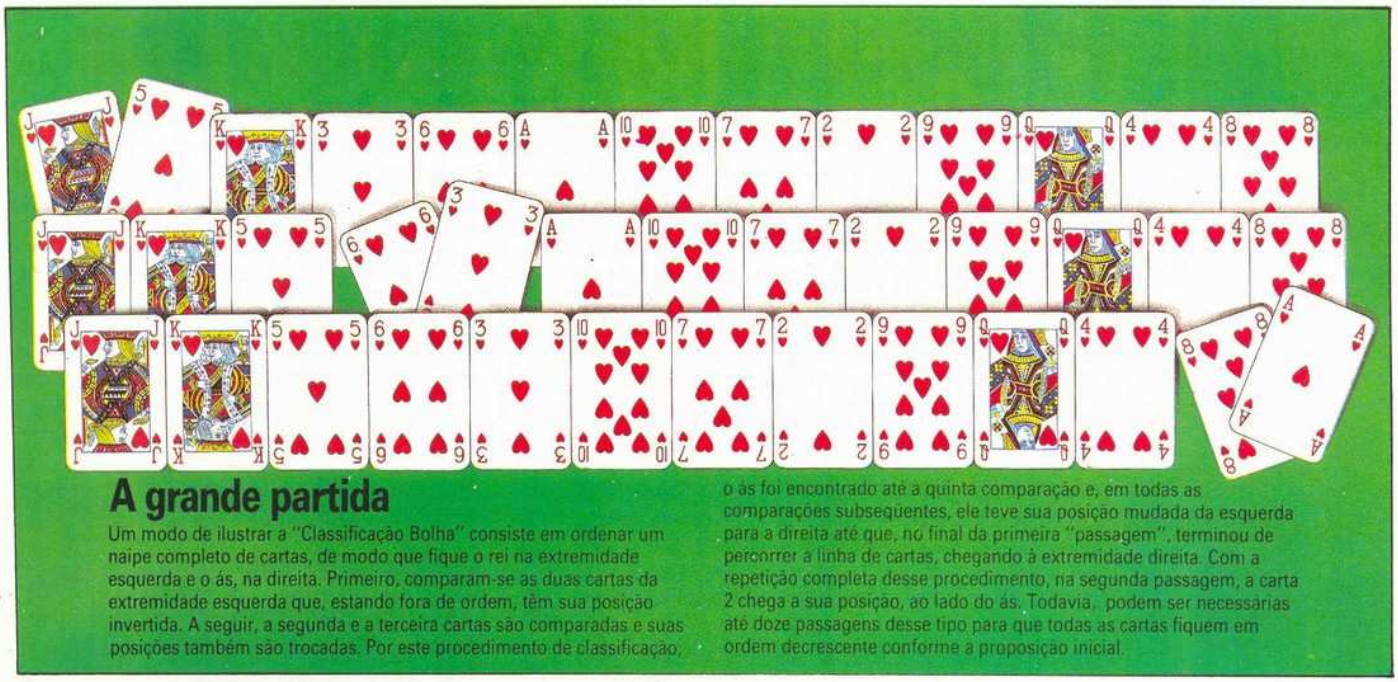
Determinar o ponto onde o procedimento deve ser interrompido é outro problema. Um computador continuará indefinidamente a comparar as cartas, a menos que receba instruções para parar. A única regra segura é a seguinte: pare após uma passagem sem inversões. Em outras palavras, se você fez a passagem pelos dados sem alteração na ordem, segue-se que eles estão na posição correta.

O método de ordenação que examinamos é chamado "Bubble Sort" ("Classificação Bolha"). Entre suas vantagens, estão as técnicas simples de programação, o uso limitado de memória auxiliar e razoável eficácia, com pequena quantidade de dados parcialmente ordenados. E é por esses critérios que os algoritmos de classificação devem ser avaliados. Não obstante, quando os dados a classificar são extensos, a velocidade possivelmente será comprometida em favor da economia de memória, porque a memória do computador talvez não comporte, ao mesmo tempo, os dados não processados e a cópia classificada. Por esse motivo, vamos ignorar os algoritmos que exigem a extração de dados de uma matriz e seu deslocamento para a posição classificada em uma segunda matriz. O segundo método de classificação simples é baseado mais diretamente no modo pelo qual colocamos cartas em uma determinada ordem.

1) Estenda outra vez na mesa as cartas embaralhadas e coloque uma moeda de 10 cruzeiros debaixo da segunda carta, a partir da esquerda. Todas as cartas sob as quais a moeda estiver, no início de cada passagem, serão chamadas "carta 10 cruzeiros"

2) Empurre a carta 10 cruzeiros para fora da linha, deixando uma lacuna, e coloque uma moeda de 50 cruzeiros debaixo da carta imediatamente à esquerda. Esta será chamada carta 50 cruzeiros.

3) Compare as duas cartas. Se estiverem em ordem, recoloque a carta 10 cruzeiros no lugar e passe para o procedimento 4. Caso contrário, coloque a carta 50 cruzeiros na lacuna e desloque a moeda de 50 cruzeiros para a posição à esquerda, a fim de marcar uma nova carta 50 cruzeiros (se esta estiver à extrema esquerda, este procedimento não será pertinente; neste caso, coloque a carta 10 cruzeiros na lacuna e siga para o procedimento 4).



A grande partida

Um modo de ilustrar a "Classificação Bolha" consiste em ordenar um naipe completo de cartas, de modo que fique o rei na extremidade esquerda e o ás, na direita. Primeiro, comparam-se as duas cartas da extremidade esquerda que, estando fora de ordem, têm sua posição invertida. A seguir, a segunda e a terceira cartas são comparadas e suas posições também são trocadas. Por este procedimento de classificação, o ás foi encontrado até a quinta comparação e, em todas as comparações subsequentes, ele teve sua posição mudada da esquerda para a direita até que, no final da primeira "passagem", terminou de percorrer a linha de cartas, chegando à extremidade direita. Com a repetição completa desse procedimento, na segunda passagem, a carta 2 chega a sua posição, ao lado do ás. Todavia, podem ser necessárias até doze passagens desse tipo para que todas as cartas fiquem em ordem decrescente conforme a proposição inicial.

Compare esta carta 50 cruzeiros com a carta 10 cruzeiros (a carta deslocada). Agora, repita o procedimento 3 até que a posição correta da carta 10 cruzeiros seja encontrada.

4) Desloque a moeda de 10 cruzeiros para a posição à direita e repita os procedimentos 2 e 3. Quando se tornar impossível deslocar a moeda de 10 cruzeiros para a direita, todas as cartas estarão na ordem correta.

Este procedimento é chamado "Classificação por Inserção" e é muito semelhante ao modo de as pessoas ordenarem um leque de cartas. Embora de programação um pouco mais difícil que a "Classificação Bolha", este método é muito mais eficiente. Depois, neste curso, veremos alguns algoritmos mais complexos para classificação de dados.

```
9  REM************************************
10 REM* ALGORITMOS DE
   CLASSIFICACAO*
11 REM************************************
100 INPUT "FORNECER O NUMERO DE
    ITENS A SER
    CLASSIFICADOS " ; LT
150 IF LT<3 THEN LET LT=3
200 LET LT=INT(LT)
250 DIM R(LT), C(LT)
300 LET Z=0:LET Q=0:LET P=0
350 LET I=1:LET O=0:LET II=2:LET TH=2
400 INPUT "DETERMINAR QUANTOS
    TESTES "; N
450 FOR CT=I TO N
500 GOSUB 4000
550 FOR SR=I TO TH
600 GOSUB 5000
650 PRINT:PRINT:PRINT:PRINT
700 PRINT "TESTE NO."; CT+SR/10
750 INPUT "PRESSIONAR TECLA DE
    RETORNO PARA INICIAR A
    CLASSIFICACAO " ; A$
800 PRINT "A LISTA NAO CLASSIFICADA E"
850 GOSUB 3000
900 ON SR GOSUB 6000, 7000
950 PRINT "A LISTA CLASSIFICADA E"
1000 GOSUB 3000
1050 NEXT SR
1100 NEXT CT
1150 END
2999 REM************************************
3000 REM* IMPRIMIR A LISTA *
3001 REM************************************
3100 FOR K=I TO LT
3200 PRINT R(K) ;
3300 NEXT K
3400 PRINT
3500 RETURN
3999 REM************************************
4000 REM* GERADOR RANDOMICO *
4001 REM************************************
4100 RANDOMIZE
4200 FOR K=I TO LT
4300 LET C(K)=INT(100*RND)
4400 NEXT K
4500 RETURN
4999 REM************************************
5000 REM* GERADOR RANDOMICO *
5001 REM************************************
5100 FOR K=I TO LT
5200 LET R(K)=C(K)
5300 NEXT K
5400 PRINT:PRINT
5500 RETURN
5999 REM************************************
6000 REM* BOLHA *
6001 REM************************************
6050 PRINT "CLASSIFICACAO BOLHA -
    INICIAR ! ! ! ! !
6100 FOR P=LT-I TO I STEP-I
6150 LET F=I
6200 FOR Q=I TO P
6250 LET Z=Q+I
6300 IF R(Q)<R(Z) THEN LET D=R(Q):
    LET R(Q)=R(Z):LET R(Z)=D:LET F=0
6350 NEXT Q
6400 IF F=I THEN LET P=I
6450 NEXT P
6500 PRINT "CLASSIFICACAO BOLHA -
    ENCERRAR ! ! ! ! !
6550 RETURN
6999 REM************************************
7000 REM* INSERCAO *
7001 REM************************************
7050 PRINT "CLASSIFICACAO POR
    INSERCAO - INICIAR ! ! ! ! !
7100 FOR P=II TO LT
7200 LET D=R(P)
7300 FOR Q=P TO II STEP-I
7400 LET R(Q)=R(Q-I)
7500 IF D<R(Q) THEN LET R(Q)=D:LET Q=II
7600 NEXT Q
7700 IF D>R(I) THEN LET R(I)=D
7800 NEXT P
7850 PRINT "CLASSIFICACAO POR
    INSERCAO - ENCERRAR ! ! ! ! !
7900 RETURN
```

Classificação em alta velocidade

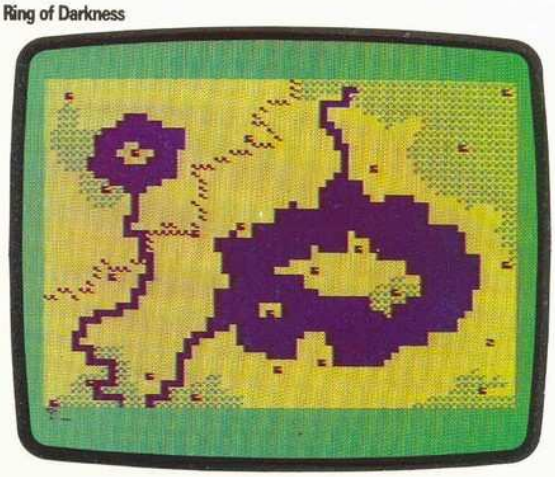
Este programa em BASIC mostra a diferença de eficácia entre a "Classificação Bolha" e o método por inserção. Como o código foi escrito tendo em vista a velocidade, não documentamos a operação das rotinas. A listagem pode ser processada na maioria dos equipamentos, mas veja na página 215 as adaptações no quadro "A propósito...", para a instrução ON...GOSUB, e, na página 175, para as funções RND e RANDOMIZE.

Procurando caminhos

Labirintos sempre exerceram fascínio sobre as pessoas — e os jogos de labirinto no microcomputador não fogem à regra.

Os labirintos, quer sejam grandes a ponto de as pessoas se perderem ou tão pequenos que caibam na palma da mão, sempre constituíram uma fonte de fascinação e divertimento tanto para jovens quanto para adultos. De fato, eles se tornaram a base de uma grande variedade de jogos de computador, que podem apresentar desde um labirinto visto de cima, bidimensional e muito simples, até outros extremamente complexos, em três dimensões. Estes últimos levam o jogador a imaginar que se encontra no interior de um labirinto de verdade. A fim de ajudá-lo a se orientar, ou então confundi-lo mais ainda, alguns desses jogos em três dimensões também mostram, de vez em quando e por alguns instantes, como eles seriam vistos de cima.

Ring of Darkness



Siren City
Este jogo do Commodore 64 é uma versão desenvolvida do tradicional jogo de "vista aérea". Um carro de polícia patrulha uma cidade.

Ring of Darkness
Embora este jogo para o micro inglês Dragon seja do tipo Aventura, apresenta um labirinto em três dimensões como um de seus principais elementos. Você anda para cima e para baixo, em fossos e escadas.

Way Out
Uma imagem realista em três dimensões é conseguida no Sinclair-Spectrum com o Way Out. Movimentando gradualmente o joystick, mudará também o cenário.

Como os efeitos sonoros e visuais dos labirintos se tornaram mais aperfeiçoados, a criatividade de seus programadores pôde desenvolver-se.

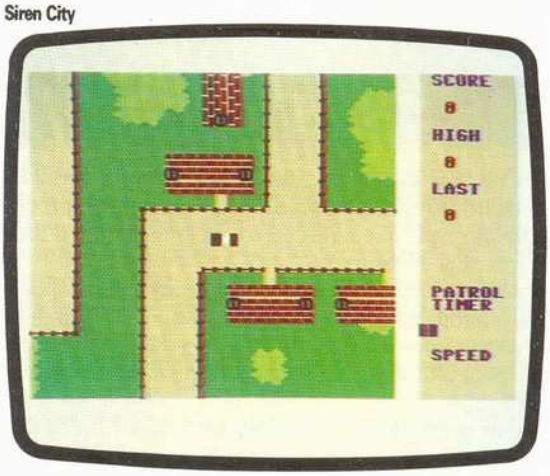
O jogador que quiser dar um passeio pelo labirinto deve evitar aqueles caminhos que escondem monstros devoradores. Um exemplo destes jogos é o 3D Glooper (próprio para o Commodore 64), onde o jogador caminha procurando ladrilhos especiais no chão e pode ser atacado a qualquer momento por monstros que ocupam a tela. No entanto, a chegada iminente dessas criaturas é anunciada pelo ruído de suas mandíbulas.

O Atic Atac (Sinclair-Spectrum) é uma perseguição totalmente animada, em que o jogador pode assumir a identidade de três personagens diferentes. O labirinto consiste em uma série de fossos, escadarias e grandes calabouços, em vários níveis, por entre os quais deve correr contra o tempo. Os calabouços são ocupados por uma variedade de estranhas criaturas e objetos.

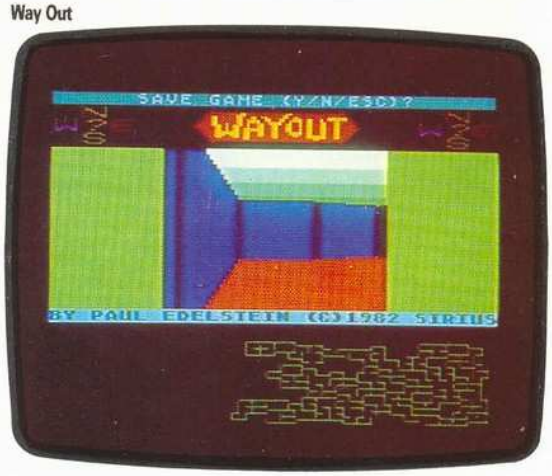
O programa que mais se aproxima da simulação do que realmente se sente ao tentar atravessar um labirinto é o Way Out. As coisas são vistas em verdadeira perspectiva tridimensional e, à medida que você move o joystick aos poucos para a esquerda e para a direita, o cenário vislumbrado muda proporcionalmente.

Vejamos algumas das técnicas básicas de programação usadas na construção de labirintos.

Siren City



Way Out



Construção de labirintos

A maneira geral de armazenar dados sobre um labirinto é usar uma matriz bidimensional. Cada elemento da matriz define as características daquele elemento do labirinto. Você poderia, por exemplo, usar uma seqüência de quatro caracteres para representar norte, sul, leste e oeste. Zero poderia indicar ausência de muro, e 1, a presença de um muro. Assim sendo, se M$(5,6) contém a seqüência "1011", isso indica que o elemento na linha cinco, coluna seis, está cercado por muros ao sul e leste.

Para poupar espaço na memória, a matriz pode ser numérica e não alfanumérica e o número de quatro dígitos considerado como um binário. No nosso exemplo, o elemento que contém muros ao norte, a leste e a sul seria representado pelo número 11 (1011).

Todos os elementos começam com quatro muros. Gerando de modo aleatório a entrada por qualquer ponto do perímetro, o próximo elemento será escolhido ao acaso entre qualquer um dos três adjacentes. Quando o elemento é escolhido, a seqüência continua — selecionando ao acaso um elemento de qualquer um dos três adjacentes, desprezando aquele de onde você veio.

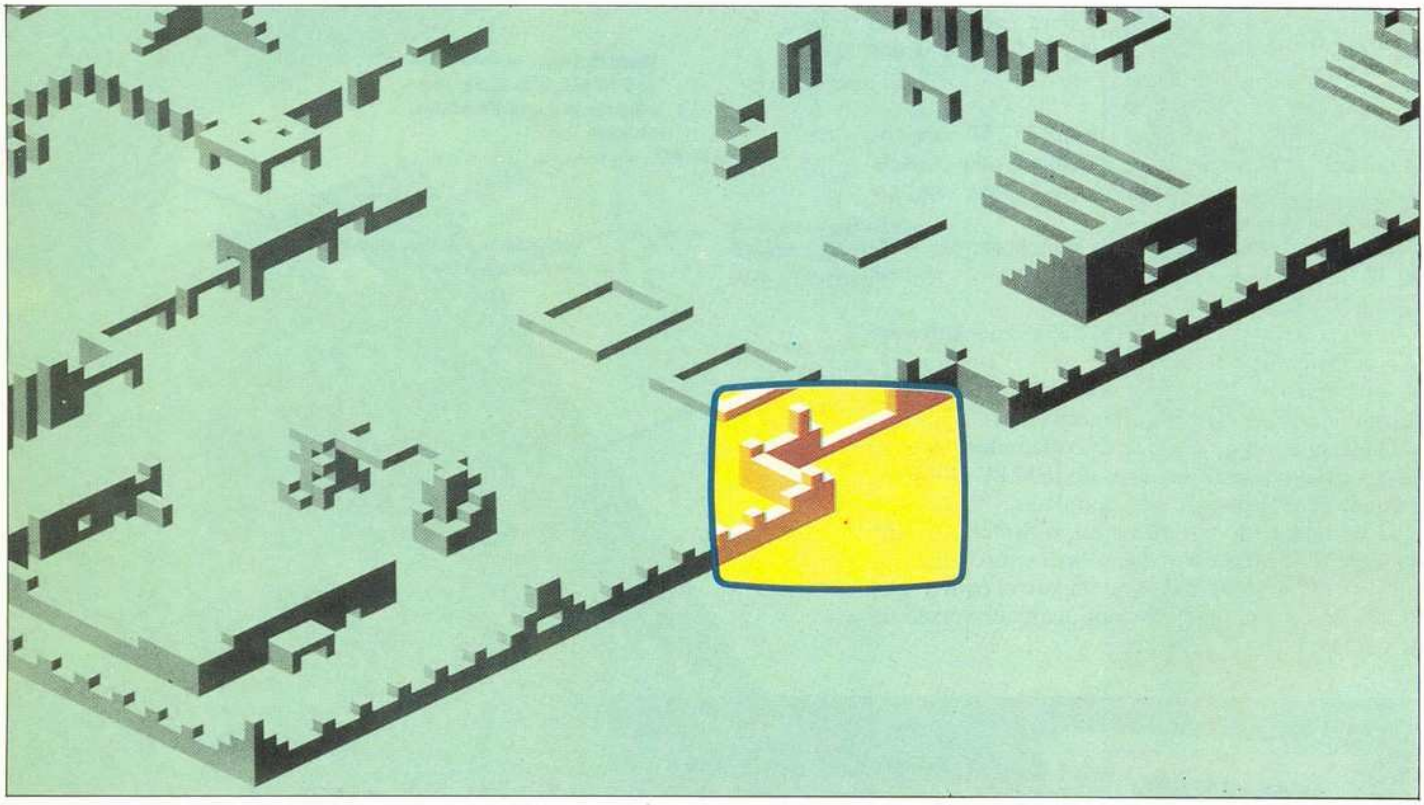
Toda vez que você escolhe uma nova direção, o muro existente entre o elemento em que você está e o elemento em que vai entrar é removido. É preciso prestar atenção para não sair dos limites do labirinto (a não ser que aquele determinado elemento do perímetro seja o ponto de saída), e não criar circuitos fechados (todas as partes de um labirinto devem ter acesso de qualquer outra parte).

Quando um elemento que tem menos de quatro muros (isto é, um elemento que já foi visitado) é encontrado, o programa deve escolher um dos elementos adjacentes que restarem. Se todos os elementos adjacentes já tiverem sido visitados, o programa deve "dar um passo atrás", voltando ao último elemento visitado, e tomar um novo caminho.

Outro método de gravar as características de um elemento consiste no uso mais sofisticado da numeração binária, que é particularmente útil para exibir perspectivas em três dimensões. Há dezesseis maneiras possíveis de se construir um elemento: sem muros, com muros de todos os lados, com um único muro (quatro possibilidades), muros em lados opostos (duas possibilidades), dois muros adjacentes (quatro possibilidades) e três muros adjacentes (quatro possibilidades).

Usando o número binário apropriado (0-15) para cada uma dessas possibilidades, pode-se "girar" o número para a esquerda ou para a direita até chegar à visão que o jogador realmente teria. Por exemplo, um muro face norte pode ser representado por 2 (0010), face sul, por 8 (1000), face leste, por 1 (0001) e face oeste, por 4 (0100). Se o jogador, estando voltado para o norte, num compartimento que tem apenas um muro face oeste (4), se voltar para oeste, sua visão do compartimento será como se ele estivesse diante de um muro face norte. Quando o jogador vira para a sua esquerda (oeste), o padrão do bit muda uma casa à direita, preenchendo a descrição que queríamos, isto é, o binário do muro face oeste 0100 (decimal 4) transforma-se no binário 0010 (decimal 2, um muro face norte!). Quando viramos à direita, os bits viram na direção oposta, e se virarmos duas vezes eles darão meia-volta. É necessário, sem dúvida, incluir um sistema para "embrulhar" os bits que caem das duas pontas do meio-byte durante este processo; caso contrário, as características que identificam um elemento serão mudadas toda vez que o jogador se virar para algum lado. Um elemento originalmente definido como 0011, por exemplo (muros face norte e leste), deve transformar-se em 0110, se o jogador virar à direita, e 1100 se ele der meia-volta.

Em linguagem de máquina há instruções especiais para girar binários para a esquerda e direita. Em BASIC, um binário de 4 bits expresso como um número decimal entre 0 e 15 pode ser virado à esquerda multiplicando-se o número por 2 e então subtraindo-se de 15, se o resultado passar de 15. Para girá-lo à direita, se o número for par, basta dividi-lo por 2 e, se for ímpar, soma-se 15 e divide-se por 2.



Ant Attack
Quando este jogo é executado pelo Spectrum, a tela do computador atua como uma janela no amplo campo de jogo em forma de labirinto. À medida que o jogo transcorre, a tela se movimenta mostrando outros aspectos do cenário.

Quadro de avisos

O que se pode fazer com um modem? Ele permite, além da comunicação direta entre proprietários de micros, o acesso a bancos de mensagens.

Um microcomputador em si tem limitada capacidade de comunicação. Mas através do telefone ele pode "falar" com uma série de outros computadores, sejam eles de grande porte, instalados em universidades, ou grandes bancos de dados abertos ao público — como o Videotexto — ou simplesmente o microcomputador do vizinho.

Primeiros passos no Brasil

Também no Brasil os quadros de avisos eletrônicos começam a entrar em funcionamento. A primeira comunidade informatizada do país — o Projeto Ciranda, da Embratel — já dispõe de um serviço desse tipo. Mais de 2.000 funcionários da estatal brasileira de telecomunicações com micros ligados ao sistema trocam recados entre si, não só numa mesma cidade, mas através de todo o território nacional. Basta para tanto que o interessado acione no seu micro o comando que dá acesso à "caixa de mensagens" do Ciranda. E este serviço deverá ficar à disposição de qualquer usuário de computador pessoal. A Embratel está abrindo o sistema para o público em geral. É o Projeto Cirandão, que entrou em operação em agosto de 1984. A Telesp, com seu Videotexto, também deverá dispor de um esquema semelhante. No início de 1984, já havia vários interessados em fornecer o serviço. Quando ele entrar em funcionamento, será possível a uma empresa com subsidiárias em diversas cidades reunir seus executivos sem deslocá-los para a sede. Os assuntos em pauta serão discutidos através dos micros ligados ao telefone, com uma vantagem adicional: as reuniões não terão horário preestabelecido, podendo estender-se por várias horas ou mesmo dias.

Na Inglaterra, com o Bulletin Board Services (Serviço de Quadro de Avisos) ou BBS, é possível deixar mensagens para outros usuários de microcomputadores, anunciar qualquer objeto à venda e até trocar programas. Um BBS equivale aos quadros de avisos forrados de cortiça encontrados em clubes e locais de trabalho. Geralmente são instalados e operados por voluntários que usam micros comuns, acrescidos de um disco com grande capacidade de armazenamento, e qualquer pessoa pode ter acesso a eles. A distância até outro computador não constitui problema: pode-se ter acesso a uma máquina na mesma cidade, no extremo do país, ou além-mar. A única restrição, obviamente, é a conta telefônica.

No entanto, antes que seu micro comece a "falar" no telefone, você precisa de uma peça de hardware para conectar os dois aparelhos. Esse dispositivo é o modem (ver p. 108), ligado na tomada serial (RS232), atrás do computador. Alguns micros não têm essa tomada e nesse caso você deve comprar um cartão de interface serial para substituí-la. O modem é ligado à linha telefônica de duas maneiras: diretamente, usando-se uma tomada extra de telefone, ou acusticamente, mediante um acoplador acústico. Há duas "velocidades" de modem (taxas de baud) de uso generalizado: 300 bauds e 1.200/75 bauds. Na Inglaterra, os operadores comerciais, tais como a Prestel e a British Telecom, usam a velocidade mais alta de 1.200/75, enquanto o BBS (usado por microcomputadores) utiliza 300 bauds. Significa 1.200/75 que a informação é enviada ao usuário a 1.200 bauds, para o computador central a 75 bauds. Por certo, é preferível um modem já adaptado para operar com as duas taxas de baud. O preço desse equipamento vem caindo consideravelmente, e pode-se adquirir um modem de 300 bauds, que funciona com bateria, por 150 dólares. Para o modem do tipo "conexão direta", aluga-se na British Telecom uma tomada extra para ligá-lo.

Na operação de um modem é necessário dispor de software apropriado, para que a máquina funcione como um terminal de computador. Há dois tipos de terminal — "dumb" (burro) e "smart" (inteligente). O "inteligente" pode carregar programas e guardar mensagens de outros computadores. Um "burro" não consegue fazer isso, mas é mais fácil de ser instalado e portanto mais adequado para as primeiras tentativas de comunicação. Este tipo de software é usado em microcomputadores "mais antigos", tais como o Apple II e o Tandy TRS-80, e em alguns modelos mais novos — o BTERM, por exemplo, funciona só com o BBC Model B. Para outros computadores, este software costuma ser distribuído gratuitamente pelos grupos de usuários. Quem não conseguir encontrá-lo, terá de escrever seu próprio software "emulador de terminais". O que se precisa é de um programa que envie todos os caracteres digitados no teclado para a interface RS232 e mostre na tela a informação recebida através dessa interface.

Para ilustrar como se usa um modem, passemos às diversas etapas do processo de comunicação com o BBS. Primeiro, precisa-se saber o número do telefone e os detalhes técnicos (como a taxa de baud) do computador ao qual se quer ter acesso. Após carregar e executar o software de comunicações no microcomputador, disca-se o número do telefone. Se o computador remoto estiver "escutando", ouve-se um sinal agudo no fone — é o "sinal de entrada" Usando-se um acoplador acústico, basta encaixar o fone nos bocais de borracha. No caso de um modem de conexão direta, aciona-se rapidamente o interruptor "LINE" ou "DATA" e recoloca-se o fone no gancho. De uma maneira ou de outra, a ligação será completada.

A primeira coisa que se vê na tela é uma "saudação" gerada pelo computador do outro lado da linha. Então, ele pede a identificação e/ou senha do usuário. Se o serviço chamado for privado e o usuário não for o assinante, provavelmente o usuário não

TELECOM GOLD

Correio eletrônico
O Telecom Gold é um sistema de correio eletrônico da British Telecom. Trata-se de uma versão mais sofisticada do Bulletin Board Services, que tem como primeiro objetivo atender o mercado comercial. Os usuários inscritos podem alugar uma "caixa do correio" em um minicomputador da British Telecom, onde outros usuários deixam mensagens Estas podem então ser lidas ou copiadas no computador do usuário. Pelo Telecom Gold tem-se acesso a outras redes eletrônicas em diversas partes do mundo.

irá muito longe — a não ser que seja muito bom em adivinhação ou muito paciente! Entretanto, muitos computadores permitem um acesso restrito a "convidados" que não são usuários registrados, e portanto vale a pena tentar usar algumas palavras como NEWUSER, GUEST ou HELP.

O BBS não apresenta este tipo de problema. É aberto a todos e gratuito (exceto a tarifa da chamada telefônica). No instante da comunicação, fornecem-se o nome e a cidade quando solicitados, e, uma vez "inscritos", responde-se a perguntas sobre detalhes do microcomputador, tais como largura da tela ou marca. Com isso, o computador "anfitrião" fica capacitado a identificar o usuário em chamadas futuras e a ajustar o sistema para que funcione adequadamente.

Uma vez feito isso, recebem-se algumas informações sobre o serviço, tais como horários de operação, detalhes técnicos e um tempo limite para a chamada — talvez 30 minutos. Então, pode-se passar ao menu principal, que consiste em uma lista de meia dúzia de comandos que se escolhe, apertando a tecla com a letra inicial correspondente. Por exemplo, para se registrar como um "Novo Usuário" digita-se N; para encerrar digita-se G, de "Goodbye" Outros menus aparecerão e deve-se continuar selecionando as opções até chegar à desejada. O BBS parece uma árvore — começa-se no tronco, que é o menu principal, e escolhem-se diferentes "ramos" com cada submenu. A mesma idéia é usada pelo Videotexto e pela maioria dos outros bancos de dados.

A seção "Novo Usuário" é para aqueles que querem inscrever-se no BBS. O nome e o endereço são solicitados e pode-se escolher a própria senha para usar nas futuras chamadas. O comando "Informação" fornece detalhes técnicos sobre o sistema. A seção "Utilidades" informa a duração da chamada e fornece pormenores de outros serviços do Bulletin Board Services.

A seção de "Avisos" contém mensagens públicas que outros usuários colocaram no sistema. O usuário também pode colocar seus avisos. A seção específica de "Mensagens" é para que ele leia recados particulares que lhe foram endereçados. Da mesma forma, pode-se enviar mensagens para outros usuários ou a um grupo de usuários que têm algo em comum (por exemplo, todos os proprietários de TRS-80). Estas funções talvez sejam os aspectos mais atraentes do serviço e são provavelmente responsáveis pela sua crescente popularidade.

Um serviço diferente, chamado Rewtel, oferece um banco de dados especializado em componentes eletrônicos e informação relacionada. Também atende a pedidos de itens do estoque feitos diretamente através do teclado, porém só de assinantes. Diferente do BBS, neste sistema utilizam-se "palavras-chaves" para especificar a área de interesse do usuário. Por exemplo, quando se quer ajuda para usar o serviço de compras, deve-se digitar HELP REWSHOP. Ele também apresenta um serviço BBS: introduz-se a palavra-chave CHALK e pode-se deixar uma mensagem. As pessoas que não são assinantes também usam o serviço e têm direito a três minutos do sistema. Distel e Maptel são serviços de bancos de dados semelhantes, destinados principalmente a atender encomendas de equipamentos eletrônicos e de computação. A vantagem destes sistemas comerciais é que operam 24 horas por dia e não ficam tão ocupados como o Bulletin Board Services.

Os computadores de grande porte já estão "no telefone" há algum tempo, mas apenas recentemente este tipo de comunicação tornou-se possível para usuários de micros, devido à crescente sofisticação destes aparelhos e à queda nos preços dos modems. Nos próximos anos, é provável que o modem passe a ser um acessório tão comum para o micro como a impressora e o gravador cassete.

Troca de recados
Uma das aplicações mais gratificantes de um modem é o acesso aos quadros de avisos eletrônicos, que são versões computadorizadas dos tradicionais quadros de avisos de clubes. As mensagens podem ser "afixadas" para serem lidas por todos ou por determinadas pessoas com as senhas certas. São anunciadas reuniões e vendas de equipamento de computador de segunda mão. Pode-se até carregar jogos ou outras listagens de programas para disco ou cassete.

A toda velocidade

Dando atenção cuidadosa às variáveis e à estrutura, você pode acelerar a execução de quase todos os programas em BASIC.

O BASIC é, apesar do que dizem os críticos, uma linguagem versátil e um recurso pedagógico muito eficiente. Você pode desenvolver qualquer programa em BASIC, desde que seu equipamento tenha memória suficiente e o tempo de operação não seja fundamental. Entretanto, uma vez que o BASIC é interpretado e não compilado (ver p. 184), pode ser penosamente lento na execução dos programas, em especial aqueles que exigem uma mesma instrução traduzida e executada repetidamente.

A ordenação, por exemplo, é um processo altamente repetitivo: o procedimento se realiza em um loop, com loops menores alojados no interior do loop principal (ver p. 286). Se tivermos 100 itens a serem ordenados, o programa pode ter de fazer de 2.500 a 5.000 interações do loop. Uma ordenação em BASIC sempre será lenta, mas o modo como o programa for escrito poderá resultar em diferença significativa na velocidade de execução. Se uma instrução deve ser repetida 5.000 vezes e se durante a codificação pudermos economizar um centésimo de segundo do tempo de execução em cada repetição, haverá economia total de 50 segundos — uma melhora considerável para o usuário.

Para perceber a diferença entre a boa e a má codificação, você necessitará de um cronômetro e de um programa "banco de teste". Se possuir um Commodore, você poderá utilizar o próprio relógio do equipamento, com as variáveis correspondentes, TI$ e TI, como parte do programa banco de teste. Se seu computador não tiver relógio, você terá de utilizar um cronômetro para medir o tempo do código em execução. Também é uma idéia interessante fazer seu programa indicar o início e o encerramento da execução, através de um bip, de modo que você possa saber quando está em operação.

O programa banco de teste pode ser o seguinte:

```
1000 L=500
2000 PRINT "**INICIO**": REM "BIP"
       instruções aqui
2100 TI$="000000"
2200 FOR K=1 TO L
.....
2900 NEXT K:T9=TI
2950 REM "BIP" instruções aqui
3000 PRINT "******STOP******"
3100 PRINT "Levou ";(T9/60)," segundos"
```

As linhas 2100 e 3100 foram elaboradas para os usuários do Commodore. Para outros equipamentos, deve-se eliminá-las ou substituí-las, de acordo com o código correspondente. No espaço entre as linhas 2200 e 2900 colocaremos o código a ser cronometrado. Observe que as cronometragens serão referidas às repetições L, onde L representa o limite do loop. O teste da execução de apenas uma unidade de código será de pouca precisão, pois o relógio do equipamento mede unicamente em sexagésimos de segundo e há uma cronometragem superior também imposta pelo código do programa de banco de teste.

Aqui estão algumas regras gerais para desenvolvimento de uma linguagem BASIC eficiente, apresentadas em ordem aproximada de importância:

1. Evitar todos os procedimentos aritméticos no interior dos loops.

A potenciação (x^3, significando "x elevado à terceira potência", por exemplo) e as funções matemáticas (cos (y), significa "o co-seno do ângulo y", por exemplo) são operações particularmente lentas. A multiplicação e a divisão são procedimentos mais morosos que a adição e a subtração; porém, mesmo a mais rápida dessas operações (a adição) é relativamente lenta.

Introduza as seguintes linhas no programa banco de teste:

```
900 Z=1.1
2300 X=Z↑3
```

e efetue o processamento. Em nosso equipamento de teste, 500 repetições levaram 27,95 segundos. Agora substitua a linha 2300 por:

```
2300 X=Z*Z*Z
```

e realize o processamento. Levou 3,55 segundos — uma diferença sensível!

O exame complementar revelará o nível da potenciação em que se torna conveniente substituir a multiplicação repetida pela função de potenciação. Em nosso computador, isso se deu na 18.ª potência (quando X=Z↑18). Todavia, lembre-se de que para calcular $Z^{2.3}$, por exemplo, a multiplicação repetida será inútil, enquanto a função de potenciação (↑) operar com qualquer número real, inclusive os negativos.

Utilize o programa banco de teste para verificar a duração dos outros processos aritméticos e compare as alternativas. Por exemplo, o que é mais rápido: dividir um número por 2 ou multiplicá-lo por 0,5?

2. Utilize variáveis em vez de constantes numéricas.

Toda vez que uma constante numérica (7.280, por exemplo) aparece em uma instrução em BASIC, gasta-se tempo com a tradução do número para forma utilizável. Experimente esta linha:

```
2300 X=X+7280
```

Em nossa máquina levou 4,63 segundos para executar 500 repetições, enquanto:

```
900 C=7280
2300 X=X+C
```

levou 2,75 segundos para realizar o mesmo número de repetições.

3. Se você tiver de utilizar a instrução GOTO, procure fazê-lo saltando para a frente e não para trás no programa. Entretanto, se tiver de retroceder, salte para o início do programa, ao invés de algumas linhas para trás.

O mesmo se dá com as instrução GOSUB. Ao encontrar uma instrução GOTO ou GOSUB, o interpretador BASIC compara o número da linha meta com o número da linha em operação no momento. Se a meta for maior que a linha em operação, o interpretador simplesmente avança buscando-a, linha por linha, até ser encontrada. Porém, se a meta for menor que a linha em operação, a busca sempre se iniciará a partir da primeira linha do programa. Isso significa que poderá ser um procedimento mais eficiente colocar sub-rotinas e unidades freqüentemente utilizadas em uma das extremidades do programa. Acrescente 56 linhas REM no início do programa para dar-lhe a extensão comumente usada e experimente:

```
2300 GOTO 2400
2400 GOTO 2500
2500 GOTO 2900
```

Levou 2,33 segundos para 500 repetições, enquanto:

```
2300 GOTO 2500
2400 GOTO 2900
2500 GOTO 2400
```

levou 4,85 segundos.

4. Inicialize todas as variáveis na ordem da freqüência de acesso.

Os nomes de variáveis são armazenados pelo interpretador, em uma tabela de símbolos, na ordem em que aparecem no programa pela primeira vez. Quanto mais tarde uma variável aparecer na tabela, mais tempo será gasto para encontrá-la e para acessar seus conteúdos. Pela mesma razão, você deve evitar o emprego de uma nova variável se puder recorrer a uma já anteriormente utilizada pelo programa, porém não em uso no momento.

Se uma variável for utilizada no interior de loops alojados — como acontece com freqüência em programas de ordenação —, essa variável será acessada constantemente; assim, faça sua inicialização no começo do programa, antes de qualquer outra variável, com um valor fictício, se necessário:

```
1000 L=500:C=7280:X=0:Z=1.1
2300 A=0
```

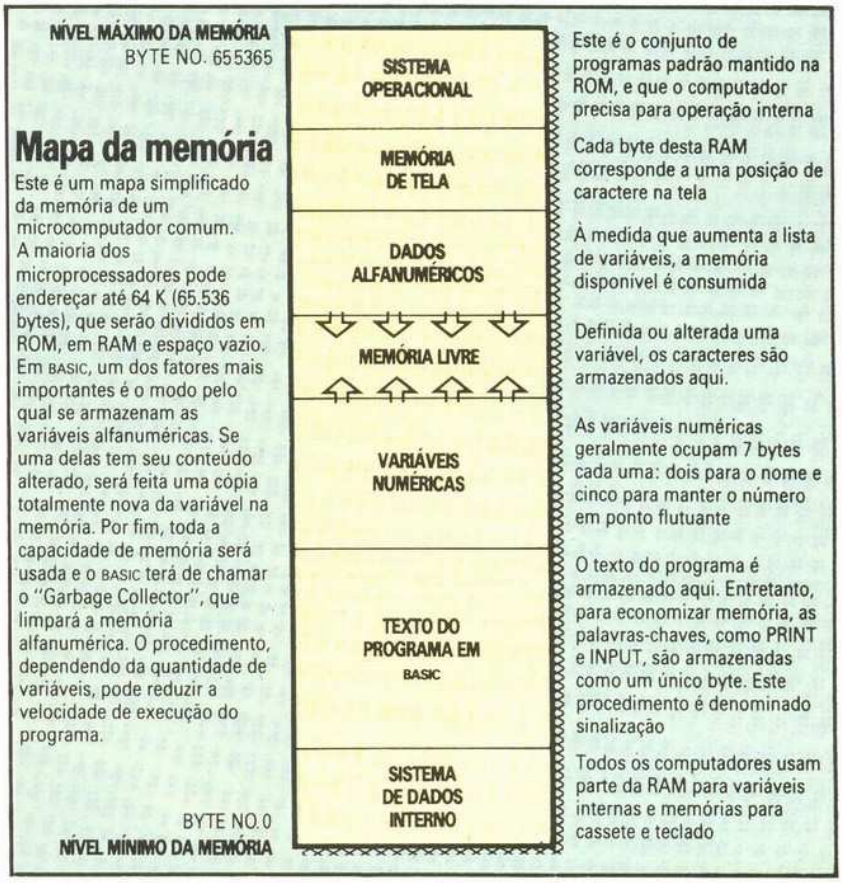
Levou 2,2 segundos para 500 repetições, enquanto:

```
1000 A=0:L=500:C=7280:X=0:Z=1.1
2300 A=0
```

levou 2,06 segundos.

5. Evite o emprego de variáveis alfanuméricas.

Operações com variáveis alfanuméricas utilizam a memória de modo aritmético diferente e um sistema de programação chamado "Garbage Collector" (coletor de lixo) pode ter de ser chamado ocasionalmente pelo interpretador para organizar os conteúdos de memória alfanumérica. Esse procedimento pode exigir muito tempo.



Uma demonstração genérica desse processo é difícil de ser escrita porque os computadores variam muito em seu sistema de gerenciamento de memória: você terá de preencher a maior parte da memória disponível para o usuário com dados — uma grande matriz numérica fará isto — e então executar manipulações com variáveis alfanuméricas que façam com que o Garbage Collector seja requisitado. Fornecemos a nossa máquina os seguintes dados:

```
40 POKE 52,32:POKE 56,32:CLR
```

para reduzir bastante o total de memória disponível para os programas em BASIC e então fornecemos:

```
1000 L=500:DIMT$(L)
1100 FOR K=1 TO L
1200 T$(K)="A"+"B"
1300 PRINT K
1400 NEXT K
```

que utiliza grande quantidade de memória alfanumérica e fornece uma matriz alfanumérica para uso posterior. As instruções PRINT são executadas em cada interação, apresentando o valor do contador do loop. Quando processamos essa versão do banco de teste, a impressão interrompeu-se repetidamente para chamada do Garbage Collector a fim de reordenar a memória. Algumas vezes a interrupção demorou mais de três segundos. O programa continua:

```
2300 A$=LEFT$(T$(L),1):B$=A$+RIGHT$(T$(L),1)
```

Isso exigiu um total de 30,03 segundos em 500 repetições. Quando processamos o mesmo programa com muito mais memória disponível, a garbage collection não foi percebida em operação e o loop cronometrado levou 8,66 segundos.

Idiomas diferentes

Fácil de aprender, porque tem uma construção matemática familiar, o BASIC, no entanto, é grosseiro quando comparado a outras linguagens.

Existe uma grande probabilidade de que o seu computador use o BASIC como linguagem de programação. Mas isto não quer dizer que você tenha de se restringir a esta escolha e, embora o BASIC seja considerado particularmente fácil de se aprender, há outras linguagens mais adequadas para escrever programas de aplicações específicas.

Para incorporar essas linguagens a seu computador, será necessário substituir as ROMs que contêm os interpretadores BASIC, ou carregar a nova linguagem em RAM — neste caso, você precisa de uma máquina com capacidade razoável de memória, de forma que sobre espaço de RAM para conter seus programas. Alguns microcomputadores, como o MZ-711 da Sharp, resolveram este problema carregando até mesmo o interpretador BASIC em fita cassete.

PASCAL-PORTUGUÊS

PORTUGUÊS-PASCAL

A linguagem PASCAL foi estruturada no começo da década de 70, para suceder ao BASIC. Sua variedade de dados e estruturas de controle deriva do grupo de linguagens FORTRAN/ALGOL, que tem por finalidade estimular o estudante a fazer a programação do computador de uma forma sistemática e a escrever códigos bem estruturados e de fácil entendimento. Este é um ponto necessário para o desenvolvimento de técnicas de programação de boa qualidade, mas significa que os primeiros estágios do aprendizado de programação são mais difíceis para o iniciante.

Eis aqui uma amostra de um programa em PASCAL, equivalente ao programa em BASIC:

```
VAR                                BEGIN
NOME          :PACKED               EXEC:=TRUE;
              ARRAY (1..30)         WHILE EXEC DO
              OF CHAR;                BEGIN
                                      WRITE ('Qual e o seu nome?');
IDADE,CONT :INTEREGER;                READLN (NOME);
RESPOSTA   :PACKED                    WRITE ('e qual sua idade?');
              ARRAY (1..3)            READLN (IDADE);
              OF CHAR;                FOR CONT:=1 TO IDADE DO
                                        WRITE (CONT:3, 'Alo':10, NOME);
EXEC          :BOOLEAN;               WRITE ('Quer mais uma vez?');
                                      READLN (RESPOSTA);
                                      IF RESPOSTA (1)='N'
                                       THE RUNNING:=FALSE;
                                       END;
                                    WRITELN ('Adeus', NOME)
                                    END
```

BASIC-PORTUGUÊS

PORTUGUÊS-BASIC

A linguagem BASIC, muito difamada, foi desenvolvida a partir do FORTRAN (uma das primeiras linguagens de programação de alto nível e ainda a mais popular para aplicações científicas e de engenharia) como uma introdução tutelar à programação para estudantes universitários. Por ter sido criado para um uso na forma autodidática, o BASIC é geralmente mais interpretado que compilado (ver p. 184), e este foi o fator de haver se tornado a linguagem original de quase todos os microcomputadores. Linguagens interpretadas são fáceis de se implantar, usam comparativamente pouca memória do computador e são bastante adequadas para desenvolvimento de programas.

O BASIC é uma linguagem forte por si só, mas tem o inconveniente de uma variedade de dialetos não padronizados (todo BASIC de um computador é exclusivo da máquina) e de lhe faltarem dados e estruturas de controle especializados.

Este programa curto ilustra não só a atração e a generalidade, mas também as limitações do BASIC.

```
100 INPUT "Qual e o seu nome?";N$
200 INPUT "e qual sua idade?";I
300 FOR K = 1 TO I
400 PRINT K, "Alo";N$
500 NEXT K
600 INPUT "Quer mais uma vez?";R$
700 IF LEFT$(R$,1) = "S" THEN GOTO 100
800 PRINT "Adeus";N$
900 END
```

COMAL-PORTUGUÊS

PORTUGUÊS-COMAL

A linguagem COMAL foi organizada para combinar a acessibilidade do BASIC com as sólidas estruturas e a execução disciplinada do PASCAL. Assim, ela se parece com as duas e teria servido de modelo para o desenvolvimento do BASIC específico do micro inglês BBC, que quase se transformou em uma nova linguagem. O COMAL é muito usado na computação escolar e na Escandinávia (seu lugar de origem), mas parece improvável que venha a substituir uma de suas antecessoras como linguagem introdutória à programação.

Este é o programa "Alô", em COMAL:

```
100  EXEC:=TRUE
200  WHILE EXEC DO
300     INPUT "Qual e o seu nome?":N$
400     INPUT "e qual sua idade?";I
500     REPEAT
600        FOR K:=1 TO I DO
700           PRINT K,"Alo";N$
800        NEXT K
900     INPUT "Quer mais uma vez?";R$
1000    IF R$="N" THEN EXEC:=FALSE
1100 ENDWHILE
1200 PRINT "Adeus";N$
```

LISP-PORTUGUÊS

PORTUGUÊS-LISP

O LISP foi desenvolvido no começo da década de 60 como uma linguagem de processamento de listas, e desde então tem sido amplamente utilizado na área da inteligência artificial, que envolve contínua busca e comparação de listas de dados, conexões e respostas. Ao contrário do BASIC, no qual se destaca o fluxo do programa por uma seqüência de instruções e procedimentos, o LISP é uma linguagem "funcional", em que o conjunto básico de comandos pode ser desenvolvido para construir funções mais sofisticadas, com nomes definidos pelo próprio programador. Por exemplo:

```
(SETQ MATRIZ1 '(4 7 2 5 1))
```

cria uma lista chamada MATRIZ1, cujos elementos são os números (4 7 2 5 1)

```
(CAR MATRIZ1)
```

fornece o primeiro elemento da lista MATRIZ1 (4, neste caso).

```
(CDR MATRIZ1)
```

fornece a lista MATRIZ1 com o primeiro elemento removido — (7 2 5 1) neste caso.

```
(SETQ MATRIZ1 (CDR MATRIZ1))
```

transforma a lista MATRIZ1 em uma cópia de si mesma, excluindo o primeiro elemento.

O LISP também é útil para aplicações "recursivas" — casos em que certos problemas, após a aplicação de uma simples função, são reduzidos a problemas menores, porém idênticos.

LOGO-PORTUGUÊS

PORTUGUÊS-LOGO

A linguagem LOGO foi estruturada pela equipe do professor Seymour Papert, que trabalhava com inteligência artificial. Ela se parece com o FORTH, tanto na interatividade quanto no uso de um número de funções "primitivas", que podem ser incorporadas em funções definidas pelo usuário. Esta linguagem é baseada em um princípio fundamental: uma maneira de aprender algo é ensinar alguém como fazê-lo — neste caso, o computador. O LOGO cria um método completamente novo de ensinar crianças a pensar. Costuma ser chamado de linguagem "tartaruga", porque é usado para controlar um pequeno robô sobre rodas, a tartaruga (ver p. 34).

Aqui apresentamos um fragmento em MLOGO (uma versão do LOGO em português), que desenha uma casa simbólica, traçando um quadrado de tamanho especificado, com um triângulo em cima:

```
AP TRIANGULO: TAMANHO
   REPITA 3 (FRENTE: TAMANHO DIREITA 120)
FIM
AP QUADRADO: TAMANHO
   REPITA 4 (FRENTE: TAMANHO DIREITA 90)
FIM
AP CASA: TAMANHO
   DIREITA 30
   TRIANGULO: TAMANHO
   ESQUERDA 90
   QUADRADO: TAMANHO
FIM
```

Se agora teclarmos CASA 15, aparecerá o desenho de uma "casa" cujos lados medirão 15 unidades.

FORTH-PORTUGUÊS

PORTUGUÊS-FORTH

A linguagem FORTH se parece com o LOGO por ser funcional e interativa, mas apresenta uma diferença importante: é a primeira, depois do BASIC, a ser implementada num computador pessoal — o micro inglês Jupiter Ace. Ela consiste em um número de funções definidas, chamadas "primitivas", e tem a capacidade de definir novas funções a partir daquelas. As operações matemáticas em FORTH são "stack-oriented", o que significa que a memória do computador é tratada como uma lista de dados que aumenta e diminui e, como resultado, a última operação sempre encabeça a lista. Outra consequência deste método é que a notação algébrica não é usada. Em lugar de escrever (12 + 4)/2 para achar a média de 12 e 4, em FORTH escreve-se 12 4 + 2 /, que é a mesma quantidade em "notação inversa", em vez de notação algébrica.

Tudo isso transforma o FORTH num tipo de linguagem muito diferente. Ele pode ser considerado quase um retrocesso na hierarquia das linguagens de alto nível.

Este fragmento em FORTH define duas novas palavras chamadas GRITO e CORO:

```
:GRITO (prints "SHAZAM!")
   ."SHAZAM!";
:CORO (uses GRITO in a loop)
   O DO GRITO LOOP;
```

Agora, se teclarmos n CORO, aparecerá SHAZAM! escrito n vezes na tela.

Nestas páginas apresentamos um apanhado geral das linguagens de programação mais comuns para micros pessoais. Assim como os idiomas falados, quanto mais linguagens de programação você dominar, mais fácil será aprender uma nova.

Faz de conta

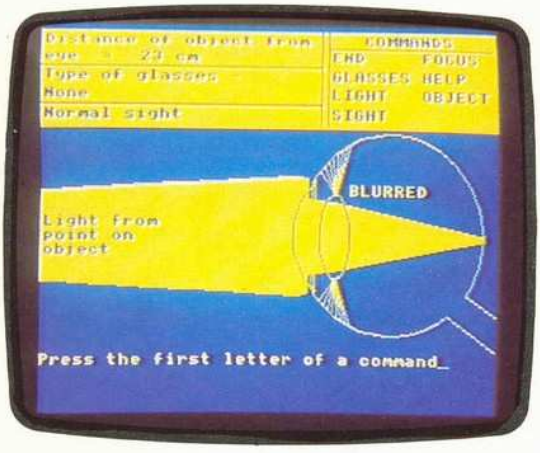
Experiências sem amostras, material, equipamento: isso é possível com programas de simulação, que divertem e ensinam.

Os programas de simulação, mesmo os triviais fliperamas que colocam o usuário no controle de um carro de corridas ou de um avião, são planejados para proporcionar uma experiência tão próxima quanto possível da situação real.

No entanto, existe grande quantidade de softwares de simulação com objetivos não apenas de entretenimento. Esses programas são muito úteis em várias áreas do currículo escolar, em particular em matérias como ciências, em que a experimentação prática mostra-se muito perigosa, exige muito tempo, é cara ou muito complexa.

Os programas de simulação podem ser úteis como recurso pedagógico também em casa. O Viagem de Carro, por exemplo, leva a criança a usar sua capacidade de raciocínio e seus conhecimentos de aritmética para "guiar" um carro através de um país. Os programas de simulação para fins educacionais são provavelmente o tipo de software mais excitante à venda no mercado internacional. Por outro lado, são compatíveis apenas com um número limitado de micros, como o Spectrum da Sinclair ou o Apple (os equipamentos preferidos em escolas).

Olho



Esse programa mostra como funciona o globo ocular e como suas partes devem estar corretamente ajustadas para que haja visão nítida, simulando o percurso que os raios luminosos fazem de um objeto até a retina (a parte posterior do olho, onde as imagens se formam). Você age como cérebro, controlando fatores como a distância do objeto, o tamanho da pupila e a distância focal do cristalino, de modo que a imagem na retina esteja clara. O vídeo mostra o corte transversal de um olho, com as partes importantes identificadas. Usando o comando LIGHT, assinala-se a trajetória de um raio de luz entre um objeto e o olho. Só quando as demais variáveis forem corretamente escolhidas, o usuário receberá pelo vídeo a mensagem EM FOCO.

Uma vez entendido o funcionamento do olho normal, pode-se simular defeitos visuais como a miopia, por exemplo. Ao perceber que é impossível a focalização de objetos distantes, o usuário deverá acrescentar uma lente diante do olho. Se for escolhida a lente adequada, a visão normal estará restabelecida.

Embora tenha sido desenvolvido, de início, para aulas de física e biologia, esse programa é empregado também como auxiliar em cursos genéricos de computação, como Introdução ao Computador para alunos mais jovens. E não será difícil entender o porquê disso se pensarmos na simplicidade do assunto, na facilidade de uso e nos excelentes recursos de detecção de erros de que o programa dispõe. Ele é produzido na Europa pela Longmans para o BBC Micro.

Balão

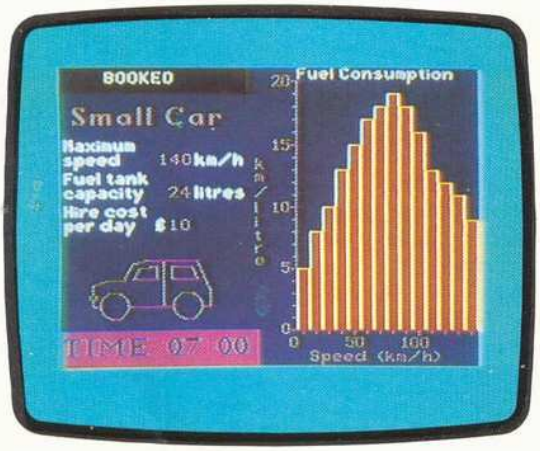
Trata-se de um programa educativo para uso doméstico. Encontrado em versões compatíveis com o computador Sinclair ZX Spectrum, é fabricado na Inglaterra pela Heinemann Educational Software. O usuário controla um balão de ar quente e deve fazê-lo voar. Na tela aparecem o perfil de uma região campestre com o balão, no solo, e os mostradores de quatro instrumentos: o indicador das velocidades de subida e descida, o medidor da temperatura do ar, o altímetro e o medidor de combustível. Existem dois controles: o do bico de gás para aquecer o ar e fazer o balão subir, e o da abertura, que deixa o ar sair e faz o balão descer.

Basta uma "lição de vôo" para ensinar o usuário a utilizar os instrumentos e controles, fazendo o balão decolar, voar e aterrissar. Domi-

nadas essas operações, você pode realizar sua "missão". Ela consiste em fazer o balão pousar em locais escolhidos (assinalados com um X), em que o balonista recebe algumas instruções. Uma das tarefas, por exemplo, é a de "ajudar o fazendeiro a encontrar carneiros" — os animais serão encontrados num campo marcado com S. Se o balão ficar sem combustível, deverá descer para receber novos cilindros de gás.

Bastam algumas decolagens experimentais e algumas colisões com árvores e outros obstáculos para se aprender a controlar o balão usando jatos curtos do bico de gás. Da mesma forma, não demorará muito até que você domine os instrumentos de modo a calcular quando deve ser usada a saída de ar ou o bico de gás. Talvez a vantagem mais importante seja aprender como controlar um sistema que inclui retardamento acentuado de tempo.

Viagem de Carro

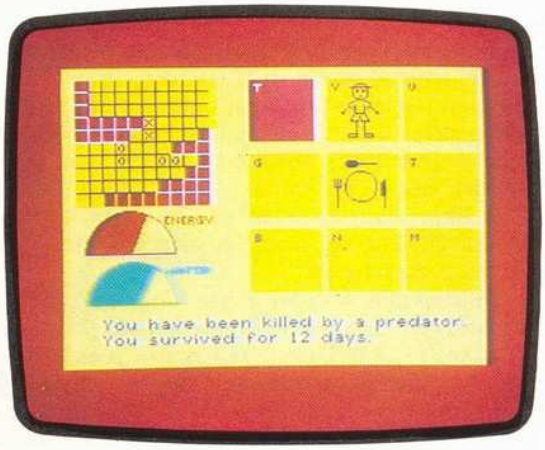


Compatível com o ZX Spectrum, é um programa educacional para uso doméstico, no qual o usuário faz o papel de proprietário de um pequeno serviço de entregas. São necessárias várias decisões relativas à escolha do contrato de entrega, à velocidade do carro e ao tipo de veículo a ser empregado. E, para tanto, o usuário precisa fazer cálculos que incluem dinheiro, distância, tempo, consumo de combustível. Um mapa do país aparece na tela, apresentando quinze cidades e as estradas principais. São também apresentados velocímetro, medidor de quilometragem, medidor de combustível e relógio.

A primeira tarefa é decidir por qual cidade começar e, então, será necessário escolher, entre as doze opções possíveis, um contrato. Um deles, por exemplo, consiste em receber uma remessa de diamantes às 12h e entregá-la em uma cidade a 350 km antes das 18h do mesmo dia. Para isso, você deve alugar um carro, ir ao local a fim de receber os diamantes e seguir para a cidade de destino. Se for bem-sucedido, receberá um prêmio em dinheiro e, se chegar antes do prazo, mais uma bonificação. O dinheiro deve ser gasto em pernoites, manutenção do carro, combustível, multas por excesso de velocidade ou por não cumprimento do contrato. Escolhendo um contrato para transporte de carga mais pesada, o carro deve ser substituído por um furgão, mais lento e de custo (aluguel e consumo de combustível) consideravelmente mais alto.

O programa ajuda a ampliar as capacidades mais abstratas de tomada de decisões e raciocínio lógico. Ensina até mesmo noções simples de Economia, pois, ao pensar nos prós e contras de determinado contrato, o usuário está fazendo uma análise de custo/benefício.

Sobrevivência



Se você já imaginou o que deve ser a vida de um leão, ou mesmo de um rato, o programa Sobrevivência foi feito para você. Ele possibilita ao usuário fazer o papel de seis animais (águia, passarinho, leão, rato, mosca ou borboleta) e viver problemas de sua existência cotidiana.

O mundo é representado com um quadriculado na tela e você deve mover-se através dele (sua posição é mostrada pela letra A) pressionando as teclas. Suas principais preocupações devem ser encontrar alimento (os quadrados assinalados por um O) e evitar predadores (assinalados por um X). Ao se aproximar de um desses quadrados assinalados, uma ampliação da área na qual você se encontra aparece no lado direito da tela, mostrando que predadores ou alimentos estão por perto. Além disso, surgem na tela dois medidores que indicam quanta energia e água você ainda tem. Se seu nível de energia baixar, você deverá procurar alimento; e, se a água acabar, será preciso buscar um quadrado azul (um rio); entretanto, se por acaso "cair" num quadrado azul, você se afogará.

Alguns animais têm vida mais difícil do que outros: a única fonte de alimento da borboleta são as flores, que podem ser difíceis de encontrar. A águia, porém, alimenta-se de cobras, moscas e ratos; por outro lado, pode ser abatida por um caçador. Trabalhando com esse programa, você aprende como várias espécies animais se encaixam na cadeia alimentar e avalia alguns dos problemas enfrentados pelos animais na luta selvagem pela sobrevivência.

Intérprete de papéis

A simulação tem revelado ótimas aplicações pedagógicas para microcomputadores, e são vários os programas disponíveis.

Ventos

Este programa de simulação, distribuído pela Longmans inglesa, coloca você na posição de comandante de uma antiga caravela (como aquelas empregadas por Cabral). Na tela aparece um mapa-múndi, onde sua embarcação é representada por um ponto que navega de acordo com a bússola — N, L, SO etc. Você escolhe a data em que será lançada ao mar, o porto de partida e o de chegada. A velocidade da nau depende da direção dos ventos predominantes; esta informação aparece no rodapé da tela, junto com a posição da nau expressa em latitude e longitude, os ventos da região (vento oeste, por exemplo), a data, a distância percorrida até o momento e a duração total da viagem.

Tomemos a rota mais direta de Londres ao Rio de Janeiro, com partida no dia 1.º de janeiro. Rumando para o sul, fazemos bom progresso aproveitando o vento oeste do Atlântico, até chegarmos à linha do Equador. Aí ficamos retidos por três dias devido à calmaria. Afinal, sopra um vento sudoeste, mas surge um problema — como podemos navegar rumo sudoeste com um vento sudoeste? A solução é navegar em ziguezague (ou "bordejar") primeiro para o sul e depois para oeste, até alcançar o Rio de Janeiro, depois de 207 dias de viagem, percorrendo 9.620 milhas (equivalentes a 15.480 km).

Outras viagens implicam inúmeros perigos: furacões, gelo polar e naufrágio constituem alguns dos riscos a que você e sua nau estão sujeitos. Esse programa de simulação pode ser usado de várias formas. Na mais simples, como um jogo para ensinar às crianças os pontos cardeais e colaterais. Numa aula de geografia, além de desenvolver o conhecimento do globo terrestre, os alunos podem examinar as diversas zonas de ventos e ter uma idéia de otimização de rota.

Simulação de vôo

Trata-se de uma versão para microcomputador de conhecido jogo de fliperama, onde o operador é o piloto de um pequeno avião. Você encontra na tela o que veria se estivesse na cabine de comando de um avião, inclusive o painel de instrumentos, onde aparecem mostradores, marcadores e luzes, que devem ser observados com atenção. Quatro teclas com setas representam o manche do avião; os demais controles (potência, trem de pouso etc.) são operados mediante outras teclas.

No jogo de fliperama seria impossível, mas nesse programa você tem a oportunidade de se familiarizar com os controles, começando com o avião em pleno ar. Para saber onde se encontra o avião, a tela mostra um mapa com a posição do aparelho, sinais de navegação e pistas de pouso e decolagem. A melhor forma de conduzir o aparelho é "fechar" num sinal de navegação e então inclinar o avião lateralmente em curva até que ele fique alinhado com o sinal. Isso é mostrado pelo ponto que pisca no "Relógio RDF" e que fica rodando até chegar ao topo do mostrador. Então você voa em linha reta até o sinal. Usando esse método, será fácil chegar ao aeroporto onde você tentará pousar.

A aterrissagem constitui a manobra mais difícil. Você precisa estar alinhado com a pista e aproximar-se na velocidade, na altura e no ângulo certos. E você ainda pode cometer um erro fatal — esquecer de baixar o trem de pouso!

Simulação fisiológica

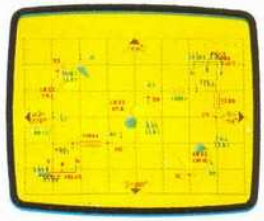
Você faz o papel do cérebro humano e sua função será manter o corpo vivo por apenas 50 minutos! O programa simula as diversas mudanças fisiológicas (temperatura do corpo, perda de água etc.) que ocorrem quando o corpo exerce alguma atividade. A primeira coisa a fazer é entrar com a idade e o sexo da pessoa e a atividade que você quer desempenhar. Dormir torna-se a atividade mais fácil de simular, pois o corpo despende pouca energia. Outras atividades consistem em andar, escalar um rochedo e — a mais exaustiva de todas — correr.

Os parâmetros que se controlam são os seguintes: o volume e a taxa da respiração e a taxa de transpiração. Você escolhe os valores iniciais — por exemplo, volume de respiração: 2,5 l; taxa de respiração: quinze inalações por minuto; taxa de transpiração: 3 g por minuto. E aí a simulação começa.

Na tela aparecem cinco gráficos representando as diferentes funções do corpo e um relógio. À medida que o tempo passa, os gráficos mostram como o corpo desempenha a atividade escolhida, e você deve evitar que qualquer um deles ultrapasse o nível de perigo. Se, por exemplo, a temperatura do corpo ficar muito alta, você suspende a atividade por um instante e aumenta a taxa de transpiração para combater a febre. Se você não tiver êxito e um dos gráficos cruzar a linha de perigo, pode receber o lacônico e definitivo diagnóstico: "A PESSOA ESTÁ MORTA". A Heinemann Educational Software produziu esse programa para o micro inglês BBC Model B.

Ventos

Simulação de vôo



Simulação fisiológica

Revisão eletrônica

Programas verificadores de ortografia, gramática e estilo já podem ser usados com muitos processadores de palavras.

Os engenheiros de computadores ainda estão longe de criar máquinas com capacidade de gerar e manipular linguagens naturais, como o português. Mas uma das aplicações planejadas para os computadores é a tradução por máquina de idiomas — do português para o japonês, por exemplo. Existem funções para tradução por máquinas, mas apenas de textos simples, como atas e relatórios. Além de sofrer essa limitação, qualquer tradução produzida pelo computador principal precisa ser corrigida a mão. As histórias de erros são muitas. Dizem que a frase evangélica "The spirit is willing, but the flesh is weak" ("O espírito está pronto, mas a carne é fraca") foi traduzida do inglês para o russo e deste para o inglês, por dois programas diferentes, com o seguinte resultado final: "The wine is agreeable, but the meat is spoiled" ("O vinho está agradável, mas a carne, estragada"). Histórias desse tipo ilustram um ponto muito importante — as dificuldades encontradas quando um computador processa informação sem "entender" o que ela significa. Um problema proposto com freqüência a estudantes de computação é determinar como um computador poderia distinguir os significados de frases como estas:

A AMA GOSTA DE MOEDA CUNHADA.
MINHA CUNHADA AMA SEU MARIDO

A construção das duas frases parece idêntica, mas, no primeiro exemplo, AMA é sujeito, enquanto, no segundo é predicado. CUNHADA também tem um sentido em cada frase. A experiência constitui o único meio de diferenciá-lo. Havendo memória suficiente, pode-se simular experiência no computador, mas isso já foge para o campo da inteligência artificial, área em que a pesquisa ainda não avançou até esse ponto. O que se evidencia é a diferença entre sintaxe e semântica. A primeira, que é o conjunto de regras referentes aos processos de construção usados numa língua, constitui área de fácil domínio para os computadores. Um programador que já tenha encontrado mensagens do tipo SYNTAX ERROR? sabe disso. A semântica, no entanto, diz respeito ao significado que aquelas frases e construções transmitem.

Na década de 50, o lingüista americano Noam Chomsky desenvolveu a base da teoria contemporânea sobre linguagens humanas e regras gramaticais. Embora ele não estivesse diretamente envolvido com computação, suas teorias são pertinentes tanto à tradução por máquina quanto ao trabalho de intérpretes e compiladores de linguagens de programação.

Um dos resultados de sua pesquisa foi a criação de vários instrumentos de software utilizados para escrita de textos. Além de pacotes para processamento de palavras utilizados na criação, edição e impressão de textos, existem programas que verificam textos à procura de possíveis erros de ortografia e datilografia, e até para verificar a correção gramatical e o estilo. Embora nenhum dos produtos atuais apresente qualquer coisa de notável no que diz respeito à inteligência artificial, convém examinar seu modo de operar, a forma em que se apresentam ao usuário e sua programação interna.

Todos os programas verificadores da correção das palavras usam um dicionário ortográfico gravado em disco que armazena até 50.000 palavras. Muitos pacotes permitem o acréscimo ao dicionário de outros itens, como termos de jargão profissional ou nomes de empresas e produtos que você possa querer que sejam verificados.

No entanto, encontrar espaço adequado na memória para um dicionário completo constitui sério problema. Um byte, que é formado por 8 bits, contém um único caractere alfanumérico, quando se usa o código ASCII. Assim, mesmo considerando uma média otimista de apenas cinco caracteres por palavra, um dicionário de 30.000 palavras exige 150 Kbytes de memória. Isso supera a capacidade normal das unidades de disco para microcomputadores. Recorre-se então a duas técnicas para comprimir essa espécie de informação.

Suponhamos que nosso dicionário precise conter apenas letras minúsculas (uma rotina no programa fará as conversões), e que, não sendo necessários símbolos numéricos e alguns de pontuação, eles sejam removidos pelo programa. Assim, podemos construir todo o dicionário usando um máximo de 32 caracteres diferentes, em lugar dos 128 do conjunto ASCII completo (ou 256, se incluirmos símbolos gráficos). Portanto, pode-se reduzir o limite de armazenamento de cada caractere de 8 para 5 bits. A palavra "computar", por exemplo, pode ficar armazenada num total de 40 bits (ou 5 bytes). Os primeiros 5 bits do primeiro byte especificam a letra "c", e os 3 próximos bits mais os 2 primeiros do segundo byte especificam a letra "o", e assim por diante.

Outra técnica empregada nos verificadores de ortografia é a "tokenising" (simbolização). Ela funciona sob a premissa de que certas combinações de caracteres aparecem com tanta freqüência que podem ser representadas por um único byte, que seria "sinalizado" de maneira a indicar que se trata do símbolo de um grupo de caracteres, e não de um único caractere. Seu microcomputador, com certeza, usa simbolização em BASIC — cada palavra-chave, como PRINT ou NEXT, fica armazenada na RAM como um único byte para poupar espaço.

No dicionário de um verificador de ortografia,

usa-se a simbolização na frente das palavras. Consideremos, por exemplo, o grande número de termos (em inglês) que começam com auto-, non-, dis-, ou con-. O pacote VizaSpell, próprio para o processador de palavras VizaWrite do Commodore 64, emprega compressão e simbolização para condensar um dicionário de 30.000 palavras em apenas 65 Kbytes armazenados num disco.

Procurar todas as palavras de um documento em seu dicionário constitui o trabalho mais difícil de um verificador de ortografia. Uma busca binária pode ser empregada mas, no caso de documento com 1.000 palavras, isso levaria horas. O ideal seria que o processador de palavras conferisse palavra por palavra à medida que elas fossem digitadas. Como isso é impraticável em programação, o documento deve ser verificado como um todo, seja em disco ou em RAM (quando se utilizam máquinas maiores). O programa processa o documento e compila, em ordem alfabética, uma lista das palavras. Em geral, mais de 50% de um relatório é constituído por apenas cem palavras diferentes.

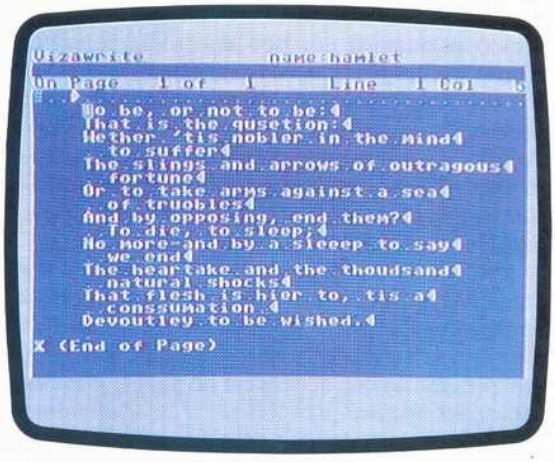
A maioria dos programas verificadores de ortografia recorre a esse processo para fornecer um relatório adicional sobre o emprego de palavras no documento, o que pode ajudar o usuário a evitar repeti-

ções desnecessárias. Assim, um simples algoritmo passa por essa lista e pela lista do dicionário, ao mesmo tempo, para que se formem os pares. O tempo utilizado para completar a busca será bastante reduzido e constante — exatamente 4 minutos no caso do VizaSpell, não importando o tamanho do documento.

As palavras não encontradas no dicionário serão impressas em forma de lista ou destacadas no documento original. Para cada palavra salientada, o usuário recebe três opções:

1) A palavra foi escrita ou digitada de forma incorreta e deve ser corrigida.
2) A palavra está correta e deve ser acrescentada ao dicionário do programa.
3) A palavra está correta, mas é improvável que seja usada novamente; não deve, portanto, ser acrescentada ao dicionário. (Isso pode ocorrer sobretudo quando se está trabalhando com endereços.)

Verificadores de gramática e estilo trabalham de maneira semelhante. Os primeiros recorrem a um número limitado de regras (por exemplo, procurar uma letra maiúscula no começo de cada sentença); em conseqüência, muitos erros gramaticais passam despercebidos. Quase todos os pacotes disponíveis de verificadores de estilo empregam um grande dicionário de exemplos a fim de identificar erros de sintaxe e expressão. Têm elementos, portanto, para sugerir construções alternativas às frases desajeitadas que encontram. Eles também costumam chamar a atenção para o uso excessivo de determinada ex-



Ser ou não ser
Esse famoso monólogo de Hamlet ilustra o funcionamento da revisão eletrônica. Primeiro, digita-se o texto no computador, usando um processador de palavras. Em seguida, o verificador de ortografia é solicitado por um par de comandos simples, o que cria uma lista em ordem alfabética de todas as palavras usadas, indicando também sua freqüência. A lista é conferida com o dicionário do disco e as palavras desconhecidas ou incorretas são evidenciadas. O programa também destaca palavras aparentemente comuns, mas que estão sendo usadas pela primeira vez e que convém acrescentar ao dicionário para uso futuro.

pressão dentro de um parágrafo, ou o uso de frases longas e deselegantes.

Escrever em linguagem BASIC uma forma simples de programa verificador de ortografia, gramática ou estilo constitui exercício muito interessante, mesmo para programadores inexperientes — embora você precise de um bom conhecimento das funções que manipulam as variáveis alfanuméricas em sua máquina. Com uma sofisticação maior do software, podemos esperar que, em breve, os pacotes para processar palavras já venham com essas funções.

Gerador de aplicações

Semelhante ao gerador automático de programas, ele tem aplicações lúdicas, além das comerciais.

Pinball
Este pacote é um tipo de gerador de aplicações para jogos. O usuário desenha o layout e estabelece a lógica para o jogo usando um menu de objetos e diversos instrumentos graficamente representados.

Em artigo anterior do MICROCOMPUTADOR — CURSO BÁSICO estudamos um tipo de programa que, de acordo com especificações dadas pelo usuário, produz um programa capaz de executar as aplicações planejadas. Usam-se geradores de programas com a maioria dos microcomputadores comerciais; alguns pacotes são adequados para microcomputadores pessoais, embora o tipo de aplicação a que se destinam exija, no mínimo, uma unidade de disco.

A forma mais comum de gerar programas que desempenhem funções específicas implica o recurso a pacotes chamados '' geradores de aplicações''. Ao contrário dos geradores de programas, eles produzem programas que dependem do pacote gerador de aplicações original para funcionar. Consideremos a criação de um programa para faturamento usando esses dois tipos de gerador. Assim poderemos destacar as diferenças entre eles.

Se usarmos um gerador de programa, em primeiro lugar o software é carregado do disco para o computador. Quando o usuário tiver respondido a todas as perguntas referentes a arquivos, registros, campos, relações matemáticas, layouts da tela e relatórios impressos que deverão ser feitos (isto é, tiver especificado o programa aplicativo necessário), o gerador pede que se insira um disco virgem na unidade de disco. Neste segundo disco, ele grava o programa que acabou de ser gerado. Repetindo-se o processo, fazem-se cópias do programa de faturamento para todas as filiais da empresa.

Por contraste, um gerador de aplicações parece, de início, menos satisfatório. Quando você tiver completado o estágio de especificações, as rotinas necessárias e o gerador serão gravados no mesmo disco. Como alternativa, o gerador de aplicações pode gravar o programa em disco separado, mas de tal forma que o disco gerador original continuará necessário para se executar a aplicação. Embora se possa empregar uma única cópia do pacote original para produzir ilimitadas aplicações diferentes, todas elas devem ser usadas no mesmo espaço físico do disco gerador. Se você colocar sua aplicação à venda, seus clientes precisarão comprar também uma cópia do gerador. (Esses geradores empregam diversos métodos de proteção de programas, a fim de dificultar a execução de cópias sem autorização.)

Um gerador de aplicações é só um programa de uso geral. Quando você especifica sua aplicação, apenas atribui valores a diversas variáveis importantes do gerador, chamadas ''parâmetros''. Estes controlam o fluxo do programa, a estrutura da informação e os layouts para tela e impressora. Quando você grava a aplicação num disco, está armazenando uma lista de parâmetros. Essa lista — também chamada de ''módulo de aplicação'' — equivale, portanto, a um conjunto de instruções que indicam ao gerador de aplicações como executar determinada aplicação.

Alguns pacotes passam para outra etapa e permitem que você especifique sua aplicação na forma de uma linguagem de nível muito alto (semelhante à pseudolinguagem que usamos de início para desenvolver uma nova rotina no curso de programação em BASIC). O gerador interpreta essa listagem, operação que pode ser repetida pelo interpretador BASIC, caso o programa gerador esteja escrito nessa linguagem. (Isso cria um caso interessante de hierarquia em software.)

Eventualmente, empresas que não respondem pela autoria dos geradores originais criam e comercializam os módulos de aplicações. Por exemplo: considera-se o dBase II (o mais popular dos sofisticados pacotes de bancos de dados usados com microcomputadores) um gerador de aplicações com módulos que consistem em conjuntos de comandos de bancos de dados de alto nível. Módulos para aplicações mais particulares — tais como um sistema de contabilidade destinado a corretores da bolsa — podem ser construídos sem que se escreva o programa desde o início. Tendo em vista a limitação do mercado, um pacote para corretores da bolsa executado sob o comando do dBase II mostra-se mais funcional

que um escrito em BASIC. No primeiro caso, o autor do programa concentrou todos os seus esforços na *operação* do programa e não no ato de escrever o código. As partes do programa mais suscetíveis a violações (por exemplo, o manuseio do arquivo) são escritas pelos autores do gerador e testadas em diferentes aplicações por milhares de usuários.

A principal diferença, porém, entre um gerador de programas e um gerador de aplicações reside na capacidade de agradar ao usuário. O programa final criado pelo primeiro tipo de pacote consiste num código artificial, quase sempre escrito numa linguagem como o BASIC. Esse código ainda é inferior, tanto em eficiência quanto em estilo, àqueles criados pelo homem. Com o gerador de aplicações, até 99% do programa final consiste em código escrito pela software house, provavelmente em linguagem de máquina. Esse é o caso do Silicon Office, um dos geradores de aplicações mais sofisticados e fáceis de se usar em microcomputadores comerciais. O programa resultante mostra-se rápido e eficiente, tem procedimentos incorporados para detectar erros do operador e produz displays de tela nítidos, com menu.

Além disso, os geradores de aplicações não se restringem a programas comerciais. No Pinball Construction Set (ver p. 241), um ótimo exemplo de pacote não-comercial, o módulo de aplicações vem bem especificado, dispondo na tela os elementos necessários para a mesa desse jogo eletrônico.

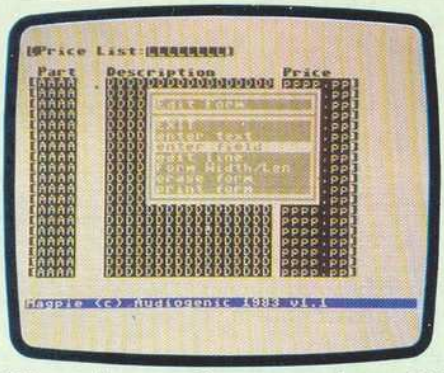
Existe muita coincidência entre as programações orientadas para o assunto e para o objetivo (ver p. 242), que se resume em incentivar o programador a executar suas aplicações pela simples especificação dos objetivos exigidos do programa. Até um simples programa para folha eletrônica, próprio para microcomputadores pessoais, é considerado um gerador de aplicações — se você especificar as relações entre os diversos campos, o pacote faz todo o trabalho de rotina.

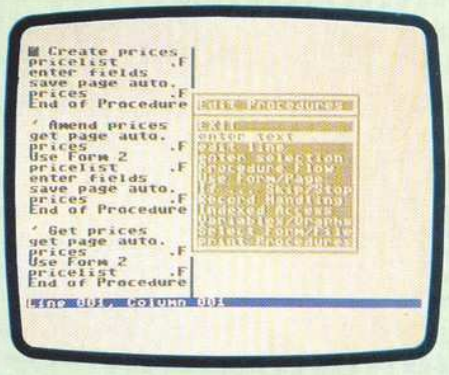
O Magpie — produzido pela Audiogenic para o Commodore 64 com uma unidade de disco — é um gerador de aplicações montado para uso comercial. Também esse pacote emprega bem a programação orientada para objetivos visuais: especifica as relações dos campos de informação entre diferentes registros, quando se desenha o layout dos arquivos.

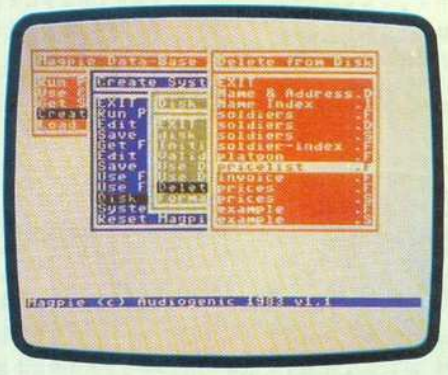
Jogos complexos
Os geradores de programas e os geradores de aplicações vêm propiciando o desenvolvimento de atividades lúdicas (como os fliperamas) cada vez mais complexas.

Embora, a rigor, não sejam considerados geradores de aplicações, um número crescente de pacotes vêm incorporando esses princípios. Quando executados pela primeira vez, os programas à base de parâmetros fazem uma série de perguntas ao usuário e gravam as respostas em disco ao lado dos programas. Essas informações determinam alguns detalhes da operação do programa. Um programa de faturamento, por exemplo, faz perguntas relacionadas com informações que a empresa quer que constem das faturas e sobre os prazos de pagamentos por ela determinados. Um jogo de fliperama pode perguntar com quantos alienígenas e foguetes o usuário gostaria de começar, ou até dar-lhe a oportunidade de desenhar os invasores.

O software vem sendo projetado de forma que o usuário não tenha de aprender programação e, ao mesmo tempo, proporciona alto grau de flexibilidade na operação. Numa situação ideal, o software se ajusta às exigências do usuário (em vez de o usuário ajustar-se ao software).







Magpie no Commodore 64

A primeira etapa na criação de uma aplicação consiste em especificar o layout de todos os formulários, transações e relações, tal como uma lista de preços.

Em seguida, ele especifica todos os cálculos e processos, na forma de uma lista de instruções. Aqui são representadas as rotinas para alterar os preços e trazê-los (GET) do disco para a tela.

O Magpie funciona com menu. Aqui a tela mostra o resultado após selecionar CREATE, depois DISK, DELETE e o arquivo a ser removido, que no caso é a PRICE LIST (lista de preços).

Elementos subversivos

O planejamento cuidadoso e a abordagem passo a passo reduzem o tempo gasto na eliminação de erros do programa.

Conforme adquire maior habilidade no desenvolvimento de programas, você tende a ficar mais apto a desembaraçá-los de erros ("debugá-los"). Os enganos lógicos e de sintaxe, que mesmo os programadores mais experientes cometem, tornam-se menos freqüentes ou problemáticos à medida que sua experiência aumenta. Os recursos aqui expostos podem ajudá-lo a evitar erros e tornar-se mais eficiente na eliminação de eventuais incorreções.

O programa começa na cabeça do usuário. Logo, se sua estrutura for pensada de modo superficial, o desenvolvimento se fará com grande quantidade de erros. De início, portanto, convém colocar o problema de modo tão claro quanto possível para si mesmo ou para outra pessoa. A seguir, o problema deve ser dividido em unidades logicamente completas — input, output, algoritmos, estruturas de dados, procedimentos etc. Cada parte deve ser examinada como um aspecto individual. Se necessário, subdivida cada um desses aspectos nas unidades que o compõem e torne a dividi-los até que o problema original se transforme num conjunto estruturado de subproblemas programáveis com facilidade. Uma abordagem formal, como o emprego de pseudolinguagem ou diagrama de fluxo, mostra-se essencial no estágio de estruturação como forma de preservar a estrutura global do programa e mantê-lo sob controle. Você deve ficar longe do teclado até ter certeza de que sabe como programar cada unidade do problema. Essa abordagem é denominada top-down, um método que pode condensar bem o tempo necessário para eliminação de erros.

A reinvenção da roda

A subdivisão de problemas em tarefas de resolução mais fácil leva você a desenvolver programas que constituem conjuntos de sub-rotinas ou procedimentos vinculados por um programa básico principal. Isso torna mais fácil a detecção de erros e possibilita a construção de um arquivo de sub-rotinas de eliminação de erros para emprego em programas futuros. A alternativa é denominada "reinvenção da roda": cada vez que desenvolve, por exemplo, um programa que classifica dados, você resolve de novo o problema de como elaborar uma rotina de classificação. E é bem provável que volte a incorrer em erros cometidos antes. Mas é simples desenvolver o programa, isento de erros, gravá-lo e chamá-lo, sempre que for necessário.

Tente utilizar nomes apropriados de variáveis, mesmo quando tiver de empregá-los sob forma abreviada. LÍQUIDO=BRUTO-IMPOSTOS, por exemplo, indica com clareza seu significado; e LQ=BR-IMP não é mau substituto; porém, L=B-I revela-se uma expressão muito ambígua e não indica as variáveis envolvidas. Constitui bom procedimento manter uma tabela que mostre todas as variáveis utilizadas no programa, bem como sua finalidade. Isso o levará a padronizar seu emprego (como o uso freqüente de certas variáveis com uma única letra para os contadores de loop) e evita a utilização da mesma variável para diferentes finalidades. De

Controle de erros

```
100 GOTO 200:X$="E SO, PESSOAL!"
120 I=12:K=1984
140 FOR K=1 TO LT
160 PRINT"QUEM NECESSITA DA ESTRUTURA?";N$:NEXT
180 RESTORE
190 FOR L= 1TO I
200 INPUT"FORNECA SEU NOME" ;N$
220  INPUT "FORNECA SUA IDADE" ; LT
240 GOSUB 100
260 PRINT SE VOCE TEM" ;LT; "AGORA"
280 PRINT"VOCE NASCEU EM" ;K-LT
300 ANO$=K-LT
320 LY=INT(ANO)/4*4
340 IF LY=ANO THEN IF INT(LY/100)*100=LY THEN GOTO 370
380 PRINT "ANO FOI UM ANO BISSEXTO" :GOTO 420
390 PRINT "ANO NAO FOI UM ANO BISSEXTO"
400 NEXT
420 PRINT X$
440 STEP
```

- Esta instrução não será executada, pois o comando GOTO se desvia dela
- Estas duas linhas estão na ordem errada. A linha 100 deveria ser: GOTO 190
- K deve representar uma constante, mas esta instrução a elimina
- Devido à falta das aspas aqui, o procedimento NEXT não será executado
- Deveria ser: RETURN
- Erro de sintaxe: o sinal de dois pontos ":" deveria ser ponto e vírgula ";"
- Esta instrução causará grandes dificuldades. Deveria ser: GOSUB 140
- Faltam os sinais de aspas
- Resultará em algum número sem significado, pois K teve seu valor alterado depois da linha 120
- O parêntese de encerramento está mal colocado, resultando em erro de cálculo. Deveria ser: INT (ANO/4)*4
- Erro de sintaxe: deveria ser ANO=K-LT
- Não existe a linha 370!
- Deveria ser ;" GOTO 420 corresponde a um desvio do loop FOR-NEXT
- Deveria ser o nome da variável do loop, isto é, NEXT L
- Cuidado, X$ não foi inicializada!
- Erro de sintaxe: deveria ser um comando STOP

modo semelhante, convém armazenar os valores freqüentes nas variáveis no começo do programa, remetendo-se depois a elas. Isso torna o programa mais rápido e organizado, e você pode alterar os valores sem ter de procurar por todo o programa.

Mesmo com essa abordagem formal, é difícil eliminar por completo os erros. Assim, convém adotar um método organizado de encontrá-los e suprimi-los.

O tipo mais comum de erros ocorre com a sintaxe. Em geral, você pode corrigi-los tão logo os encontre. Observe:

```
10 PRINT"GRANDES ERROS POSSUEM
   PEQUENOS"
20 PRINT"ERROS PARA ATRAPALHA-LOS"
```

Essas linhas quase sempre ocasionam mensagens de erro, caso não sejam digitadas como duas linhas separadas. A linha 10 tem quarenta caracteres; desse modo, quando você a digita numa tela de 40 colunas, o cursor pára no início da linha seguinte, o que pode fazer você esquecer de teclar RETURN na linha 10, antes de iniciar a digitação da linha 20. Caso isso aconteça, o que parece ser um conjunto de linhas perfeitas em seu programa será, de fato, uma linha com um erro de sintaxe (o número 20) no meio dela. Um modo de detectar esses erros consiste em listar linhas suspeitas uma por vez, e não como partes de um programa.

As mensagens de erro podem se mostrar enganadoras. Observe este exemplo:

```
25 DATA 10.2,34,56,9,0.O08,15.6
30 FOR K=1 TO 5: READ N(K):NEXT K
```

Esse programa pode não realizar sua tarefa em razão de suposto erro de sintaxe na linha 30, quando efetivamente o erro está nos dados da linha 25 (um dos caracteres zero foi erroneamente digitado como a letra O).

Instruções com erros que não prejudicam a sintaxe são a falha mais comum e, em geral, a mais difícil de se detectar. Nesse caso, o método torna-se fundamental. Comece pela tentativa de encontrar de modo aproximativo o erro do programa. Isso é relativamente simples com programas modulares bem estruturados e torna-se ainda mais fácil pelo emprego do recurso TRACE, que ocasiona a impressão da linha atual do programa na tela, à medida que é executado. Se sua máquina não possibilita esse procedimento, crie instruções TRACE a intervalos regulares por todo o programa (PRINT "LINHA 150", no início da linha 150, por exemplo). Empregue também o comando STOP para interromper a execução do programa em seus pontos significativos, de modo que você possa examinar os valores das variáveis mais importantes. Realize isso de modo direto pelo emprego de PRINT, ou desenvolva uma sub-rotina no final do programa:

```
11000 REM IMPRIMIR AS VARIAVEIS
11100 PRINT"RESULTADO,TAMANHO,FLAGS"
11200 PRINT RS;TM;F1;F2
11300 PRINT"MATRIZ DO PAINEL"
11400 FOR K= TO 10:PRINT PN$(K):NEXT K
```

Por conseqüência, quando o programa encontra um comando STOP, você pode digitar GOTO 11000 e apresentar o estado atual das variáveis. Convém até mesmo alterá-las (pela digitação, digamos, de TM=17 e o pressionamento de RETURN) e a seguir reiniciar o programa com o comando CONTinuar.

Erro de principiante

Para o iniciante em programação, os erros parecem reagir instintivamente, como animais, ocultando-se. O curioso é que um erro histórico de fato resultou da ação de um inseto — "bug", em inglês que também é uma buzzword para designar "erro". Ao tentar eliminar o erro do programa que estava desenvolvendo no equipamento Harvard Mark II, em 1945, Grace Hopper descobriu que uma mariposa havia se enroscado na aparelhagem eletromecânica do computador e estava ocasionando o problema.

Quando tiver descoberto que o erro está num conjunto de linhas ou em determinada variável, você estará próximo de eliminá-lo. Aja, porém, com cuidado. Tente uma solução por vez, para verificar seu exato efeito na execução do programa. É muito fácil realizar várias alterações no intervalo dos processamentos, eliminando um erro mas criando outros, e depois esquecer o que você fez.

Os loops e desvios, sobretudo quando internos, são solos férteis para erros e exigem atenção especial tanto no escrever quanto no eliminar incorreções. Observe este trecho de programa:

```
460 IF SM< 0 AND SC < > -1 THEN IF SC>0 OR
    SM=SC-F9 THEN LT=500
470 FOR C1=1 TO LT:FOR C2=LT TO C1 STEP-1
480 SC=SM+SC*C2
490 NEXT C2:SM=0:NEXT C1
```

O que significa todo esse procedimento? Mesmo que você saiba o que ele deve realizar, pode avaliar sua eficácia? A inserção de instruções no interior de loops, quando deveriam estar fora deles, implica quase sempre criação de erros. O mesmo ocorre quando não se cobrem todas as condições possíveis, desenvolvendo instruções do tipo IF-THEN. Um caso especial desse tipo de erro acontece quando você escreve instruções múltiplas, após IF-THEN. Por exemplo:

```
655 IF A$= " " THEN GOTO 980:A$=B$
660 PRINT A$
```

A instrução A$=B$ não será executada porque, ou A$=" ", em que o controle passa para a linha 980, ou A$<>" ", em que o restante da linha 655 é ignorado.

Na eliminação de erros, a experiência é a melhor mestra, mas um procedimento passo a passo e um método disciplinado mostram-se auxiliares inestimáveis. Vá com calma e — acima de tudo — *não entre em pânico*.

Kits de ferramentas

Esses pacotes de software expandem um dialeto limitado de BASIC e facilitam a tarefa do programador.

Os primeiros micros domésticos, como o Apple II e o Commodore PET, tinham capacidade limitada — foram projetados especialmente para manipular números e textos. O BASIC desenvolvido para eles apenas supria rotinas e comandos necessários a esses fins. Em conseqüência, escreveram-se muitos programas "utilitários" ou "kits de ferramentas", a maior parte em linguagem de máquina, que operavam fora da área de programação BASIC. Em forma de comandos diretos adicionais, ajudavam na construção e na correção de programas.

Desde essa época, os engenheiros têm criado uma infinidade de funções gráficas e sonoras, como resultado da explosão do interesse por jogos do tipo fliperama em microcomputadores. Cada modelo novo apresenta mais funções extensivas, que são logo embutidas no software escrito profissionalmente.

Instrumentos de trabalho
Estes são alguns dos pacotes e ferramentas de extensão do BASIC para os microcomputadores mais populares no exterior. Os pacotes para criação de facilidades para sprite em computadores que não as possuem normalmente estão crescendo em popularidade. No Brasil existem vários recursos desse tipo.

Ferramentas	
SUPER TOOL KIT	Disponível para o Spectrum Sinclair 16 K e 48 K, da Nectarine
SPECTRUM EXTENDED BASIC	Para o Spectrum Sinclair 48 K, da CP Software
SPECTRUM KEYDEFINE	Para o Spectrum Sinclair 48 K, da Scientific Software
PROGRAMMER'S AID	Para o Commodore Vic-20, da Commodore
BUTI	Para o Commodore Vic-20, da Audiogenic
TOOL BOX	Para os micros ingleses modelo BBC, da BBC Software
SPRITE MAGIC	Disponível para o micro inglês Dragon 32, da Merlin Micro Systems
SPRITE GRAPHICS	Para o Spectrum Sinclair 48 K, da B Sides Software
SPRITE MASTER	Para o micro inglês BBC Model B, da Micro Dealer UK

Contudo, com uma ou duas exceções, o BASIC incorporado tem pouco ou nenhum aperfeiçoamento sobre as versões anteriores. Isso resulta na obrigatoriedade de o usuário criar as rotinas, empregando repetidas vezes os comandos PEEK e POKE para incorporar essas novas características no conjunto de comandos disponíveis. Assim, existem alguns pacotes, kits de instrumentos e extensões para BASIC adaptáveis à maioria das máquinas mais populares de uso doméstico. Em geral, permitem fácil acesso aos recursos existentes (como editores de sprites ou de sons), ampliam as facilidades de software (por exemplo, criadores de sprite), ou funcionam como simples auxiliares na programação BASIC.

Extensões desse tipo podem se localizar em RAM, ROM interna ou cartucho de ROM. É preferível a extensão em ROM a uma que esteja carregada em RAM, já que ela não emprega a memória do usuário e não corre o risco de ser apagada por engano. Em geral, o programa escrito com ajuda de uma ferramenta dessas só pode ser executado em computador que possua equipamento similar. Contudo, existem pacotes geradores de programas independentes que são executados em versões não expandidas do computador. Essa é a base da maioria dos editores de gráficos e de sprites, como também de editores de sons.

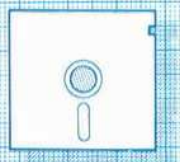
Algumas características úteis para se procurar nas extensões de BASIC são os comandos gráficos especiais (como PAINT, DRAW, PLOT, CIRCLE etc.) e comandos de som (SOUND, PLAY, MUSIC, ENVELOPE etc. ou palavras que descrevem um efeito sonoro, como BANG ou ZAP). Outros recursos úteis são os comandos de programação estruturada, como REPEAT-UNTIL e IF-THEN-ELSE. Palavras como essas capacitam o usuário a escrever programas que seguem uma seqüência lógica e evitam as linguagens confusas e difíceis de entender, resultantes do uso indiscriminado do GOTO.

Simon's BASIC

A extensão mais completa da linguagem BASIC é o Simon's BASIC, próprio para o Commodore 64 na forma de cartucho ROM. O BASIC padrão do Commodore, que faz parte do 64, mostra-se um tanto antiquado, pois oferece um mínimo de comandos dedicados e nenhum de programação estruturada. Apesar de ter um hardware avançado, como sintetizador de som completo, alta resolução gráfica e gráficos sprites, o controle do BASIC sobre essas funções faz-se pelos comandos PEEK e POKE. O Simon's BASIC oferece uma extensão considerável ao BASIC do Commodore, mediante estes recursos extras:

Instrumentos da máquina

Estes comandos representam os recursos geralmente incluídos nas boas extensões do BASIC



Diversos comandos para a utilização de discos são oferecidos, para formatá-los, copiar e anular unidades de arquivo e fazer cópias completas de reserva de um disco inteiro

DISK.

Além dos comandos para desenhar linhas retas em alta resolução gráfica, também pode haver recursos para desenho de ponto a ponto, uma forma mais fácil de construir imagens

DRAW

O comando PAINT proporciona um meio de preencher uma área definida na tela com qualquer cor. Em certas máquinas, o cursor fica posicionado no meio da figura, e o computador preenche a área, pintando do centro para fora, até encontrar uma linha sólida

PAINT

Alguns comandos gráficos do tipo CIRCLE chegam a desenhar círculos parciais (arcos) e elipses

CIRCLE

Embora as funções sonoras variem muito de uma máquina para outra, um comando MUSIC em geral oferece o recurso de tocar uma sequência de notas predefinidas numa cadeia de dígitos

MUSIC

O diretório de um disco mantém uma lista atualizada dos nomes, tipos e tamanhos de todos os arquivos. O comando DIR proporciona a visão do diretório na tela, sem perder o programa em RAM

DIR

Em muitos computadores, o acesso aos joysticks é tarefa difícil, envolvendo os comandos PEEK e POKE. Os comandos de JOYstick colocam a posição desses instrumentos diretamente nas variáveis BASIC

JOY

1) Conjunto completo de dispositivos auxiliares de programação, incluindo funções que permitem controle extra sobre listagem de programas, correção e dispositivos auxiliares de segurança (proteção contra cópias clandestinas de seus programas).

2) Controle adicional de variáveis alfanuméricas e comandos de manipulação de texto.

3) Operadores extras de matemática e comandos para conversões numéricas.

4) Comandos simplificados para manipulação de discos.

5) Comandos de alta resolução gráfica que permitem misturar o texto com traçado de gráficos por meio de pontos e desenhar formas. Também está incluído um recurso para colorir dentro de contornos.

6) Comandos de baixa resolução gráfica e de controle de tela que podem duplicar áreas gráficas determinadas e manipular com facilidade a área da tela. Eles também permitem que o conteúdo de qualquer tela seja gravado em disco, fita ou impressora.

7) Criador e editor de sprites fáceis de usar.

8) Programação estruturada usando comandos de procedimentos — tais como PROC, CALL e EXEC; além disso, rotinas para testar loops e condições, como REPEAT-UNTIL, LOOP-EXIT, IF-END LOOP e IF-THEN-ELSE — que em geral elimina a necessidade de usar GOTOs e GOSUBs.

9) Rotinas para criação de som que permitem o aproveitamento máximo da capacidade sonora do Commodore 64, usando comandos simples de formação e execução do som.

10) Comandos simples para caneta óptica, joystick e paddle.

Poucas extensões incluem uma linha tão completa de rotinas adicionais. A maioria dos pacotes fornece recursos e comandos para uma área específica de programação. Por exemplo, o cartucho Super Expander da Commodore, para o Vic-20, fornece apenas uma linha simples de comandos para alta resolução gráfica e música. As extensões mais populares incluem dispositivos que auxiliam na construção de programas. Em geral, eles apresentam comandos de entrada por uma única tecla e diversas rotinas automáticas que simplificam a numeração de linhas, edição e correção no modo direto.

Um prazer criativo

É comum utilitários e extensões permitirem que funções incorporadas num computador sejam alcançadas com facilidade. As rotinas que aumentam de modo significativo a capacidade de um microcomputador são difíceis de se achar, mas alguns pacotes engenhosos já estão à venda. Por exemplo, as muitas vantagens dos gráficos sprites para ação rápida em jogos de fliperama inspiraram algumas empresas a escrever pacotes criadores de sprites para computadores que não possuem essa função.

As facilidades para BASIC, as ferramentas e as extensões que destacamos representam pequena fração dos melhoramentos e dispositivos auxiliares disponíveis. Embora a atual tendência dos fabricantes seja apresentar versões completas e avançadas do BASIC, haverá sempre a necessidade de software auxiliar, para fazer da programação um prazer criativo, e não uma tarefa árdua.

Descubra o código

A elaboração de mensagens cifradas foi uma das primeiras aplicações dos computadores. Hoje, a criação e a decifração de códigos simples fazem parte das habilidades do programador BASIC.

Todas as nossas comunicações com os outros são codificadas. Tanto a fala como a linguagem escrita são apresentadas de forma inteligível, se a pessoa receptora da mensagem for capaz de interpretar o código. Isso também vale em nossa "conversa" com os computadores. A maioria dos micros utiliza um dialeto de BASIC acessível ao usuário comum. Mas a máquina em si não recorre a essa linguagem para executar suas funções: deve primeiro converter as instruções BASIC na forma puramente numérica que pode então usar para estabelecer as seqüências de comutações definidas no programa. Assim, produz os resultados desejados. Códigos desse tipo — linguagens humanas e de programação — são bem acessíveis em nossa vida cotidiana. Qualquer pessoa aprende francês, alemão, BASIC ou FORTRAN, desde que empregue esforço e vontade.

Compressão de dados

O usuário de computador que precisa armazenar grande quantidade de arquivos de texto está constantemente buscando métodos de comprimir os dados. Um modo de obter esse resultado é a simbolização. O símbolo pode ser substituído por uma palavra ou chave freqüentemente utilizada, de modo semelhante à maneira como os computadores TK85 ou CP 200 apresentam uma palavra em BASIC ao toque de uma única tecla.

Além disso, as técnicas de codificação são usadas para comprimir mais ainda os dados. Compact, um utilitário do sistema operacional Unix, é considerado capaz de comprimir arquivos de texto a uma média de 38%, e Clip, que processa sob o CP/M, obtém resultados ainda melhores. Compactor, que processa no Commodore 64, realiza a mesma função nos programas em BASIC, pela eliminação de todas as instruções REM, espaços desnecessários etc.

Mas há outro tipo de codificação (mais precisamente denominado criptografia), com objetivo oposto ao da comunicação: negar compreensão a todos, menos ao pequeno grupo ao qual a comunicação se destina. Antes da segunda metade do século XX, a transmissão de dados de forma não inteligível a todos restringia-se a governos e algumas empresas de grande porte. Porém, com o intenso emprego das linhas públicas de telefones — vulneráveis à escuta — para troca de informações com valor comercial, o uso da criptografia tornou-se mais comum.

As cifras e códigos variam do muito simples — a soma ou subtração de determinado valor para cada byte, por exemplo, ou a substituição formatada de um caractere por outro — até os esquemas complexos que estão sendo elaborados de acordo com os mais recentes progressos da teoria dos números. Essas cifras não contêm quaisquer elementos de repetição e, por isso, não são vulneráveis a métodos de decodificação de análise de freqüência.

A mais simples de todas as técnicas de criptografia de importância significativa é a Cifra de César (com certeza utilizada na época do Império Romano). A cifragem do código de César exige que se tenha apenas a mensagem e conhecimento do código; desse modo, não são necessários livros de código volumosos ou documentos a serem ocultos, nem máquinas sofisticadas. Eis uma mensagem simples, cifrada segundo essa técnica:

KGAPMAMKNSRYBMP ASPQM ZYQGAM

Podemos fazer algumas conjeturas sobre essas palavras codificadas a partir do modo pelo qual os grupos cifrados estão espacejados (embora isso pudesse ter sido feito apenas para nos confundir). A primeira coisa que salta aos olhos é o fato de que a mensagem consiste em três palavras: a primeira tem quinze letras; a segunda, cinco; e a terceira, seis. Também podemos admitir que a segunda palavra e a terceira terminam com a mesma letra. A letra final comum aqui (M) é também uma das duas letras na mensagem com ocorrência maior (a outra é A). Essa observação é de grande valor para o decifrador, desde que ele saiba com que língua está trabalhando.

Com uma amostra de tão poucos elementos como a nossa (um total de apenas 26 letras, que qualquer estatístico considerará uma amostragem insuficiente para basear a análise), nossos resultados provavelmente serão falhos. Mesmo assim, tentaremos uma substituição por freqüência para ver se os resultados têm algum significado. Substituímos primeiramente o M pela letra O:

KGAPoAoKNSRYBoP ASPQo ZYQGAo

A mensagem ainda não apresenta nenhum significado, mas há outras pistas. Qual a relação entre a letra original e aquela pela qual nós a substituímos? M está situada duas posições antes do O no alfabeto. O que acontecerá se fizermos a mesma substituição no restante da mensagem? Duas posições após o A está C. Assim, tentemos acrescentar esse dado:

KGcPocoKNSRYBoP cSPQo ZYQGco

Na terceira palavra temos agora um final com *co* que é uma construção válida em português. A segunda palavra começa com *c* e termina com *o*, que também pode nos indicar algo viável; desse modo, talvez estejamos na direção certa. Façamos o mesmo com o restante da mensagem. A letra situada duas posições

depois de K é M; duas após G é I; e assim a primeira palavra poderá ser MICRO...

A Cifra de César é, portanto, um código de substituição que se apóia no ''deslocamento'' do alfabeto, para diante ou para trás, num certo número de posições, com o objetivo de determinar o novo valor de cada caractere. Pode ser refinada depois pelo emprego de uma chave de transformação — 24225, por exemplo. Neste caso, a primeira letra estará deslocada duas posições, a segunda, quatro, a terceira, duas e assim sucessivamente. Quando o final da chave é atingido, retornamos ao início. Com o emprego dessa chave de transformação em nossa mensagem, o exemplo seria:

KEAPJAKKNPRWBMM AQPQJ ZWQGXM

Neste exemplo, a análise de freqüência será inútil, pois não há uniformidade na substituição — uma letra terá substitutas diferentes, de acordo com sua posição na mensagem completa. Outra cifragem independente torna a mensagem assim:

MOPDCOSIRCMUAO US AIOCOTRRBC

Examinando-a atentamente, verificamos que esta série de caracteres corresponde a um anagrama de MICROCOMPUTADOR — CURSO BÁSICO, com os dois espaços entre as palavras. Aqui, simplesmente procuramos determinar o algoritmo de codificação, dando amostras do texto cifrado e do não-cifrado — um procedimento bastante comum. Se a cifra tiver de ser compreensível ao receptor da mensagem, a combinação das letras deve ser de algum modo previsível. Esta cifra, em particular, conhecida como Cerca de Barras, por razões que logo se evidenciarão, também exige o conhecimento da chave pelo decodificador — e, neste caso, ela é 3. Tomemos os sete primeiros caracteres e vamos escrevê-los com três espaços entre si:

M***O***P***D***C***O***S

Você consegue perceber algo? Bem, então tente: escreva o texto não-codificado em três linhas, colocando letras acima e abaixo da linha central, da seguinte forma:

```
M   O   P   D   C   O   S
 I R C M U A O
  - U S - A I O
    C   O   T   R   R   B   C
```

O sinal de subtração representa o espaço entre as palavras e o método de codificação é evidente.

Os exemplos citados até aqui são todos de cifras, entendidas como um método de escrita secreta que emprega a substituição ou a transformação de letras, de acordo com uma chave. Os códigos são diferentes porque tendem a substituir blocos inteiros por outros, em geral menores (possibilitando ainda condensar os dados). Mas apresentam um inconveniente: o receptor e o transmissor devem dispor de um livro de código antes que as mensagens sejam comunicadas. Um exemplo dessa técnica utiliza um romance, jornal ou qualquer outro texto acessível a ambos e indica *as palavras* que constituirão a mensagem apenas dando o número de seqüência em que ocorrem. Um texto como ''João gosta de brincar com seu microcomputador. Sua irmã prefere o curso de inglês. A mãe acha que o básico é que eles estudem.'' poderia ser a chave para o código 7, 5, 8. Talvez você possa deduzir a mensagem...

Computadores de qualquer tipo são de grande valor ao se tentar cifrar ou decifrar mensagens em código. Um requisito primário da Cifra de César, por exemplo, é a capacidade de deslocamento por uma série alfanumérica, acrescentando ou subtraindo uma constante ao valor ASCII de cada caractere a ser impresso. Pode-se alterar essa constante quando o programa for processado. O alfabeto, então, se inverte (onde a chave é 1, A resulta em Z).

Descobrindo a constante, você decodifica esta mensagem:

EXI E TVSBMQE!

A Cifra de César

Este programa (desenvolvido em BASIC) codifica textos pela Cifra de César com o emprego de uma chave múltipla de cinco elementos. A mensagem aparece em texto normal, à medida que é fornecida e, ao ser pressionada a tecla RETURN, imprime-se a versão cifrada. A mensagem deve ser fornecida sem espaços nem pontuação.

```
10 INPUT "FORNECER UMA CHAVE DE CINCO
   ALGARISMOS";K$
20 INPUT "FORNECER A MENSAGEM";M$
30 FOR I=1 TO LEN(M$)
40 LET J=I — INT(I/5)*5+1
50 REM ***RODA O TEXTO POR MEIO DA CHAVE***
60 LET M=ASC(MID$(M$,I,1)) — VAL(MID$(K$,J,1))
70 IF M<65 THEN LET M=M+26
80 PRINT CHR$(M);
90 NEXT I
```

Criptoanálise
Uma das primeiras aplicações dos computadores foi a decifração de códigos muito complexos de substituição de múltiplas chaves utilizados na Segunda Guerra Mundial. Os alemães haviam desenvolvido a máquina chamada ENIGMA, que gerava suas próprias cifras. Os criptogramas resultantes fizeram com que os Aliados se empenhassem em sua interpretação. O êxito foi afinal obtido pelo grupo Colossus, na Inglaterra, do qual Alan Turing foi destacado membro.

Risco calculado

Uma tarde no Jóquei
Prognósticos sobre corridas de cavalos constituem aplicação interessante de microcomputadores, mas especialistas constataram que as chances de se acertar o vencedor de um páreo continuam muito limitadas.

Os computadores têm muitas aplicações no mundo dos jogos de azar. Existem programas de sistemas de apostas até para microcomputadores.

Os jogos de azar giram em torno de probabilidades. Embora a maior parte dos jogadores contumazes prefira dizer que eles estão aí para serem ganhos, essa afirmação mostra-se infundada, pois quase todos os jogadores perdem com freqüência, e no total essas perdas são consideráveis. Isso acontece porque as chances estão contra eles e a favor do Jóquei Clube, do cassino, da Loto, da Loteria Esportiva. Para avaliar se os computadores poderiam ajudar a restabelecer o equilíbrio, precisamos, de início, considerar as chances. (*Chance*, aqui, entendida como probabilidade; por exemplo: a chance de se obter determinada face jogando um dado não viciado é de 1/6.)

Desguarnecidos das armadilhas externas, todos os jogos de azar se resumem em apostar no resultado de um evento fortuito. Em geral, este é gerado por algum dispositivo randômico (aleatório), como uma bola rodando numa roleta ou uma carta puxada de um baralho bem misturado. Se os parâmetros — digamos, o número de cartas — são conhecidos, o cálculo da probabilidade permite certas previsões sobre as chances de ocorrência do evento. A roleta mais usual, com 37 divisões numeradas de 0 a 36, exemplifica o cálculo das probabilidades: a bola pode cair em dezoito casas pares ou dezoito ímpares, mais o zero. A probabilidade de a bola cair numa casa com número ímpar pode ser expressa como 18/37, ou seja, cerca de 48,6%. Isso é um pouco menos que a probabilidade (50%) de uma moeda arremessada para o ar dar "cara"; a diferença, devido à casa do zero, representa a margem de lucro do cassino.

Essa margem faz com que os jogos de acaso sejam pouco gratificantes. Apesar de os fornecedores de sistemas para esses jogos afirmarem o contrário, um computador nada pode fazer para melhorar as chances básicas do apostador. De fato, grandes autoridades no assunto calculam que, a longo prazo, é impossível ganhar em qualquer jogo de cassino, exceto, talvez, em alguns jogos de cartas.

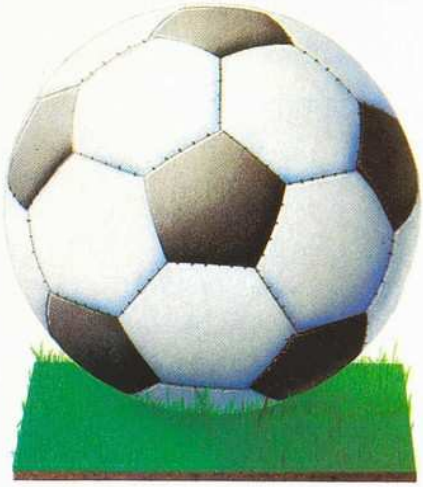
Essas objeções teóricas não desencorajaram inventores entusiasmados, e os proprietários de microcomputadores têm agora à disposição uma variedade de sistemas de jogo alegadamente seguros — e alguns deles parecem funcionar mesmo.

Loteria Esportiva

Os sistemas de jogo dos cassinos quase sempre são variações de "dobrar a aposta", processo que leva a desvantagem de exigir quantia infinita de dinheiro para aposta, a fim de ser bem-sucedido. Há ainda outro problema: não se permite, nos cassinos, a utilização de qualquer tipo de computador, pois essas máquinas são consideradas de grande ajuda quando habilidade, memória ou estratégia estão envolvidas. O computador pode impor a quem joga a disciplina necessária e atuar como recurso de memória. No entanto, o valor dessa assistência tende a ser inversamente proporcional à perícia do jogador.

Por outro lado, o fator habilidade envolvido em quase todos os jogos de azar é mínimo. As apostas em futebol (como o concurso de prognósticos da Loteria Esportiva) são um caso à parte. O programa de previsão de apostas mais sofisticado é o famoso F4 Football Forecast, do professor Frank George, encontrado no mercado internacional em versões adequadas para a maioria dos microcomputadores. Baseado em dez anos de análise estatística, o programa atribui determinado valor ao desempenho médio de cada time. Quando ajustados pela aplicação de pesos relativos ao desempenho a longo prazo — extraídos de tabelas oficiais ou do noticiário esportivo —, ao desempenho a curto prazo e ao resultado da última partida, a comparação desses números possibilita ao programa predizer o resultado mais provável de determinada partida.

Com o Palmeiras jogando em casa contra o Ferroviário Ituano, por exemplo, prevê-se com facilidade o resultado, mas o programa mostra de fato seu valor quando prognostica o escore de uma partida entre times mais equilibrados. Isso não significa que o programa sempre acerte, mas a análise estatística indica que seu uso quase triplica a probabilidade de sucesso. "Reconheço que as chances *contra* ainda são enormes, mas é melhor jogar da forma mais inteligente possível, não é?", pergunta o autor.



Sistemas de apostas
Existem vários pacotes para microcomputadores que alegadamente melhoram as chances de ganhar no sistema de apostas do tipo da Loteria Esportiva brasileira. Grande número de programadores já tentou escrever seu próprio programa com essa finalidade. Os melhores sistemas desenvolvidos fazem uso de vasto banco de dados com informações sobre as partidas anteriores entre as equipes envolvidas e podem ser úteis na previsão dos resultados das partidas.

Corridas de cavalos

Mesmo com essa ajuda, as probabilidades continuam bem desfavoráveis. Um dos maiores promotores ingleses de apostas concluiu que nenhum usuário de microcomputador jamais ganhou qualquer grande prêmio. "Se houvesse um sistema que funcionasse mesmo, eu seria o primeiro a saber", afirma ele. Embora as entradas sejam feitas por máquinas automáticas especiais, sua empresa de jogos só usa computadores para manter os registros.

As corridas de cavalos parecem oferecer variedade maior ao programador. O estudante inglês David Stewart criou um programa de microcomputador destinado a prever os vencedores dos páreos. Escrito para o Sinclair ZX-81 e adaptado para o Spectrum, o programa obteve êxito muitas vezes. Embora os palpites de David sejam divulgados por diversas estações de rádio, ele não amealhou grande fortuna.

Rodando a roleta
Numa roleta, é o número 0 que dá lucro ao cassino. A probabilidade de se acertar em cheio num dos outros 36 números é de 1/37. Praticamente nada pode ser feito para melhorar as chances básicas do jogador.

Significativamente, os profissionais de corridas de cavalos rejeitam o uso de computadores. Na Inglaterra, por exemplo, os prognósticos oficiais ainda são feitos a mão pelo Jóquei Clube (embora os dados fiquem armazenados no computador). *Timeform*, revista especializada de grande aceitação entre os apostadores, também compila a mão a maior parte de seus dados. "Só usamos o computador para calcular o tempo padrão de cada cavalo, levando em consideração a resistência do vento", explica um diretor da publicação. "Não existe nenhum sistema de prognósticos verdadeiramente computadorizado. Os computadores não conseguem dar conta dos resultados extraordinários que surgem todo dia."

O uso de computadores também aumentou do outro lado do balcão de apostas, mas não para calcular as chances. As equipes contratadas pelas grandes cadeias de lojas de *bookmakers* britânicos empregam calculadoras especiais para computar o retorno sobre as apostas. O setor de crédito do negócio está ficando cada vez mais computadorizado. O apostador que tenha conta numa das lojas pode fazer sua aposta por telefone, no centro de computação. Os detalhes são digitados e debita-se o valor da aposta na conta do cliente. Se o cavalo escolhido tiver o desempenho previsto, os ganhos são calculados e creditados na mesma conta.

Os *bookmakers*, na maioria, não acreditam em sistemas de prognósticos por computadores. "Ninguém jamais apareceu com um computador que ganhasse sempre. Caso contrário, não estaríamos aqui", zombam eles.

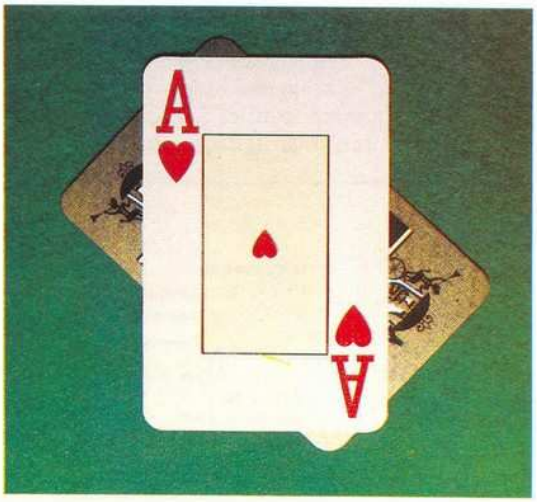
Uma empresa especializada em corridas de cavalos empreendeu extraordinária e controvertida simulação por computador. Registros pormenorizados do histórico dos ganhadores do Derby inglês foram incorporados num programa encomendado especificamente para esse fim, e convidaram-se leitores de jornais a dar palpites para os seis primeiros lugares. O resultado foi uma polêmica em torno da colocação do cavalo italiano Ribop, que nunca havia perdido uma corrida: a máquina colocou-o em quarto lugar!

Talvez o mais famoso "computador" para jogos de azar seja o ERNIE (Electronic Random Number Indicator Equipment, máquina eletrônica indicadora de números randômicos), que escolhe os ganhadores de uma loteria, na Inglaterra. Embora não seja programável, o ERNIE executa um programa. A máquina foi desenvolvida por Plessey em 1973 e sua função consiste em gerar randomicamente 200.000 números e gravá-los em fita magnética. Esses números são gerados a partir de uma série que começa pelo número de emissão mais baixo e termina com o mais alto. Então a fita é colocada num computador ICL de grande porte, que compara os números com uma listagem em fita dos números que já foram premiados. Eliminados estes números que não podem ser sorteados, o computador imprime os certificados dos prêmios e as cartas para os ganhadores. A probabilidade de um número ser sorteado na extração mensal é de apenas 1/15.000.

Pretas X brancas

Outro jogo de probabilidade aparece em jornais diários britânicos. As chances de ganhar um dos prêmios de 1 milhão de libras oferecidos nas promoções de jornais são ainda mais remotas. Os números dos cartões distribuídos aos leitores contêm doze dígitos. Uma seqüência de doze dígitos que vai de 000000000000 a 999999999999 oferece 1 bilhão de combinações. Estatisticamente, a chance de sorteio de determinado número seria pouco melhor que 1/1 bilhão, pois o jornal envolvido publica dois números por dia. Nessa base, é bem duvidoso que o jornal tenha de pagar o grande prêmio, pois não vende 1 bilhão de exemplares diários.

A situação pode ser melhor visualizada se você imaginar um saco que contenha 2,5 milhões de bolas brancas, representando os competidores (o número de cartões em circulação) e 1 bilhão de bolas pretas para o número total de combinações possíveis. As chances de sortear uma bola branca na primeira vez são bastante remotas. E não melhoram muito quando são tiradas as bolas correspondentes a um ano de extração. Os estatísticos computam em 1/667 as chances de o jornal ter de pagar o prêmio de 1 milhão de libras.



Lance de sorte
Jogos de baralho proporcionam um dos melhores campos para o desenvolvimento de programas que possam ajudar a sorte.
A capacidade de memorizar as cartas já jogadas aumenta as chances de sucesso do apostador.
Os cassinos não permitem o uso de computadores nas mesas de jogo, mas em todas as grandes façanhas foram usados micros escondidos no corpo do apostador ou ligações radiofônicas com máquinas operadas fora do salão de jogo.

Jogo de dados

A maior parte das versões de BASIC possui função de geração de números randômicos. Contudo, em muitos casos, os números assim gerados não são de fato randômicos, como demonstra este programa:

```
10 LET A = RND
20 LET B = RND
30 LET C = RND
40 PRINT A,B,C
```

Nas três primeiras linhas, um número supostamente randômico é designado para as variáveis A, B e C. Depois elas são impressas. Isso pode fornecer os seguintes resultados, entre outros:

```
,014007      ,964370      ,457397
```

Mas, se você tornar a executar o programa, a maioria dos microcomputadores mostrará de novo a mesma seqüência. Acontece que quando pedimos RND, o computador responde com o próximo número, numa seqüência fixa de números. Isso pode compreender 1 milhão de frações de seis dígitos entre 0,000000 e 0,999999, cada um ocorrendo uma vez no ciclo completo (mas não em seqüência).

Alguns tipos de BASIC usam sintaxe um pouco diferente, exigindo uma expressão em parênteses, chamada "argumento", que assume a forma LET A = RND(X). O efeito mostra-se bem semelhante: RND e RND(X) podem ser utilizadas da mesma forma que outras variáveis.

Versões diferentes de BASIC também apresentam a função RANDOMIZE, que faz a seqüência começar num ponto imprevisto. Inserindo o comando RANDOMIZE em qualquer programa onde o RND seja usado, assegura a geração de uma seqüência diferente de números cada vez que o programa for executado (RUN).

Para simular o arremesso de um dado, precisamos de números inteiros, que vão de 1 a 6. No entanto, devem-se eliminar as frações. Isso é feito por meio do uso da função INTeger (número inteiro). PRINT INT(6,99) produz o resultado 6 tão certamente quanto PRINT INT(6,01). Qualquer coisa que venha depois da vírgula é sempre descartada, por definição.

Como o maior número que RND pode gerar é 0,999999 (o qual, quando expresso como um número inteiro, assume o valor 0), uma pequena multiplicação se faz necessária. A fórmula tradicional é:

```
LET A = INT(6*RND)+1
```

Multiplicamos por 6 porque um dado tem seis faces. O "mais um" assegura que os resultados vão de 1 a 6, e não de 0 a 5.