MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

Cr$ 6.500

5

O rifle leve • Micros portáteis
Inteligência artificial

rioGráfica

MANAUS, ALTAMIRA, MACAPÁ, JI.PARANÁ, PORTO VELHO, RIO BRANCO, SANTARÉM, BOA VISTA, SINOP, ALTA FLORESTA: Cr$ 8.450

MICROCOMPUTADOR CURSO PRÁTICO

TESTE

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

1) O que é um programa heurístico?
2) Quais foram os jogos lançados pela Atari que obtiveram maior sucesso?
3) Os computadores portáteis costumam incorporar um processador de texto. Onde ele fica armazenado?
4) Qual o tipo de arquivo que permite acesso direto a qualquer dado?
5) Qual o microcomputador cujo teclado dispõe de um pequeno LCD?

RESPOSTAS DO TESTE ANTERIOR

1) O Macintosh utiliza o mouse para movimentar o cursor.
2) Num loop do tipo WHILE o teste de saída vem antes das instruções; no tipo REPEAT, vem depois.
3) A impressora matricial conta com pequenas agulhas dispostas numa matriz, que se projetam de encontro ao papel para formar os caracteres.
4) O despistador converte informações escritas numa mensagem enviada silenciosamente pelas linhas telefônicas.
5) Os diagramas de Venn são representações gráficas de expressões da álgebra booleana.

PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de trinta fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 terá doze páginas e quatro capas, e incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores.
As duas primeiras capas são descartáveis; as 3.as e 4.as trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

COMO ENCADERNAR

As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 25 (**Volume 2** e **Jogos**).
Volume 1 - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.
Volume 2 - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.
Jogos - Guardas; frontispício e sumário (que constituem as 3.as e 4.as capas do fascículo 30); 3.as e 4.as capas dos fascículos de 1 a 29; e guardas.

Nota: Muitos dos equipamentos que aparecem na seção RAIO X não são normalmente comercializados no Brasil. Sua divulgação é feita a título de informação ao leitor.

rioGráfica **Diretoria Executiva:** Oscar Neves; Filipe Zander; Nilo Sérgio de Almeida; Danilo Esteves Costa; **GRUPO EDUCAÇÃO E CULTURA - Planejamento e Comercialização - Diretor:** Roberto Combochi; **Gerente de Marketing:** Jaime Rodrigues; **Gerente de Produto:** José Mauro Porto; **Gerente do P.C.P.:** Aylton Menezes; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgar Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto - **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Maurício Rittner; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Revisores:** Isis Augusta Loyolla; Cacilda Guerra; Luiz Vicente Vieira Filho; Marcos Domingos Agathão; Maria da Graça Mendonça Couto; **Diretor de Arte:** Aníbal Monteiro; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistente de Arte:** Miguel Luis Escámez Simón - **Colaboradores - Editores-assistentes:** Isa Mara Lando; Luiz Carlos Pizarro Marin; Claudio Marcondes; **Texto:** Carlos Eduardo Silveira Matos; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Maria Ester M. Marinho; Maria Clara Cescato; Patrícia de Paiva e Castro; Claudio Marcondes; **Pesquisa:** Stella Maria Quentel; **Arte:** Toshio Morobosi; **Assessoria Técnica:** Durval Junqueira de Aquino; Lauro de Lauro; Luiz Henrique Alayon; Luiz Antônio Meloni (LOGO). As fotos não creditadas pertencem à obra original. Seguem-se os créditos das demais: Fernando Scavone. © APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa. **Composição:** ERG Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

NÚMEROS ATRASADOS - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electroliber Ltda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.

OS PESOS LEVES

Em 1981, Adam Osborne lançou o primeiro micro portátil. Quatro anos depois, o Osborne e os outros modelos nele inspirados parecem incômodos e pesados, diante de equipamentos menores que uma lista telefônica.

O que é um microcomputador "portátil"? Não é tão simples responder: o alcance do qualificativo teve de ser revisto desde o aparecimento da última geração de computadores "de mão". Na verdade, os micros portáteis introduzidos no início dos anos 80 podem agora ser chamados, no máximo, de micros "transportáveis". Em 1985, os microcomputadores portáteis são os que possuem sua própria fonte de energia, tela e dispositivos para armazenamento de dados num conjunto menor que uma lista telefônica.

O Epson HX-20 foi o primeiro a oferecer esse grau de "portabilidade". Hoje, porém, seu pequeno visor de cristal líquido com capacidade de vinte caracteres por quatro linhas denuncia a idade do projeto. Os portáteis mais recentes, como o Tandy 100, o NEC PC 8201-A e o Olivetti M 10, têm preços similares, mas podem mostrar quatro vezes mais caracteres no visor.

Então, o que podem fazer esses computadores? Quais as suas vantagens e desvantagens em relação aos micros convencionais de mesa?

O motivo mais óbvio para se adquirir um micro portátil é o acesso total à capacidade de processamento a qualquer hora e em qualquer lugar. Muita gente passa grande parte do tempo longe de seu computador; horas improdutivas são gastas em escritórios de clientes, quartos de hotel, aeroportos e trens. O computador portátil permite que todo esse tempo seja aproveitado.

A última geração de portáteis oferece recursos de computação suficientes para trabalhos científicos ou de engenharia, cálculos, gerenciamento financeiro e processamento de texto — praticamente para todas as aplicações em que se usam computadores pessoais convencionais. Além disso, podem ser conectados a impressoras, modems e outros periféricos.

Os computadores portáteis trazem normalmente pelo menos três programas incorporados em chips de memória — um interpretador BASIC, um processador de texto e um software de telecomunicações. O Tandy 100 e o Olivetti possuem, além destes, programas de endereço e de agenda, que permitem encontrar endereços, números de telefone e compromissos diários.

O programa de telecomunicações é extremamente importante, pois permite que o portátil tenha acesso a outros micros e bancos de dados remotos pela rede telefônica. Com esse recurso, o micro pode transformar-se num terminal de telex ou num transmissor-receptor de correio eletrônico: basta que se utilize um modem ou um acoplador acústico. Dessa maneira, um executivo em viagem pode manter-se em contato com o escritório central, ou um jornalista em trânsito pode escrever seus artigos no computador e transmiti-los imediatamente ao computador do jornal.

Os portáteis mais caros, como o Sharp PC-5000 e o Epson PX-8, utilizam os sistemas operacionais MS-DOS e CP/M, adotados por seus equivalentes de mesa. Estão, portanto, capacitados a rodar inúmeros programas administrativos e financeiros.

Em viagem
O uso de computadores durante as viagens vem se tornando um hábito cada vez mais comum entre profissionais. A nova geração de micros portáteis permite que os homens de negócios ganhem um tempo extra na confecção de textos, enquanto, por exemplo, correm de táxi para o aeroporto. Já os vendedores podem calcular na hora um orçamento — operação que normalmente exige vários dias para ser realizada.

Os executivos em trânsito podem, usando um modem e a rede telefônica, transmitir dados à sede de sua empresa ou, retornando ao escritório no fim do dia, enviar os dados diretamente a um computador de maior porte.

O Epson PX-8 traz o processador de texto WordStar já residente em seus chips de ROM. O Sharp utiliza cartuchos de memória de bolha, que garantem, cada um, 128 Kbytes de memória extra. Esses cartuchos lidam com os dados mais rapidamente que as unidades de disco.

Os programas aplicativos dos portáteis mais baratos são normalmente carregados em sua RAM a partir de fitas cassete, um processo muito mais lento do que o dos cartuchos de memória de bolha ou dos disquetes. O NEC PC 8201-A vem com uma fita cassete que contém aplicativos para cálculos, formatação de textos, gerenciamento de investimentos e avaliações de empréstimos. O programa de cálculos transforma a máquina numa calculadora capaz de memorizar até 99 entradas. O formatador prepara para impressão os materiais elaborados pelo editor de texto, especificando a largura das margens, dividindo o texto em páginas, numerando as páginas etc. Com o programa de investimentos, o micro pode analisar um conjunto de até cinqüenta deles, calculando ganhos e perdas.

O micro portátil tem ainda como características alimentação por baterias, visor próprio e o processador de texto e software de comunicações que traz na ROM.

Isso não acontece com o Apple IIc e o Apricot, geralmente anunciados como portáteis. Esses micros precisam ser ligados à rede elétrica, conectados a um monitor de vídeo e seus programas são carregados na RAM a partir de um disco. Apesar do tamanho e do peso reduzidos, estão mais próximos dos computadores de mesa do que dos verdadeiros portáteis, pertencendo à categoria "transportável".

Além de suas baterias principais, os micros portáteis possuem pequenas pilhas de cádmio e níquel, que constituem uma fonte de energia em caso de emergência. Isso é essencial, uma vez que todos os dados seriam perdidos no caso de acabarem ou falharem as baterias.

A maioria dos portáteis tem também uma interface para leitura do código de barras, permitindo sua utilização no controle de estoques. Ao passar-se a leitora sobre o código de barras impresso na embalagem dos produtos, ela decodifica as informações de preços e datas. Tais informações são processadas pelo computador, facilitando aos lojistas a manutenção de um arquivo atualizado das mercadorias.

Tanto o Tandy Modelo 100 como o NEC PC 8201-A e o Olivetti M 10, mostrados nestas páginas, dispõem de leitoras de código de barra, o que não chega a surpreender: produzidos pela mesma fábrica japonesa, os três apresentam muitas semelhanças. Existem, porém, diferenças significativas entre eles. O Olivetti é o único com tela inclinável. O NEC tem menos software incorporado, mas sua memória pode ser expandi-

Epson HX-20
Embora dotado de um visor pequeno, o HX-20 tem a vantagem de trazer, incorporados, um gravador cassete e uma miniimpressora. Um processador de texto com recursos básicos também está incluído.

Casio FP-200
O Casio é o mais barato de todos os computadores portáteis. Todavia, não possui processador de texto, fornecendo, em vez disso, uma planilha eletrônica rudimentar.

da até 64 Kbytes: o dobro da possibilidade de expansão de memória do Tandy e do Olivetti. Além disso, o NEC pode utilizar cartuchos sobressalentes com 32 Kbytes de memória, que retêm os dados mesmo quando removidos do computador.

Assim como as máquinas de escrever portáteis não tornaram obsoletas as pesadas máquinas de escritório, os micros portáteis não pretendem tomar o lugar dos microcomputadores de mesa. Para começar, seus pequenos visores de cristal líquido (LCD) os tornam pouco adequados a longas sessões de trabalho. O LCD é de leitura mais difícil e responde mais lentamente à digitação que os vídeos de raios catódicos.

Outra desvantagem dos computadores portáteis são os teclados planos, mais cansativos de usar. E os modelos de preço mais acessível não rodam os programas aplicativos comerciais populares.

Apesar de tudo isso, é inegável que os computadores "pesos leves" vieram para ficar. Com o uso generalizado dos micros, um número cada vez maior de pessoas está descobrindo que, enquanto um computador pode ajudar a administrar suas atividades com maior eficiência, os portáteis lhes permitem ter acesso a esse recurso estejam onde estiverem. Não vai demorar muito para que os micros portáteis se tornem tão populares quanto as calculadoras de bolso.

Epson PX-8
Esta máquina pode rodar softwares comerciais com o sistema CP/M, inclusive o processador de texto WordStar, que vem junto com a máquina.

Tandy TRS-80 100/NEC PC 8201-A/Olivetti M 10
Três versões diferentes do mesmo micro. Possuem em comum um excelente processador de texto incorporado e razoável gama de interfaces. Também na ilustração, um modem a bateria da Olivetti.

Modelo	Memória padrão	Memória máxima	Visor	Peso
Casio FP-200	8 K	32 K	8x20	1,400 kg
Epson HX-20	16 K	32 K	4x20	1,800 kg
Epson PX-8	64 K	64 K + 120 K*	8x80	2,300 kg
NEC PC 8201-A	16 K	64 K + 32 K*	8x40	1,800 kg
Olivetti M 10	8 K	32 K	8x40	1,800 kg
Tandy TRS-80 Modelo 100	8 K	32 K	8x40	1,800 kg

* O NEC aceita um cartucho de RAM com 32 K e o Epson PX-8, um disco com 120 K.

NA MIRA DO RIFLE

Projetado para aumentar o realismo de jogos do tipo tiro ao alvo nos micros, o rifle óptico combina a aparência de uma arma com um sistema semelhante ao de uma câmara. Assim, o usuário pode dispensar o joystick.

O principal componente do rifle óptico da Stack (SLR, Stack Light Rifle) é a pistola de alvo eletrônico conectada ao computador por um longo cabo condutor. No terminal do computador, dependendo da versão, há um conector para o encaixe adequado.

Na versão do rifle para o micro ZX da Spectrum, o conector contém dois chips e alguns componentes simples para ligar a parte eletrônica interna da arma ao computador.

A pistola é provida de uma elegante coronha de ombro que se prende em sua parte traseira, um cano de rifle e uma imitação de mira telescópica, complementos que aumentam muito a precisão do tiro.

A parte eletrônica da pistola consiste num detector de luz ou fotodiodo, um pequeno amplificador e um buffer. A luz que vem pelo cano do rifle é focalizada por uma lente plástica sobre o fotodiodo, e o dispositivo detecta mudanças na intensidade da imagem. Depois de passar pelo amplificador, o sinal (uma forma ondulatória analógica) transforma-se num pulso digital transmitido ao computador quando se pressiona o gatilho.

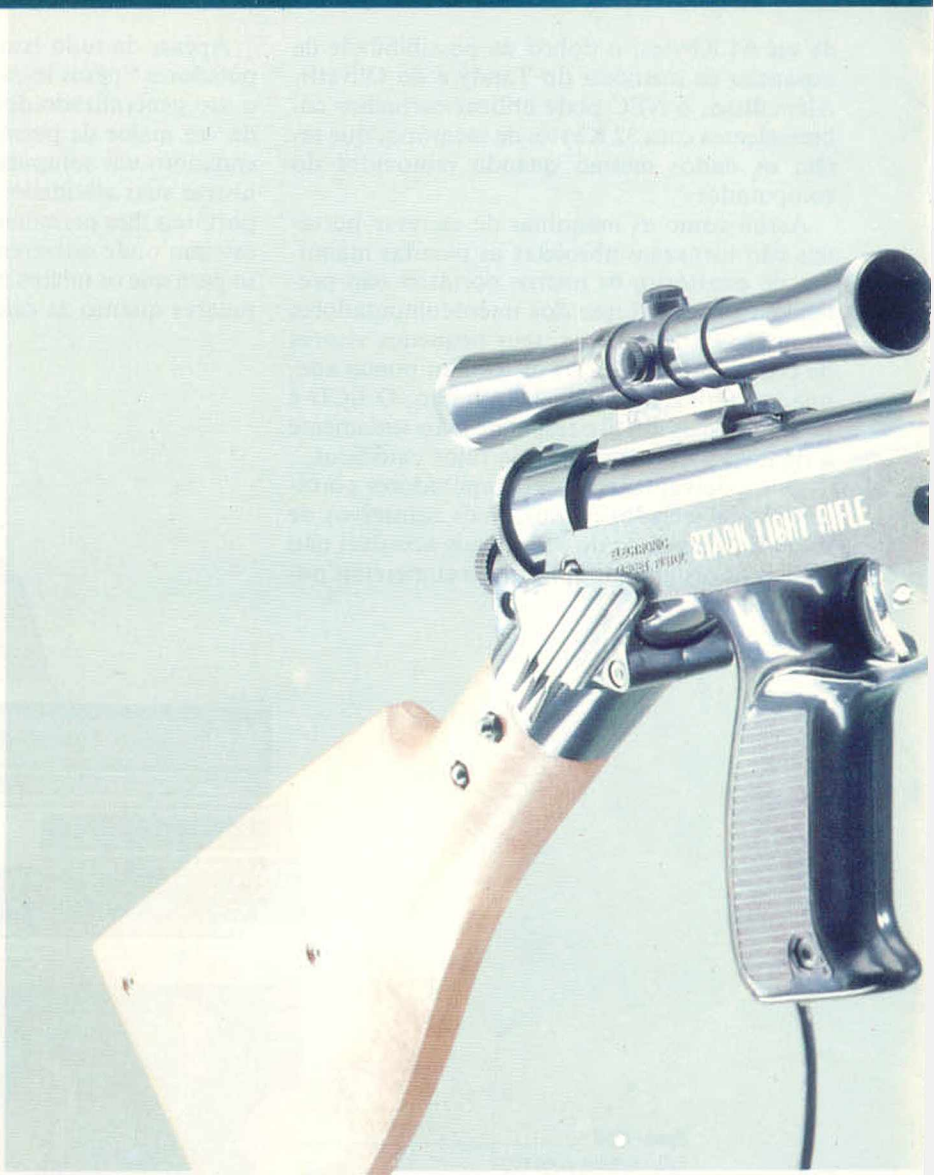
A posição da tela observada no momento é aquela para a qual o rifle está apontando. Assim que recebe o pulso do SLR, o computador compara a posição do rifle naquele instante com a posição do alvo na tela. Se forem correspondentes, o jogador marca o ponto.

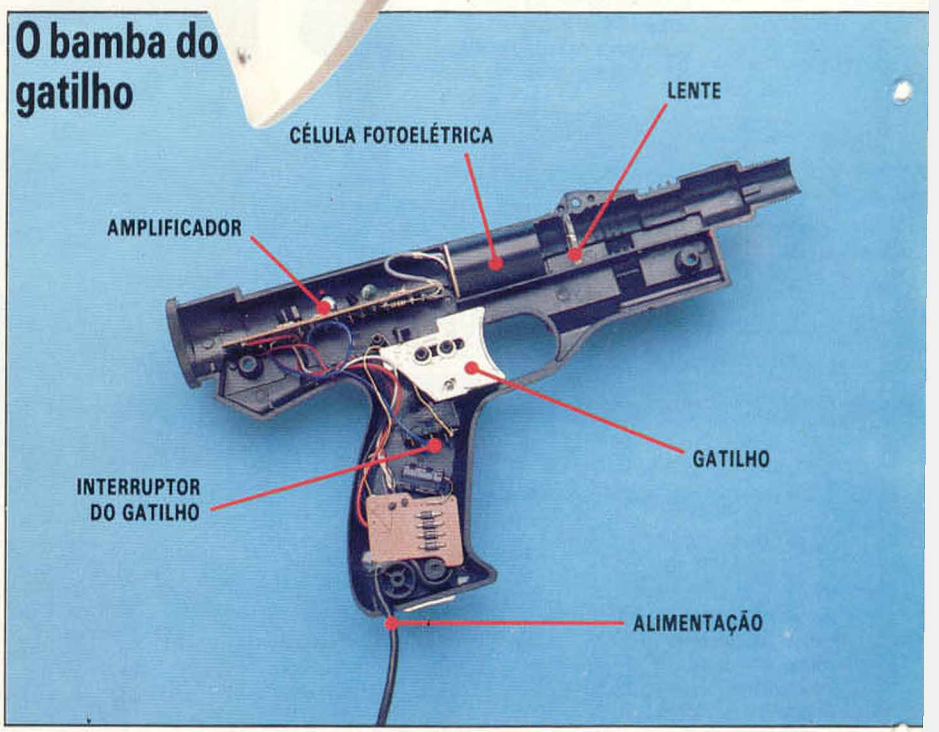
Disparo óptico

Existem variações do rifle óptico para o ZX da Spectrum, o Vic-20 da Commodore, o Commodore 64 e compatíveis. Todas desempenham as mesmas funções. Quanto ao software, a Stack fornece apenas três jogos em cassete com o SLR. Várias empresas de desenvolvimento de software fazem jogos apropriados a esse tipo de dispositivo, mas poucas produziram ou converteram programas para funcionar com ele.

A Stack não fornece programas utilitários que permitam ao usuário escrever seus próprios programas, nem divulga informações técnicas detalhadas sobre o funcionamento do rifle.

O SLR baseia-se no mesmo princípio de operação da caneta óptica. Contudo, é muito maior





e seu projeto prevê que seja mantido a uns 3 m da televisão, e não que esteja em contato com a tela. Para ajudar na filtragem da luz do ambiente, o SLR é provido de um tubo escuro (o cano da arma) e uma lente. Estes se combinam para alcançar um alto grau de precisão e permitem ao usuário atirar estando confortavelmente sentado numa poltrona. Os jogos disponíveis no mercado são, na verdade, exemplos pobres do que seria possível fazer; o uso de gráficos e a lógica e a ação dos jogos não sobressaem.

Ação congelada

Um dos maiores problemas na programação de canetas ópticas ou versões gigantes como o SLR é que os programas precisam ser muito eficientes. Em todos os cartuchos fornecidos pela Stack, os jogos param por um momento quando o gatilho é acionado. Isso porque, se a tela fosse varrida continuamente, como em geral ocorre com a caneta óptica, a velocidade do jogo diminuiria muito. Portanto, quando se aciona o gatilho, o software congela a ação para determinar se o alvo (na tela) está alinhado com a posição da arma. Uma vez que o software tenha determinado se o jogador acertou ou não o alvo, o jogo pode continuar.

Teoricamente, quando o gatilho é acionado, a quantidade de instruções necessárias para estabelecer a posição da tela em relação à posição detectada pela arma deveria ser muito pequena. Mas, observando-se o software em ação, percebe-se que nem sempre é esse o caso: às vezes a quantidade exigida é bem grande.

A provisão de recursos para a caneta óptica dentro do chip de vídeo tornaria a tarefa do software muito mais simples. O Commodore 64 oferece tal sistema, mas o ZX Spectrum não possui esses recursos, e a deficiência aparece quando se trata de calcular a posição do rifle, após o acionamento do gatilho.

Um tiro no escuro

A fotocélula detecta um ponto luminoso que se move varrendo a tela continuamente. O software tem o controle contínuo das coordenadas do ponto na tela; quando o rifle atira, o software pode comparar o valor das coordenadas com a posição do alvo para o qual o rifle estava apontado.

Temporada de caça

High Noon (Matar ou Morrer, também nome de um filme exibido no Brasil) é o melhor dos três programas de demonstração da Stack. A animação é boa e o pistoleiro da tela atira no jogador de modo verossímil. O Grouse Shoot (Tiro ao Faisão) e o Shooting Gallery (Tiro ao Alvo) apresentam um único alvo móvel, que precisa ser atingido antes que saia dos limites da tela. A animação é irregular e o acionamento do gatilho causa notável parada na tela.

SELEÇÃO ALEATÓRIA

Os arquivos de acesso aleatório, mais rápidos de manipular que os seqüenciais, exigem maior espaço de armazenamento, além de estruturação cuidadosa e rigorosamente uniforme.

As limitações dos arquivos seqüenciais devem-se à obrigatoriedade de ler as informações na ordem em que foram gravadas. Os arquivos de acesso aleatório ou direto não apresentam tais limitações porque os registros são acessados em qualquer ordem e com muita rapidez. A palavra "aleatório" não implica construção ou uso dos arquivos de maneira caótica. Significa apenas que qualquer segmento pode ser gravado ou lido sem a necessidade de passar pelas informações precedentes.

Os arquivos mantidos em fitas cassete, no entanto, são seqüenciais. Assim, não há possibilidade de acesso direto a um item de dados. O único modo de usar o acesso aleatório, nesse caso, consiste em carregar todos os dados na memória, mas isso limita o tamanho do arquivo. É preciso dispor de unidades de disco para se conseguir acesso aleatório útil; mesmo assim, algumas marcas de unidades de disco não admitem esse tipo de manipulação de arquivo.

Acesso aleatório x seqüencial

	ARQUIVOS DE ACESSO ALEATÓRIO	ARQUIVOS SEQÜENCIAIS
PRÓS	• Acesso rápido a registros específicos	• Conservam espaço • Disponíveis para sistemas de fitas
CONTRAS	• Desperdício de espaço • Exigem discos	• Lentos
APLICAÇÕES PARA AS QUAIS SÃO ADEQUADOS	• Dados previsíveis que estejam em formato definido • Quando pequenas quantidades de registros diferentes são acessadas; por exemplo, numa biblioteca onde os usuários solicitam detalhes sobre determinados livros. Esse é um aplicativo de baixa taxa de acerto.	• Grandes quantidades de dados não estruturados • Quando a maioria dos registros de um arquivo é processada; por exemplo, num sistema de folha de pagamento, em que cada empregado deve ser pago. Esse é considerado de alta taxa de acerto.

Os arquivos de acesso aleatório devem ser divididos em registros e campos. Para acessar o arquivo, especifica-se o registro requisitado, que, juntamente com os campos, será colocado num buffer na memória do computador. Aí, os campos podem ser apagados, corrigidos ou impressos. As estruturas mais complexas ficam a cargo do sistema operacional, que em curto tempo localiza o início de determinado registro no disco. Para facilitar a rápida localização, os registros têm o mesmo comprimento. Se cada um tiver 100 bytes de caracteres de comprimento e o programa receber instruções para gravar o número 83, o sistema operacional posicionará a cabeça do disco no início do byte de n.º 8.300 do arquivo. Ele possui um registro de quantos bytes estão em cada setor do disco, o que lhe permite calcular a posição do registro procurado. Esse método de leitura de arquivo pode parecer complexo e lento, mas é muito mais rápido que num arquivo seqüencial.

Ao padronizar os arquivos, é necessário escolher o tamanho que acomodará o mais longo dos registros. Os menores são preenchidos, em geral, com espaços (32 em código ASCII). Esse é o maior problema dos arquivos aleatórios, pois o preenchimento necessário para deixar o registro no tamanho escolhido acarreta um desperdício de valioso espaço de armazenamento. Por isso, reservam-se os arquivos aleatórios para pequenas quantidades de informação, quando o acesso precisa ser muito rápido, e deixam-se os arquivos seqüenciais para o armazenamento em massa.

Também os campos dentro de um registro devem ter dimensão padronizada, em especial nos sistemas que oferecem o recurso de acesso aleatório para campos e registros específicos. Mesmo nos sistemas sem tal recurso, esse procedimento constitui um modo mais organizado e eficiente de definição de arquivos. Quando se projeta um arquivo de acesso aleatório, o primeiro passo consiste em relacionar os diferentes campos e escolher tamanhos apropriados para eles. Por exemplo, o campo para o nome de uma pessoa compreende pelo menos vinte caracteres de comprimento, enquanto para armazenar sua idade bastam dois.

A economia é essencial ao se projetar um arquivo, pois sempre haverá intercâmbio entre o total de informações armazenadas e o número de registros diferentes. É freqüente que os sistemas de codificação sejam projetados para reduzir o espaço tomado pelos dados. Por exemplo, os códigos 1, 2 e 3, para preto, vermelho e verde, ou códigos de dados, como 841011, para 11

de outubro de 1984. No entanto, os sistemas de codificação devem permanecer internos ao sistema, e os programas precisam reconverter os códigos para uma forma de fácil compreensão, quando um campo é introduzido ou apresentado na tela.

Comprimento dos arquivos

A maioria dos sistemas limita o comprimento disponível — pode variar de 128 bytes a até 2.048 bytes. Além disso, muitas vezes é mais eficiente escolher um tamanho múltiplo do setor — usam-se, em geral, números como 64, 128, 256 ou 512.

Evita-se, assim, que os registros individuais se dividam, abrangendo mais de um setor — ou seja, reduz-se drasticamente o número de acessos ao disco.

A manipulação dos arquivos aleatórios costuma ser bem mais simples que a dos arquivos seqüenciais. Em ambos os sistemas, é necessário manter uma contagem atualizada do número de registros; e muitas vezes, nos arquivos de acesso aleatório, usa-se o primeiro registro (em geral, o de n.º 0) para armazenar essa e outras informações relevantes, como a data de criação do arquivo. A rígida estrutura do campo e do registro seria descartada para esse registro.

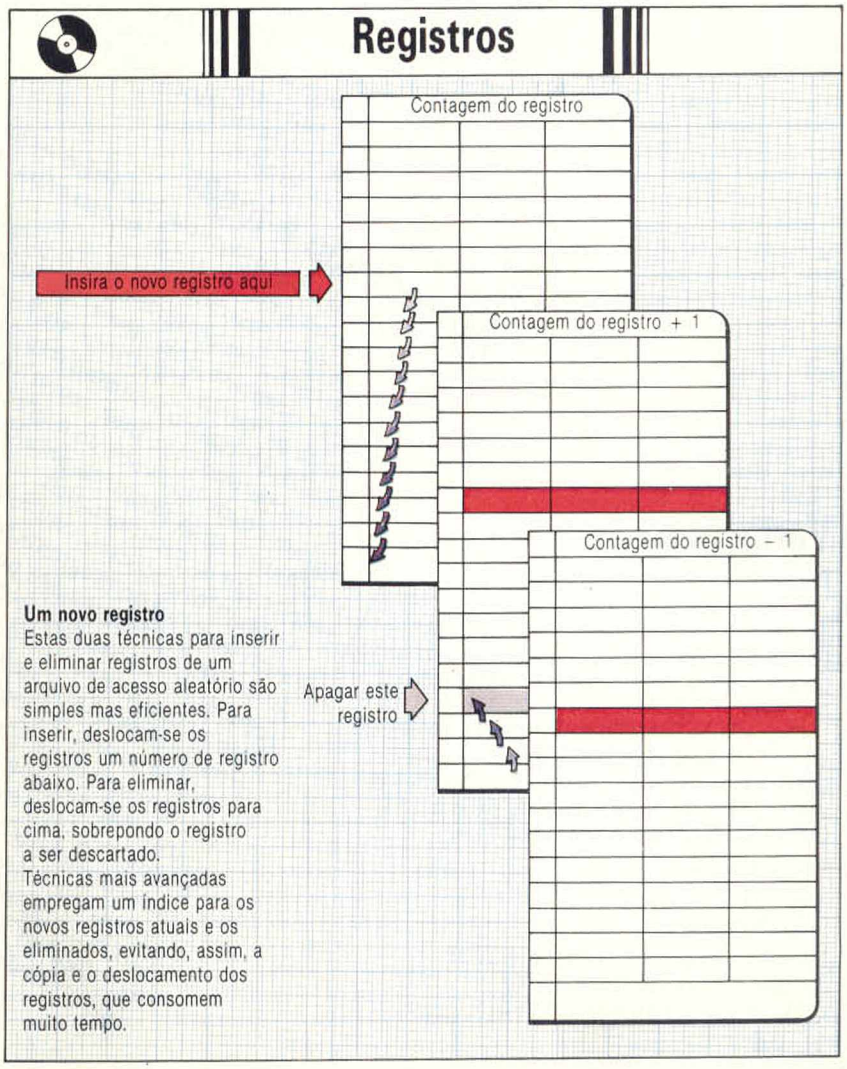
A leitura do registro é feita pelo número, segundo técnicas semelhantes às usadas para pesquisar dados de matrizes em BASIC. Muitas vezes, usa-se como chave para o arquivo um campo específico, como, por exemplo, o campo do nome. O computador lê o campo-chave e monta um índice que identifica onde estão armazenados os vários nomes.

Os arquivos aleatórios não indexados costumam ser acessados registro por registro, como os arquivos seqüenciais. Mas, se os registros estão ordenados pelo campo-chave, podem-se usar métodos rápidos de busca. Imagine que se pretenda encontrar "João" num arquivo ordenado pelo nome. Começa-se a procura no meio do arquivo e descobre-se, por exemplo, que o nome aí registrado é "Paulo", que vem depois de "João", na ordem alfabética. Assim, elimina-se o restante do arquivo. O palpite seguinte será um registro no meio dessa primeira metade do arquivo. O nome poderá ser "Homero", exigindo novo avanço, e assim por diante. Tais técnicas tornam-se, às vezes, bastante sofisticadas. O desempenho de muitos programas aumenta quando se mantêm na RAM os registros mais usados. Como resultado, os registros são localizados e armazenados em arquivos grandes a alta velocidade.

Eliminar e inserir novos registros pode ser relativamente lento. O método mais simples de eliminar consiste em copiar o registro seguinte em seu espaço, sobrepondo as informações nele contidas. Cada registro subseqüente é copiado uma posição antes e, afinal, a contagem de registros fica com um a menos. De modo semelhante, insere-se novo registro em qualquer ponto movendo-se o último registro um número para a frente e copiando um espaço à frente todos os registros entre ele e o número do novo registro. Isso cria um espaço em que o novo registro pode ser gravado.

Nenhuma dessas técnicas é rápida, embora todas sejam mais eficientes que operações semelhantes com arquivos seqüenciais. As inserções e eliminações de registros ficam mais rápidas se o arquivo tiver um índice separado. Marca-se no índice que o registro foi eliminado. Os dados permanecem inalterados. À medida que se acrescentam novos registros, eles podem ser encaixados em arquivos não utilizados ou eliminados e o índice é atualizado.

Há duas vantagens ainda a considerar no sistema de arquivo de acesso aleatório. Em primeiro lugar, embora seja mais rápido ler e gravar grupos de registros juntos, os arquivos podem sair da ordem. A maioria dos programas oferece um recurso de ordenação que organiza os registros em ordem lógica e apaga os registros eliminados. Em segundo lugar, o sistema de marcar registros eliminados oferece segurança, pois permite recuperar esses dados, se necessário. A segurança existirá até que os registros eliminados sejam sobrepostos ou descartados por um programa de ordenação.



Um novo registro
Estas duas técnicas para inserir e eliminar registros de um arquivo de acesso aleatório são simples mas eficientes. Para inserir, deslocam-se os registros um número de registro abaixo. Para eliminar, deslocam-se os registros para cima, sobrepondo o registro a ser descartado. Técnicas mais avançadas empregam um índice para os novos registros atuais e os eliminados, evitando, assim, a cópia e o deslocamento dos registros, que consomem muito tempo.

MAGIA ANIMAL

Neste programa, o computador descobre o animal em que o usuário está pensando. O jogo usa princípios da inteligência artificial e dá à máquina uma aparente capacidade de raciocinar e adquirir conhecimentos.

Este é um jogo em que o computador tenta adivinhar o nome do animal escolhido pelo jogador. Ele o faz por meio de perguntas do tipo "voa?", "tem pêlos?", etc. Você só pode responder "sim" ou "não", e o computador vai ponderando esses dados até se sentir em condições de fazer uma tentativa abalizada. É surpreendente, sobretudo para pessoas não familiarizadas com a informática e com a heurística, que o programa possa fazer isso.

Dois aspectos tornam o programa muito divertido: a capacidade do computador de se comunicar em linguagem compreensível (embora as próprias respostas do usuário a suas perguntas sejam limitadas a sim e não) e o estoque de conhecimento a que o computador pode recorrer para adivinhar o animal.

Aprendendo a jogar

Magia Animal é um programa heurístico — ensina a si mesmo a melhorar seu desempenho enquanto vai sendo executado.

Quando usado pela primeira vez, o programa "conhece" somente dois nomes de animais e uma única pergunta. Conforme você responda sim ou não, ele tenta adivinhar a resposta. Quando o computador faz uma tentativa errada, o programa pede que você dê o nome do animal escolhido e uma pergunta para distingui-lo do palpite que o programa deu.

Essas informações são então acrescentadas ao banco de dados do programa para construir uma árvore de opções, que ele poderá usar no jogo seguinte. Toda vez que você utiliza o jogo, a árvore aumenta de tamanho, até que, afinal, o programa vai acertar a maioria dos animais.

Ainda assim, o programa *não sabe* nada sobre animais. Está seguindo às cegas um guia constituído pelo conhecimento combinado de todos os jogadores que o usaram. As informações poderiam muito bem ser a respeito de diferentes tipos de cerveja, nomes de escritores, peças de motocicleta, sintomas de doenças ou amigos e parentes do jogador. Uma versão do programa que possibilitasse definir a pergunta inicial e duas respostas poderia ser usada em várias tarefas diferentes. Não são os dados que fazem o programa funcionar, mas o modo como vão sendo organizados.

É fácil construir a árvore com um simples programa em linguagem BASIC. A maioria das estruturas desse tipo é mantida em matrizes; no caso, usam-se T$() para as questões e os nomes dos animais e "sim" e "não" para as ligações entre as entradas específicas em T$. Essas ligações formam o trajeto até a resposta.

Para qualquer entrada em T$, o elemento correspondente em S() informa o programa sobre onde buscar, se a resposta à questão for sim. De modo semelhante, o elemento N() é a ligação para a resposta negativa. No final da árvore, o texto em T$() não é uma pergunta, mas o nome de um animal. Nesse caso, tanto S() como N() são colocados em 0 e o programa precisa adivinhar em que o jogador está pensando.

Nossa versão do programa, curta e simples, mostra os princípios envolvidos. Se quiser aperfeiçoá-la, você pode melhorar a apresentação para sua máquina pelo acréscimo de gráficos em cores, de som etc.

Variações

Para a linha Sinclair

Esse programa foi escrito em BASIC para os compatíveis com Apple, podendo ser convertido para a maioria dos micros com adaptações mínimas.

Para a linha Sinclair, as instruções devem vir em linhas separadas, o que implica alterar os GOTOs e os GOSUBs. Todas as atribuições de valor devem ser precedidas pela palavra LET. Outras alterações que você deve fazer:

Inclua estas duas linhas:

```
45 LET L=40
55 T$ (N,L)
```

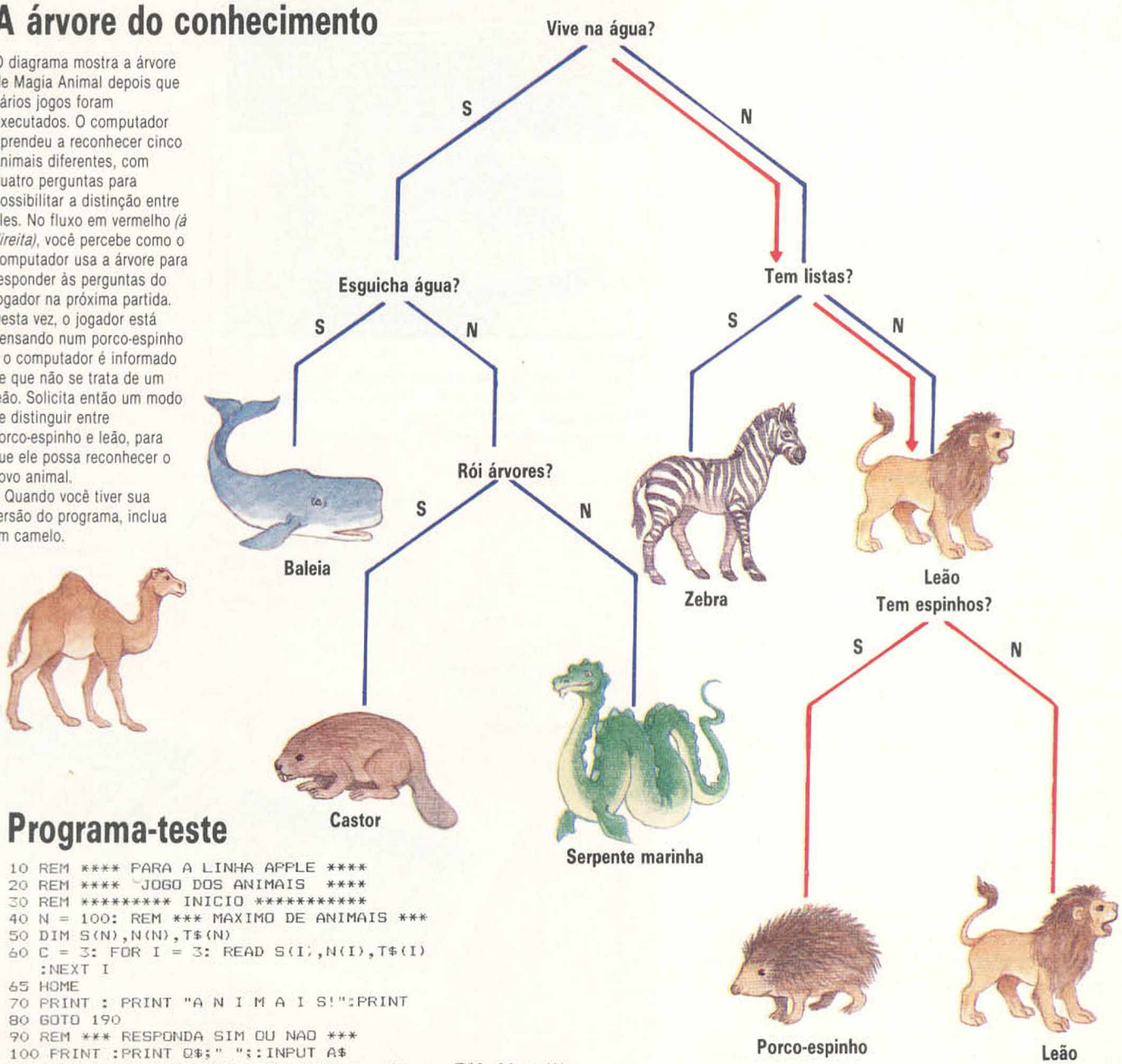
Na linha 200, substitua END por STOP. Escreva a linha 230 deste modo:

```
230 IF Y(P)=0 AND N(P)=0 THEN GOTO
  290
```

A árvore do conhecimento

O diagrama mostra a árvore de Magia Animal depois que vários jogos foram executados. O computador aprendeu a reconhecer cinco animais diferentes, com quatro perguntas para possibilitar a distinção entre eles. No fluxo em vermelho *(à direita)*, você percebe como o computador usa a árvore para responder às perguntas do jogador na próxima partida. Desta vez, o jogador está pensando num porco-espinho e o computador é informado de que não se trata de um leão. Solicita então um modo de distinguir entre porco-espinho e leão, para que ele possa reconhecer o novo animal.

Quando você tiver sua versão do programa, inclua um camelo.



Programa-teste

```
10 REM **** PARA A LINHA APPLE ****
20 REM **** JOGO DOS ANIMAIS  ****
30 REM ********* INICIO **********
40 N = 100: REM *** MAXIMO DE ANIMAIS ***
50 DIM S(N),N(N),T$(N)
60 C = 3: FOR I = 3: READ S(I),N(I),T$(I)
   :NEXT I
65 HOME
70 PRINT : PRINT "A N I M A I S!":PRINT
80 GOTO 190
90 REM *** RESPONDA SIM OU NAO ***
100 PRINT :PRINT Q$;" ";:INPUT A$
110 IF A$ = "SIM" OR A$="S" THEN A = 1:
    RETURN
120 IF A$ = "NAO" OR A$="N" THEN A = 0:
    RETURN
130 PRINT : PRINT " RESPONDA SIM OU
    NAO": GOTO 100
180 REM *** INICIA UM NOVO JOGO ***
190 Q$ = "QUER JOGAR":GOSUB 100
200 IF A = 0 THEN PRINT :PRINT "TCHAU !!"
    : END
210 P = 1
220 REM *** ROTINA PRINCIPAL ***
230 IF S(P) = 0 AND N(P) = 0 THEN 290
240 Q$ = T$(P): GOSUB 100
250 IF A = 1 THEN P = S(P)
260 IF A = 0 THEN P = N(P)
270 GOTO 230
280 REM *** ADIVINHA QUAL E' O ANIMAL ***
290 A$ = T$(P): T$ = A$
300 Q$="E' " + A$: GOSUB 100
310 IF A = 1 THEN PRINT " ACERTEI!!!":
    GOTO 430
320 REM *** APRENDE OUTRO ANIMAL ***
330 PRINT : PRINT " DESISTO!!!":PRINT
    "QUAL E' O ANIMAL ";:INPUT N$
340 A$ = N$
350 PRINT : PRINT "PROPONHA UMA PERGUNTA
    QUE FAÇA A DISTINCAO ENTRE ";A$;" E "
    ;T$: INPUT D$
360 Q$ ="QUAL A RESPOSTA PARA" + T$:GOSUB
    100
370 A$= T$(P): T$(P) = D$: T$ (C+1) = A$:
    T$(C+2) = N$
380 IF A = 1 THEN S(P) = C + 1: N(P) = C
    + 2
390 IF A = 0 THEN S(P) = C + 2: N(P) = C
    + 1
400 S(C + 1) = 0: N(C+1) = 0:S
    (C+2)=0:N(C+2)=0
410 C = C + 2
420 REM *** FIM DO JOGO & LOOP PARA NOVA
    PARTIDA ***
430 A = INT (C/2) + 1
440 PRINT : PRINT "AGORA JA' SEI ";A;"
    ANIMAIS!"
450 GOTO 190
460 REM *** DADOS INICIAIS ***
470 DATA 2,3,"VIVE NA AGUA"
480 DATA 0,0,"A BALEIA"
490 DATA 0,0,"O LEAO"
```

APRICOT

Compacto, mas não verdadeiramente portátil, o Apricot segue a tendência atual de apresentar o gabinete da CPU, o vídeo e o teclado separados. Para amadores, é uma máquina cara.

Com inúmeras características projetadas para atrair o profissional, o microcomputador inglês Apricot destaca-se sobretudo pelo design do teclado e do monitor. Versátil e inovador, o teclado do Apricot conta com o recurso extra da Microtela — um visor de cristal líquido de 2 linhas e 40 colunas — e mais seis teclas programáveis pelo usuário.

Quando se liga a máquina, um programa-teste mostra a quantidade de memória disponível (256 Kbytes é o padrão, mas pode ser ampliado para 768 Kbytes) e solicita ao usuário que insira o disco mestre do MS-DOS. Para usuários não familiarizados com sistemas operacionais como o CP/M ou o MS-DOS, um menu interativo — o Manager ("gerenciador") — possibilita a fácil escolha de softwares aplicativos (incluindo o Supercalc, o Multiplan e o BASIC da Microsoft) ou utilitários (a exemplo do configurador do teclado ou do gerador de caracteres).

As funções atribuídas às teclas programáveis podem ser apresentadas no visor LCD. Assim, os menus de opções exibidos na tela principal são duplicados na Microtela. Tocar a tecla de função apropriada equivale a selecionar o item no menu de tela com o uso das teclas de movimentação de cursor e [Return]. O único inconveniente encontra-se nas teclas tipo membrana, sensíveis ao toque — menos eficientes do que as do tipo máquina de escrever.

Há também as oito teclas habituais de função, marcadas com os nomes de suas funções normais — HELP, PRINT, MENU, FINISH, entre outras. Como as demais teclas do Apricot, essas podem ser reconfiguradas por meio do programa Keyedit (Editor de Teclas) que acompanha a máquina.

O software que acompanha o Apricot consiste na planilha financeira Supercalc e num conjunto amplo de utilitários.

O fabricante fornece poucas informações sobre o hardware, embora os utilitários que o acompanham sejam suficientes para começar a trabalhar o computador. Não há detalhes sobre o mapeamento da memória nem sobre as chamadas do sistema, que poderiam ser necessários ao desenvolvimento de novos softwares.

Apricot XI
Em vista da reduzida capacidade de armazenamento dos microdiscos, a versão Apricot XI incorpora um disco rígido de 10 Mbytes e uma unidade de microdisco.

Tela
Um monitor de fósfo
alta resolução forne
imagem clara e bem

LCD
Um visor de cristal líquido, com 2 linhas, que pode ser usado para mensagens, como relógio, ou calculadora. O relógio permanece funcionando com o micro desligado.

Microdiscos Sony
Dois microdiscos de 315 K permitem o armazenamento compacto e apropriado.

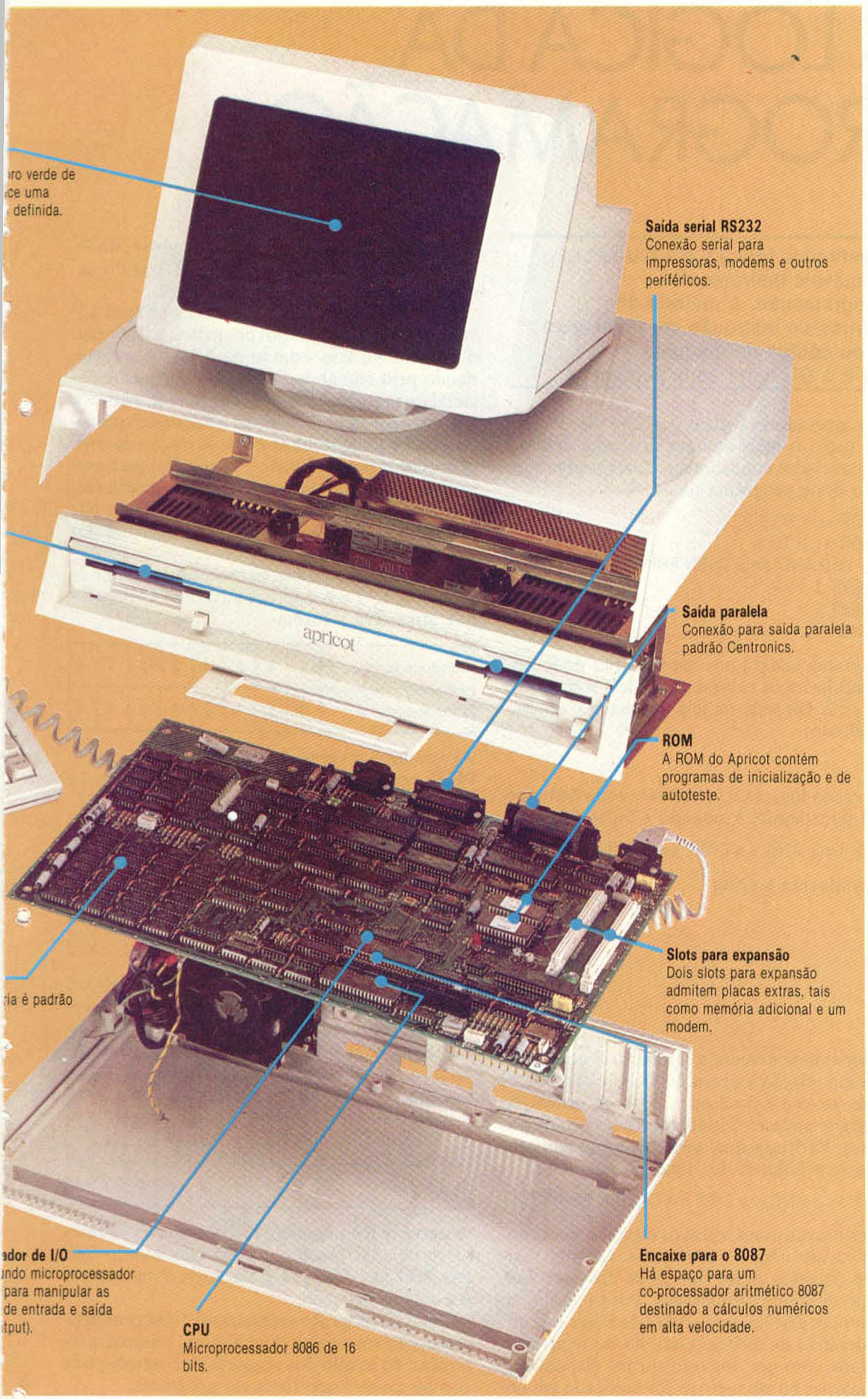
Teclas de função interativas
Proporcionam funções padrão do tipo HELP e REPEAT para programas aplicativos.

Teclas de função tipo membrana
Estas seis teclas podem ser programadas segundo as informações exibidas no visor de cristal líquido, para operar com programas específicos.

RAM com 256 K
Esta ampla memó
no equipamento.

O monitor do Apricot
Uma característica original do Apricot é o monitor de vídeo deslocável em três direções: para os lados, deslizando sobre o gabinete da CPU; em rotação sobre o seu pedestal; e inclinando-se para a frente e para trás, de modo a facilitar a visão sob diferentes condições de iluminação. Apesar das 9" de largura, a tela permite mostrar 132 caracteres de largura por cinqüenta de altura.

Process
Um seg
é usado
funções
(input/o



APRICOT

MICROPROCESSADOR

8086, com opção para processador aritmético 8087.

MEMÓRIA

256 K de RAM, expansível para 768 K.

SISTEMAS OPERACIONAIS

MS-DOS, CP/M-86 e CP/M-86 Concorrente.

VÍDEO

Monitor monocromático, com resolução gráfica de 800 x 400 pixels. No modo texto, a tela apresenta 132 x 50 caracteres, ou 80 x 25. O monitor permite três tipos de movimento, para maior visibilidade.

TECLADO

Destacável, padrão IBM PC, com noventa teclas, mais seis do tipo membrana, programáveis para funções definidas pelo LCD.

ARMAZENAMENTO DE DADOS

O modelo padrão tem duas unidades para microdiscos Sony de 3 1/2", com 315 ou 720 K de capacidade cada um. Há outra versão do Apricot que possui um disco rígido de 10 Mbytes e uma só unidade para microdisco.

INTERFACES

Conectores para saída paralela, serial e mouse. Dois slots para placas opcionais ou modem.

DOCUMENTAÇÃO

Ampla e bem apresentada.

A LÓGICA DA PROGRAMAÇÃO

As operações lógicas E e OU são inestimáveis instrumentos de programação. A maioria dos conjuntos de instrução em linguagem BASIC ou código de máquina inclui E e OU entre seus comandos.

Os dois operadores lógicos E (AND) e OU (OR) têm vários usos; o mais comum relaciona dois ou mais enunciados condicionais. Tente prever o resultado deste programa BASIC:

```
10 FOR I=1 TO 5
20 FOR J=1 TO 5
30 IF I=3 AND J=2 THEN PRINT I,J
40 NEXT J
50 NEXT I
60 END
```

O programa rodará por meio do par de loops, mas imprimirá os valores de I e de J somente se I=3 e J=2. Ou seja, na tela aparecerá o seguinte resultado:

```
 2
```

A operação lógica OU pode ser usada de modo muito semelhante. Mudando a linha 30 para:

```
30 IF I=3 AND J=2 OR J=4 THEN PRINT I,J
```

será produzida a saída

```
 1 4
 2 4
 3 2
 3 4
 4 4
 5 4
```

O computador executa a operação E com prioridade em relação à operação OU: I e J serão impressos se I=3 e J=2 ou se J=4. A ordem de prioridade pode ser modificada pelo uso de parênteses. Verifique qual será o resultado do programa se a linha 30 for mudada para

```
30 IF I=3 AND (J=2 OR J=4) THEN PRINT I,J
```

Muitos micros usam registros especiais para controlar várias funções da máquina. Cada bit dentro de registros desse tipo pode controlar um aspecto diferente da operação. Por exemplo, no Commodore 64 há um registro de 8 bits que ativa e desativa os sprites. Cada bit no registro se relaciona com um dos oito sprites disponíveis. Se qualquer bit no registro for colocado em um (1), o sprite que ele controla ficará visível na tela. Se o bit for colocado em zero (0), o sprite será desativado.

Com a utilização do BASIC torna-se fácil ativar qualquer combinação de sprites, calculando o número binário exigido de 8 bits e armazenando pelo comando POKE seu equivalente decimal no registro. Esse método, porém, não leva em conta o estado do registro anterior ao comando POKE e pode resultar na desativação de sprites anteriormente ativados. A solução para esse problema está no desenvolvimento de uma técnica que permita ao programador isolar os bits a serem mudados sem alterar qualquer um dos outros.

Para demonstrar essa técnica, suponha que os sprites 0, 1, 5 e 6 estejam ativados. O registro que ativa terá a forma

Número do sprite	7	6	5	4	3	2	1	0
Bit correspondente	0	1	1	0	0	0	1	1

Agora ative o sprite 4, lendo o registro com o PEEK, operando com OR os conteúdos com 16 (00010000) e obtendo o resultado com o POKE.

Byte original	0	1	1	0	0	0	1	1
Byte operado com OU	0	0	0	1	0	0	0	0
Dá o novo byte	0	1	1	1	0	0	1	1

Com o comando BASIC POKE reg, PEEK (reg) OR 16, pode-se ativar o bit 4. Para desativá-lo, é preciso acessar o registro com PEEK e operar com AND seu conteúdo, usando o número 239.

Novo byte	0	1	1	1	0	0	1	1
E	1	1	1	0	1	1	1	1
Byte original	0	1	1	0	0	0	1	1

Observe que o número 239 é obtido subtraindo-se 16 de 255. Usando o comando BASIC POKE reg, PEEK (reg) AND 239, restaura-se o registro em seu estado original.

Essas técnicas são amplamente usadas na programação em código de máquina, na qual a alteração do estado dos registros de controle pode formar parte importante do programa.

CENAS ANIMADAS

O sprite é uma forma gráfica definida pelo usuário, que pode ser movida na tela à velocidade especificada pelo computador. A combinação de sprites permite compor cenas animadas.

Respostas do exercício anterior

1a) $A . (\overline{A} + \overline{B}) =$
$= A . \overline{A} + A . \overline{B}$ (lei distributiva)
$= A . \overline{B}$ ($A . \overline{A} = 0$)

b) $X + Y . (X + Y) + X . (\overline{X} + Y) =$
$= X + Y + X . (\overline{X} + Y)$ (relação 5)
$= X + Y + X . Y$ (relação 6)
$= X + Y$ (absorção)

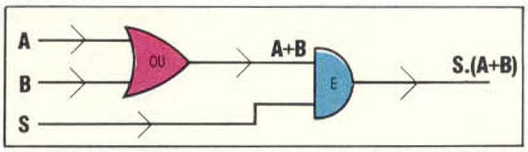
c) $P . Q + \overline{P} . Q + \overline{P} . \overline{Q} =$
$= P . Q + \overline{P} . (Q + \overline{Q})$ (lei distributiva)
$= P . Q + \overline{P}$ ($Q + \overline{Q} = 1$)
$= \overline{P} + Q$ (dual da relação 6)

d) $\overline{\overline{X} + \overline{Y} . Z + \overline{Z} . Y} =$
$= \overline{\overline{X}} . \overline{\overline{Y} . Z} . \overline{\overline{Z} . Y}$ (de Morgan)
$= X . Y . Z . (Z + \overline{Y})$ ($\overline{\overline{X}} = X$, de Morgan)
$= X . Y . Z . Z + X . Y . Z . \overline{Y}$ (lei distributiva)
$= X . Y . Z + 0$ ($Z . Z = Z$, $Y . \overline{Y} = 0$)
$= X . Y . Z$

2) A tabela de validação do sistema de alarma é:

ENTRADAS			SAÍDAS
A	B	S	Alarma
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Alarma $= \overline{A} . B . S + A . \overline{B} . S + A . B . S$
$= \overline{A} . B . S + A . S . (\overline{B} + B)$ (lei distributiva)
$= \overline{A} . B . S + A . S$ ($\overline{B} + B = 1$)
$= S . (A + \overline{A} . B)$ (lei distributiva)
$= S . (A + B)$ (dual da relação 6)



3) Se os três interruptores forem X, Y e Z e a luz for P, a tabela verdade será:

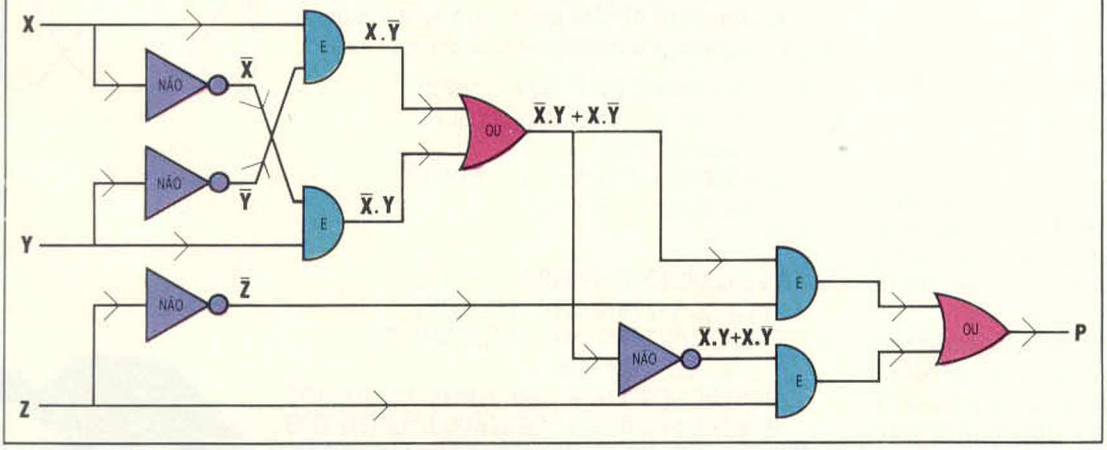
ENTRADAS			SAÍDAS
X	Y	Z	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$P = \overline{X} . \overline{Y} . Z + \overline{X} . Y . \overline{Z} + X . \overline{Y} . \overline{Z} + X . Y . Z$
$= Z . (\overline{X} . \overline{Y} + X . Y) + \overline{Z} . (\overline{X} . Y + X . \overline{Y})$ (lei distributiva)
$= Z . \overline{(\overline{X} . Y + X . \overline{Y})} + \overline{Z} . (\overline{X} . Y + X . \overline{Y})$ (de Morgan)

4) A pergunta "Você fala a verdade?" é pouco útil, pois tanto o mentiroso como o que fala a verdade darão a mesma resposta. A tabela de validação tem a mesma forma que a função $X . Y + \overline{X} . Y$, que pode ser simplificada para Y. Isto é, a resposta depende de uma variável, não de duas, o que não permite diferenciar os mentirosos dos sinceros. Mas, se fizermos a pergunta "Porcos têm asas?", a tabela será:

		RESPOSTAS POSSÍVEIS	
		SIM	NÃO
IDENTIDADE POSSÍVEL DO QUE RESPONDE	MENTIROSO	1	0
	SINCERO	0	1

e essa é a tabela de validação para a função $X . \overline{Y} + \overline{X} . Y$, que é também uma tabela para OU-exclusivo. Essa pergunta permite identificar aquele que responde.



CALCULANDO COM O LOGO

Se você precisa realizar muitos cálculos numéricos, dificilmente a linguagem LOGO será sua primeira opção. Apesar disso, ela dispõe de surpreendentes recursos para o processamento de números.

Quase todas as versões do LOGO possibilitam a realização de operações aritméticas seja com números reais (decimais), seja com números inteiros. Para isso são utilizados os operadores infixos: +, −, * e /. Esses operadores denominam-se "infixos" porque são escritos entre os números que relacionam — por exemplo, 3 + 4. Algumas versões do LOGO também operam com a notação por prefixos, segundo a qual o exemplo acima seria reescrito como SOMA 3 4. Essa notação por prefixos possui a vantagem de ser compatível com a forma na qual são escritos os outros comandos e operações do LOGO.

O MLOGO utiliza apenas a notação por infixos. Contudo, permite a programação de comandos com prefixos, caso esta seja necessária. Defina e teste as rotinas SOMA e PRODUTO:

```
AP SOMA :A :B
  SAIDA :A + :B
FIM

AP PRODUTO :A :B
  SAIDA :A * :B
FIM
```

A "precedência" das operações (a ordem na qual elas são realizadas) segue as regras usuais da matemática. Todas as operações indicadas entre parênteses são efetuadas em primeiro lugar, seguidas pelas multiplicações e divisões e, por fim, pelas adições e subtrações:

```
MOSTRE (3 + 4) * 5
MOSTRE 3 + 4 * 5
```

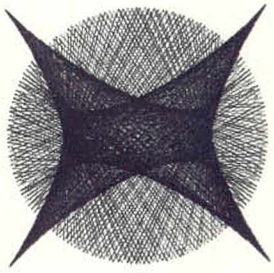
Experimente agora a notação por prefixos:

```
MOSTRE PRODUTO 5 SOMA 3 4
MOSTRE SOMA 3 PRODUTO 4 5
```

Isso demonstra uma outra vantagem da notação por prefixos — não há necessidade de regras de precedência e a linha é considerada do mesmo modo que qualquer outra linha de comandos do LOGO.

A operação normal de divisão (/) produz um número real como resultado. Para se trabalhar com números inteiros, duas outras operações — QUOC (QUOCIENTE) e RESTO — são muitas vezes de grande utilidade.

```
QUOC 47 5 E 9
RESTO 47 5 E 2
```

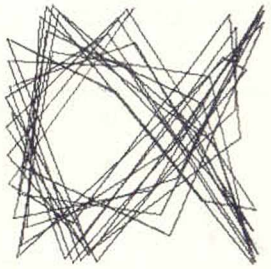


Um método padronizado para se converter em binário um número decimal consiste em dividi-lo sucessivamente por dois até que o resultado seja zero. O número binário é obtido ao se escrever, em ordem inversa, os restos encontrados em cada divisão. Veja, por exemplo, a conversão de 12 para o sistema binário:

```
12/2 = 6; RESTO = 0
6/2 = 3; RESTO = 0
3/2 = 1; RESTO = 1
1/2 = 0; RESTO = 1
```

Assim, lendo-se os restos de baixo para cima, obtemos o número binário equivalente ao decimal 12, ou seja, 1.100.

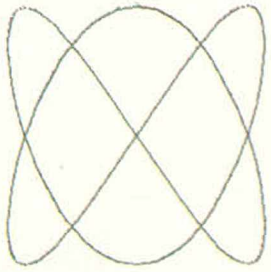
Os comandos QUOC e RESTO facilitam bastante a implementação desta técnica no LOGO. Se colocarmos a instrução MOSTRE *depois* da chamada recursiva, os restos serão apresentados na ordem correta (inversa).



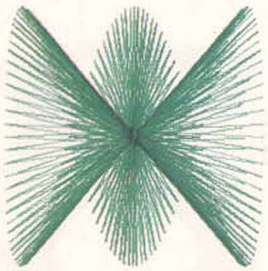
```
AP BINARIO :X
  SE :X = 0 ENTAO PARE
  BIN QUOC :X 2
  MOSTRAR RESTO :X 2
FIM
```

Existem duas operações para o arredondamento dos números — INT (INTEIRO) e APROX (APROXIMAR). INT apresenta como dado de saída a parte inteira de um número, ignorando todos os algarismos depois da vírgula. APROX aproxima o número para o inteiro mais próximo, seja este maior ou menor.

As rotinas apresentadas a seguir calculam os juros compostos sobre um investimento de acordo com determinada taxa. Em VER.LUCRO, INT apresenta como dado de saída o valor em cruzeiros, enquanto APROX serve para aproximar os centavos para o número inteiro mais próximo.



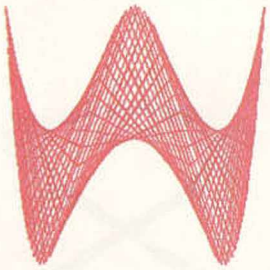
```
AP JURO :PRINCIPAL :TAXA :ANOS
  SE :ANOS = 0 ENTAO VER.LUCRO
  :PRINCIPAL PARE
  JURO :PRINCIPAL * (1 + :TAXA/100)
  :TAXA :ANOS − 1
FIM

AP VER.LUCRO :DINHEIRO
  FACA "CRUZEIROS INT :DINHEIRO
  FACA "CENTAVOS APROX (:DINHEIRO −
    :CRUZEIROS) * 100
  MOSTRE SENTENCA :CRUZEIROS "CRUZEIROS
  MOSTRE SENTENCA :CENTAVOS "CENTAVOS
FIM
```



Testes lógicos

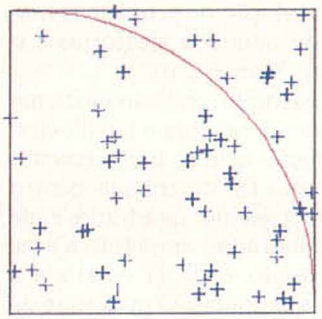
Já empregamos =, < e > como testes lógicos em várias rotinas. As operações lógicas SETODOS, SEUM e NAO podem ainda ser combinadas de modo a constituir outros testes. SETODOS é verdadeiro apenas quando ambos os seus dados de entrada são verdadeiros. SEUM é verdadeiro quando um de seus dados de entrada for verdadeiro, enquanto NAO só é verdadeiro quando seu dado de entrada for falso. Desse modo obtemos:

```
SE SEUM :X>0 :X = 0 ENTAO MOSTRE "POSITIVO
SE NAO :X<0 ENTAO MOSTRE "POSITIVO
SE SETODOS :X>0 :X<100 ENTAO MOSTRE
  [ENTRE 0 E 100]
```



A operação NUMERO? tem como resultado VERD (VERDADEIRO) se o dado introduzido for um número. Caso contrário, o resultado será FALSO. Esta operação foi implementada na rotina PRIMO?, que apresenta como resultado VERD se o dado de entrada for um número primo, ou FALSO em caso contrário. Ela começa pela confirmação de que o dado introduzido é realmente um número, e de que este é maior do que dois. PRIMO.TESTE verifica a seguir se algum inteiro entre a raiz quadrada do número e o número dois pode dividi-lo sem deixar resto.

```
AP PRIMO? :NUM
  SE NAO NUMERO? :NUM ENTAO MOSTRE [ISTO
    NAO E UM NUMERO] PARE
  SE :NUM<2 ENTAO SAIDA "FALSO
  SAIDA PRIMO.TESTE :NUM INT RQD :NUM
FIM
```



MÉTODO DE MONTE CARLO

```
AP PRIMO.TESTE :NUM :FATO
  SE :FATO = 1 ENTAO SAIDA "VERD
  SE (RESTO :NUM :FATO) = 0 ENTAO SAIDA
    "FALSO
  SAIDA PRIMO.TESTE :NUM :FATO - 1
FIM
```

Números aleatórios

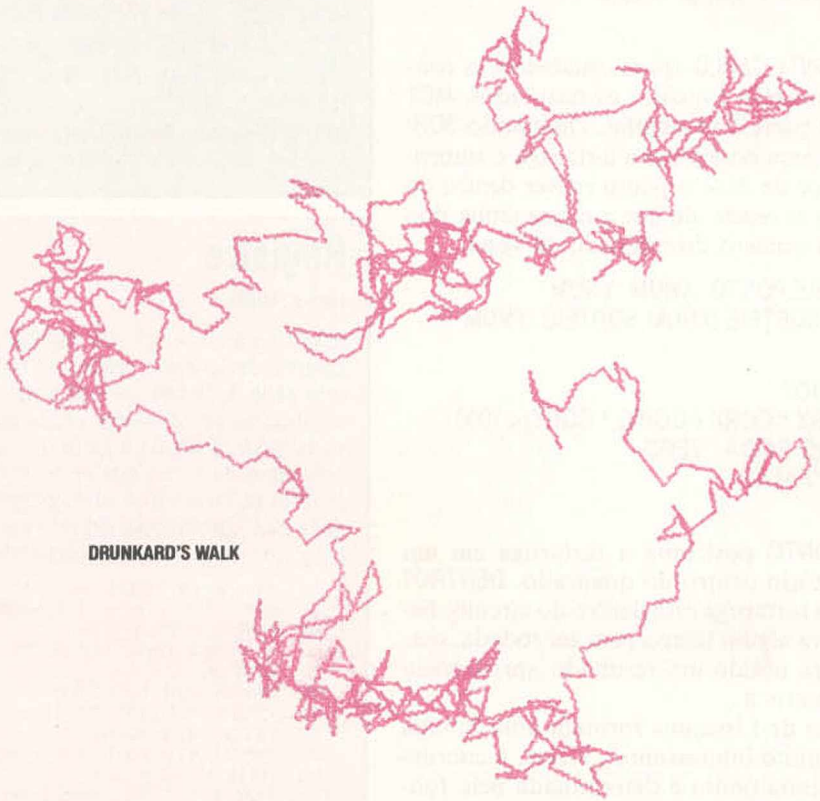
O resultado de SORTEIE n é um número inteiro aleatório entre 0 e n-1. A rotina BEBADO faz com que a tartaruga atravesse a tela cambaleando. A cada passo que dá, ela faz um giro de ângulo aleatório. O dado de entrada A determina o maior ângulo possível de giro. Se você executar essa rotina, verá que a tartaruga se move em círculos imprecisos, girando para a direita ou para a esquerda conforme o valor atribuído a A.

```
AP BEBADO :A
  FR 1
  DI (-:A/2+SORTEIE :A)
  BEBADO :A
FIM
```

Um passo fora da linha

Segundo o teorema Passeio de Bêbado, após n passos em direções perfeitamente aleatórias, supera 0,5 a probabilidade de que a distância percorrida pelo bêbado desde o ponto inicial seja menor do que a raiz quadrada de n. Esta previsão estatística baseia-se num número muito alto de passos. Você pode verificá-la com o LOGO:

```
AP PASSEIOBEBADO :PASSONUM
  :PASSO
  LT REPITA :PASSONUM [DI
    (SORTEIE-361) FR :PASSO]
FIM
```



DRUNKARD'S WALK

Uma técnica para a resolução de problemas matemáticos por meio de números aleatórios é o chamado "método de Monte Carlo".

Em seguida, como exemplo, realizaremos uma aproximação ao número pi (π) com o auxílio desse método. Na ilustração acima, um segmento de um quarto de círculo foi desenhado dentro de um quadrado. A área do quadrado é de 100 x 100 unidades quadradas, enquanto a área de um quarto de círculo é de (1 ÷ 4) x π x x 100 x 100 unidades quadradas. O quociente da divisão da área do círculo pela do quadrado é igual a π ÷ 4. Agora, lance ao acaso um alfinete sobre o quadrado, repita isto 1.000 vezes e conte quantas vezes o alfinete cai dentro da área do quarto de círculo. Esse número será chamado de IN. O valor de IN/1.000 deve ser aproximadamente o mesmo que o resultado de: círculo ÷ quadrado — isto é, π ÷ 4. Assim, se você fizer essa experiência, multiplicar IN por 4 e dividir o resultado por 1.000, terá uma aproximação de π. Isso é o que as rotinas seguintes fazem:

```
AP MONTE.CARLO
  DESENHE
  SEMCOR
  FACA "IN 0
  MC1 1000 100 100
  MOSTRE SENTENCA [VALOR
    DE π E] 0.004 * :IN
FIM

AP MC1 :NUM :XNUM :YNUM
  SE :NUM = 0 ENTAO PARE
  SORTEIE.PONTO :XNUM YNUM
  SE DENTRO? ENTAO FACA "IN :IN + 1
  MC1 :NUM – 1 :XNUM :YNUM
FIM
```

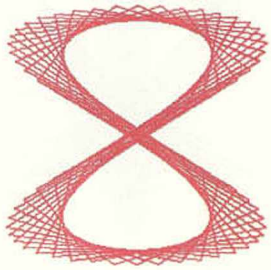
A rotina MONTE.CARLO apenas estabelece as condições, chama MC1 e mostra os resultados. MC1 faz a maior parte do trabalho, chamando SORTEIE.PONTO para posicionar a tartaruga e aumentando o valor de IN se o ponto estiver dentro do círculo. Isso se repete até que a rotina tenha sido executada o número determinado de vezes.

```
AP SORTEIE.PONTO :XNUM :YNUM
  DEFXY SORTEIE :XNUM SORTEIE :YNUM
FIM

AP DENTRO?
  SE (CORX * CORX + CORY) * CORY) < 1000
    ENTAO SAIDA "VERD
  SAIDA "FALSO
FIM
```

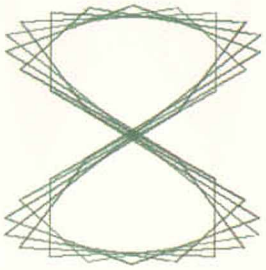
SORTEIE.PONTO posiciona a tartaruga em um ponto aleatório dentro do quadrado. DENTRO? verifica se a tartaruga está dentro do círculo. Essa rotina leva algum tempo para ser rodada, mas no final será obtido um resultado aproximado para o número π.

As curvas de Lissajous formam uma família de curvas muito interessantes. Nelas, a coordenada x de cada ponto é determinada pela função seno, e a coordenada y pela função co-seno:

```
AP LJ :COEF1 :COEF2 :PASSO
  DESENHE SEMCOR SEMT
  POS :COEF1 :COEF2 0 COLORIDO
  LJ1 :COEF1 :COEF2 0 :PASSO
FIM

AP POS :COEF1 :COEF2 :ANG
  FACA "X100 * SEN (:COEF1 * :ANG)
  FACA "Y100 * COS (:COEF2 * :ANG)
  DEFXY :X :Y
FIM

AP LJ1 :COEF1 :COEF2 :ANG :PASSO
  POS :COEF1 :COEF2 :ANG
  LJ1 :COEF1 :COEF2 (:ANG + :PASSO) :PASSO
FIM
```



Por tudo isso, você já deve ter se convencido do extraordinário potencial do LOGO para o tratamento de números. Por meio da construção de novos comandos a partir dos primitivos numéricos, podemos solucionar praticamente qualquer problema matemático com um mínimo de esforço. Outra vantagem apresentada pelo LOGO é que você pode visualizar o resultado, graças à tartaruga, além do resultado numérico normal que se obtém com o emprego das outras linguagens.



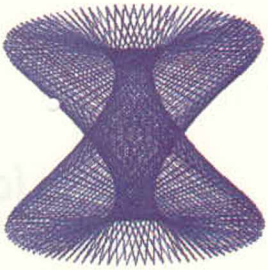
Exercício 8

A) Escreva uma rotina para calcular a enésima potência de um número, de modo que POTENCIA 42 tenha como resultado 16.

B) Elabore um conjunto de rotinas para converter em hexadecimal um número decimal (empregue a mesma técnica usada no exemplo dos números binários, mas desta vez divida por 16).

C) Escreva uma rotina PAR? que tenha como resultado VERD se o número for par, e FALSO se for ímpar.

D) Use o método Monte Carlo para determinar a área sob a curva $y = x^2$ entre $x = 0$ e $x = 10$.

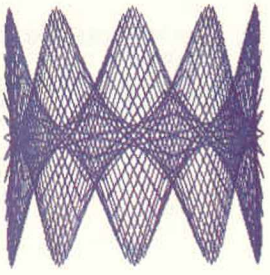


Registre

Linha Apple

Este programinha em BASIC também produz um surpreendente efeito gráfico na tela de seu micro: uma série de linhas começa a varrer a tela, partindo do canto inferior esquerdo, e em seguida outro feixe se irradia a partir do canto direito. Há sete cores de linhas que se sucedem aleatoriamente. Veja as linhas 30 e 40 do programa: alterando a linha 30 e colocando-a em diferentes lugares do programa, você obterá belos padrões coloridos.

```
  5  REM **** FEIXE DE LINHAS ***
 10  HGR : HCOLOR = 3 : HOME
 20  FOR N = 0 TO 159
 30  W = INT (RND (1) * 7)
 40  HCOLOR = W
 50  HPLOT 0,N TO 159,N
 60  HPLOT 0,N TO 159,159 - N
 70  HPLOT N,0 TO N,159
 80  HPLOT N,0 TO N,159 - N
 90  NEXT N
100 FOR I = 1 TO 3000 : NEXT I
```



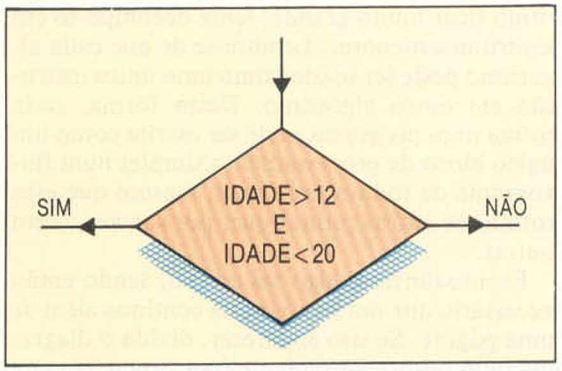
DECISÕES CRUCIAIS

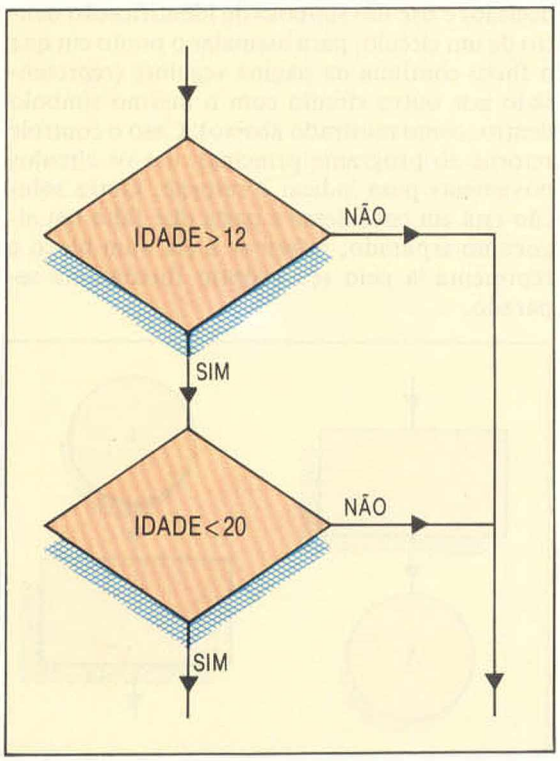
Na utilização de fluxogramas para o desenvolvimento de programas, as decisões compostas podem dividir-se em componentes simples. Nos casos mais complexos, usam-se as tabelas de decisão.

Os programadores precisam usar às vezes decisões compostas em seus programas, como:

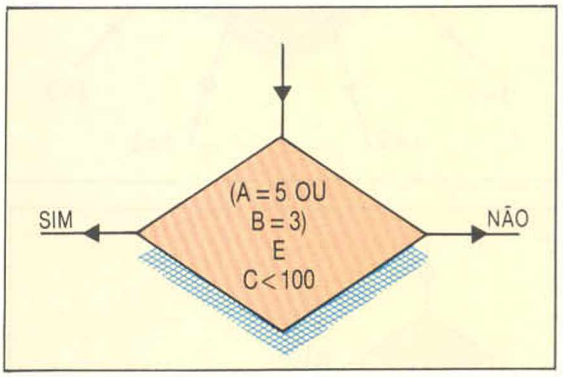
```
SE IDADE>12 E IDADE<20 ENTAO FAIXA =
  "ADOLESCENTE"
```

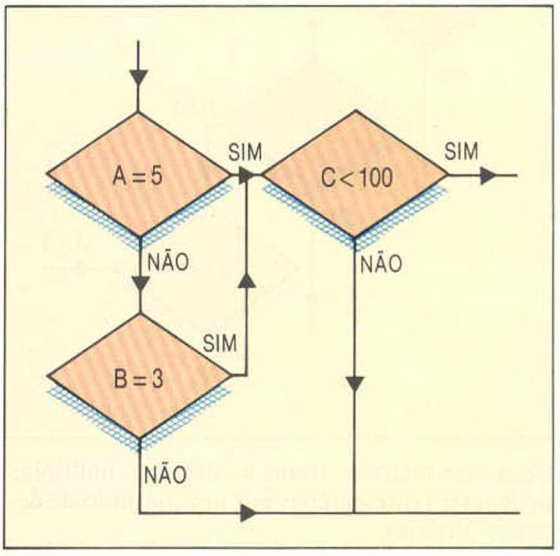
Os algoritmos com instruções como essa serão mais fáceis de entender se a decisão composta for subdividida nas decisões que a compõem.





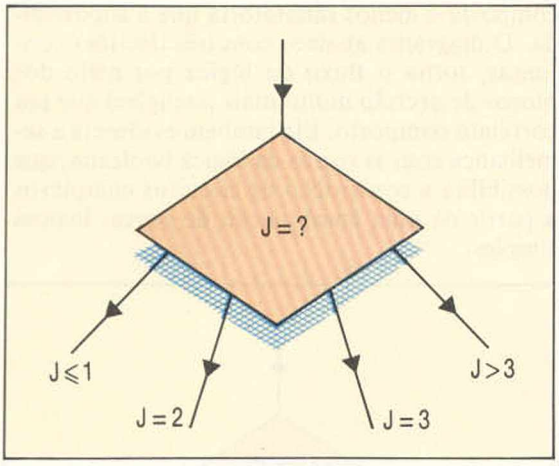
Apresentamos o exemplo em BASIC sob a forma de um diagrama, para mostrar como a versão composta é menos satisfatória que a subdividida. O diagrama abaixo, com três decisões conjuntas, torna o fluxo da lógica por meio dos blocos de decisão muito mais inteligível que seu correlato composto. Ele também evidencia a semelhança com as regras da lógica booleana, que possibilita a construção de circuitos complexos a partir de uma combinação de portas lógicas simples.

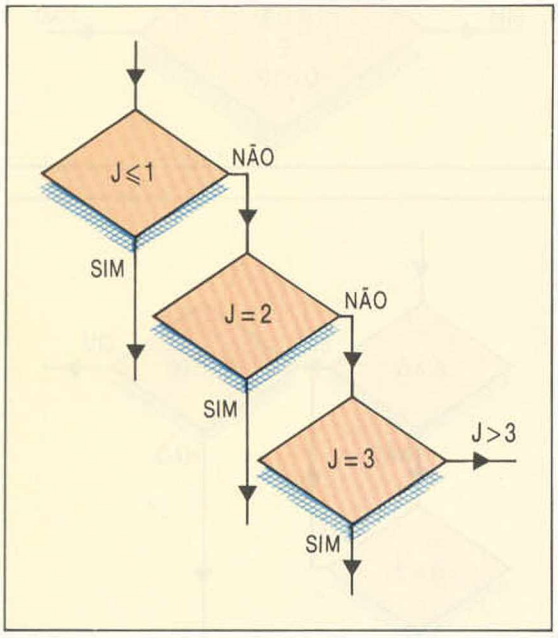




Nos exemplos apresentados, todas as decisões são binárias, mas é bastante comum um algoritmo envolver decisões com mais de dois resultados possíveis. Se o programa aceitar uma entrada pelo teclado que signifique uma escolha num menu, será necessário desviar para uma sub-rotina. A maioria das linguagens de programação fornece estruturas de desvio como ON-GOTO e ON-GOSUB, em BASIC, e CASE-OF, em PASCAL. As regras para as decisões binárias também se aplicam às decisões múltiplas: pode-se escolher somente uma rota a partir de uma decisão, e todas

as saídas devem ser bem definidas, com as rotas possíveis sendo mutuamente exclusivas e abrangendo todas as possibilidades. É possível representar uma decisão múltipla, como no exemplo mostrado, com um conjunto de rotas de saída provindo da mesma decisão. No entanto, isso não é muito comum e, mais freqüentemente, a decisão terá de se decompor em várias decisões binárias, como no exemplo.





Com esse método, todas as decisões múltiplas podem ser representadas por um conjunto de decisões binárias.

A tabela de decisão

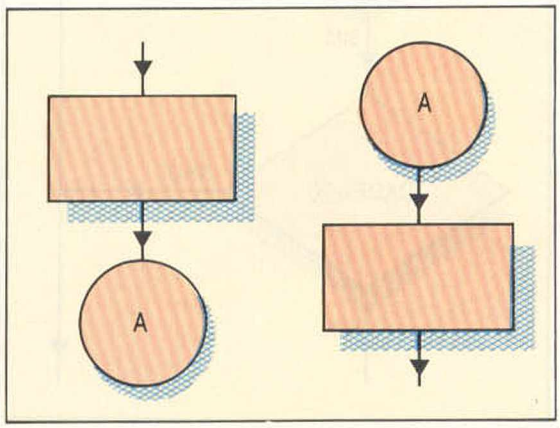
Como alternativa aos fluxogramas, especialmente quando são possíveis muitas decisões múltiplas, recomenda-se usar tabelas de decisão. Esse tipo de tabela apresenta quatro partes principais: as condições para as regras; as ações a ser realizadas; um quadriculado, que mostra como as condições se encaixam nas regras; e mais um quadriculado, com as ações apropriadas para cada regra. No diagrama "condições/regras", os valores das variáveis aparecem nas células; já no diagrama "ações/regras", abaixo do primeiro,

	REGRAS							
CONDIÇÕES	1	2	3	4	5	6	7	8
BOTÃO DETONADOR ACIONADO	✓	✗	✗	✗	✓	✗	✓	✗
NÍVEL DO JOGO	1	1	2	2	1	1	2	2
NÍVEL DO JOGADOR	NOVATO	NOVATO	NOVATO	NOVATO	AVANÇADO	AVANÇADO	AVANÇADO	AVANÇADO
AÇÕES								
RETIRADA DOS ALIENÍGENAS							✓	✓
DETONAÇÃO ALEATÓRIA			✓	✓			✓	✓
REDUZIR O NÍVEL DE ENERGIA EM	1%	1%	2%	2%	2%	2%	4%	4%

uma marca indica que ação deve ser executada, e um valor age como seu parâmetro de entrada. A regra 4, por exemplo, é a seguinte: "Se o botão de disparo for pressionado, o nível do jogo for 2 e o jogador for novato, acione os detonadores aleatoriamente e reduza o nível de energia em 2%".

Sempre que possível, limite os fluxogramas a uma página. Pode ser irritante e demorado folhear várias páginas de papel. Quando seu algoritmo ficar muito grande, tente decompô-lo em algoritmos menores. Lembre-se de que cada algoritmo pode ser usado como uma única instrução em outro algoritmo. Dessa forma, cada rotina num programa pode ser escrita como um único bloco de processamento simples num fluxograma de todo o programa, mesmo que essa rotina use outras rotinas que, por sua vez, usem outras.

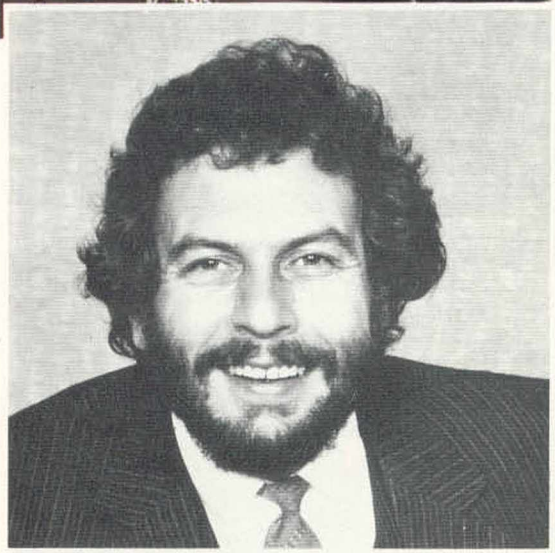
Eventualmente, algo sai errado, sendo então necessário que um fluxograma continue além de uma página. Se isso acontecer, divida o diagrama num ponto conveniente (por exemplo, uma decisão) e use um símbolo de identificação dentro de um círculo, para assinalar o ponto em que o fluxo continua na página seguinte (representado por outro círculo com o mesmo símbolo dentro, como mostrado abaixo). Caso o controle retorne ao programa principal, use os círculos novamente para indicar o retorno. Outra solução está em considerar a parte que falta um algoritmo separado, referir-se a ela num bloco e representá-la pelo seu próprio fluxograma separado.



O INÍCIO DO JOGO

Space Invaders
O mais conhecido dos jogos eletrônicos é um dos descendentes do Pong, criado por Nolan Bushnell.

Quando conectou um microprocessador a seu aparelho de televisão e inventou o jogo Pong (que, como o nome sugere, se assemelhava ao pingue-pongue), Bushnell deu início à história da Atari.



O método de Nolan Bushnell para colocar sob o controle do usuário as imagens que aparecem na tela mudou o conceito popular de lazer e conquistou milhões de jovens.

Ele e dois parceiros (Ted Dabney e Larry Bryan) investiram algum dinheiro e lançaram o jogo eletrônico Pong em Sunnyvale, Califórnia, em 1972. Logo perceberam que tinham em mãos um sucesso e uma fonte de lucros.

O domínio da Atari sobre o mercado de videogames começou com uma decisão oportuna tomada por essa empresa logo depois: comprar os direitos da invenção de Bushnell.

A Atari manteve a dianteira por quase toda a década de 70, até o gosto do público se transferir dos jogos do tipo fliperama para os micros. O mercado de jogos é como o de discos: devem-se descobrir as estrelas em potencial e promovê-las. Talvez por isso a Atari tenha sido comprada pela Warner Communications International, ligada a filmes e discos.

Console para jogos
O Sistema de Vídeo Computador (VCS, Video Computer System) da Atari é fornecido com dois joysticks, adaptador de cabos e um cartucho. Destinado exclusivamente a jogos, o VCS não pode ser usado como computador para aplicações gerais. Todos os jogos vêm em cartuchos.

Os fliperamas operados por moedas ou fichas proporcionaram lucros enormes até os últimos anos da década de 70. Mas logo o negócio entrou em declínio e, em meados da década seguinte, causou prejuízos consideráveis.

Space Invaders, da Taito, habilmente comercializado pela Atari, é o mais conhecido de todos os jogos para computador. Tornou-se um fenômeno social e originou uma enxurrada de outros cartuchos com temas galácticos. A empresa esteve no centro da febre dos jogos eletrônicos, com vários de seus produtos nas listas dos preferidos: Asteroids, Battlezone, Centipede, Missile Comand e The Tempest, por exemplo.

A obsessão pelos fliperamas foi meteórica. Os freqüentadores logo se voltaram para os microcomputadores, que ofereciam duas grandes vantagens: o mesmo divertimento das casas de jogos eletrônicos, sem dispêndio de dinheiro. Além disso, o usuário passava a possuir uma máquina de computação bastante flexível.

O grande declínio
Durante a maior parte da década de 70, a Atari prosperou com os lucros gerados por máquinas de fliperama como a Major Havoc. O advento do microcomputador, contudo, exigiu uma nova estratégia de mercado.

Inicialmente, a Atari respondeu a essa mudança na preferência dos consumidores convertendo seus melhores softwares para fliperama em jogos destinados a micros. Conectava-se um cartucho ao micro, expandindo ou substituindo a ROM do próprio computador. Assim, não era necessário carregar o jogo na memória a partir de um cassete ou disquete. A tecnologia necessária, porém, tornava os cartuchos muito caros e não reprogramáveis. Em conseqüência, a empresa se via freqüentemente com pilhas de encalhes dos jogos impopulares.

Declínio da fortuna

A estratégia de mercado utilizada pela Atari para superar o problema não deu bons resultados. Baseando as projeções de venda num jogo que alcançara enorme sucesso — o PacMan —, ela pagou um alto preço por esse otimismo exagerado: catorze caminhões despejaram os cartuchos não vendáveis num grande buraco no deserto de Nevada.

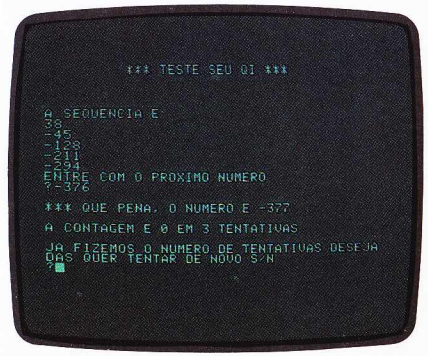
A Atari também falhou por não investir numa característica operacional fundamental dos computadores: as instruções de um programa não precisam converter-se em entidade física para serem distribuídas. Sua transmissão pode ser feita por telefone, cabo, rádio ou televisão. Novos produtos e técnicas que permitem essas formas de transmissão tornam-se cada vez mais disponíveis no mercado internacional.

Os problemas da Atari foram causados, em parte, pelo desentendimento entre a divisão de videogames, que estava em declínio, e a crescente divisão de microcomputadores. Agora, as duas se integraram.

Os microcomputadores da empresa têm como característica três chips especiais (Pokey, Antic e GTIA), que controlam, respectivamente, as portas de entrada e saída, os gráficos e a cor. Todos utilizam o processador 6502 e há grande variedade de programas à disposição.

Surpresas lógicas
Se você acha que inferir o número seguinte de uma série logicamente estruturada é muito simples, vai se surpreender com este jogo.

```
160 IF Z <.7 THEN LET K = - K
170 LET P = INT (RND*(5)+1)
180 LET B = RND * (1)
190 REM *** 50  % DE POSSIBILIDADE DE P
    SER NEGATIVO
200 IF B > .5 THEN LET P = - P
210 REM *** 70  % DE POSSIBILIDADE DE P
    SER IGUAL A 1
220 IF B < .7 THEN LET P = 1
230 REM *** 30  % DE POSSIBILIDADE DE P
    SER IGUAL A 1/P
240 LET Q = INT(RND*(3)) + 1
250 LET E = RND * (1)
260 REM *** 40  % DE POSSIBILIDADE DE Q
    SER IGUAL A 1/Q
270 IF E > .6 AND Q LET Q = 1/Q
280 REM *** 90  % DE POSSIBILIDADE DE Q
    SER IGUAL A ZERO
290 IF E < .9 THEN LET Q = 0
300 LET N = INT (RND*(39))+1
310 FOR I = 1 TO 5
320 PRINT N
330 LET N = N * P + K + (N*Q)
340 NEXT I
350 PRINT "ENTRE COM O PROXIMO NUMERO"
360 INPUT M
370 PRINT
380 IF M = N THEN PRINT "***VOCE
    ACERTOU"***
390 IF M = N THEN GOTO 420
400 PRINT "SEU NUMERO ERA: ";M
410 PRINT "O NUMERO CERTO E: ";N
420 PRINT
430 LET S = S + (M=N)
440 PRINT "A CONTAGEM E ";S;" EM ";T;
    " TENTATIVAS"
450 PRINT
460 FOR N = 1 TO 100
470 NEXT N
480 IF T = V THEN GOTO 500
490 GOTO 460
500 PRINT TAB (35);"*** TESTE SEU QI **
510 PRINT
520 PRINT
```

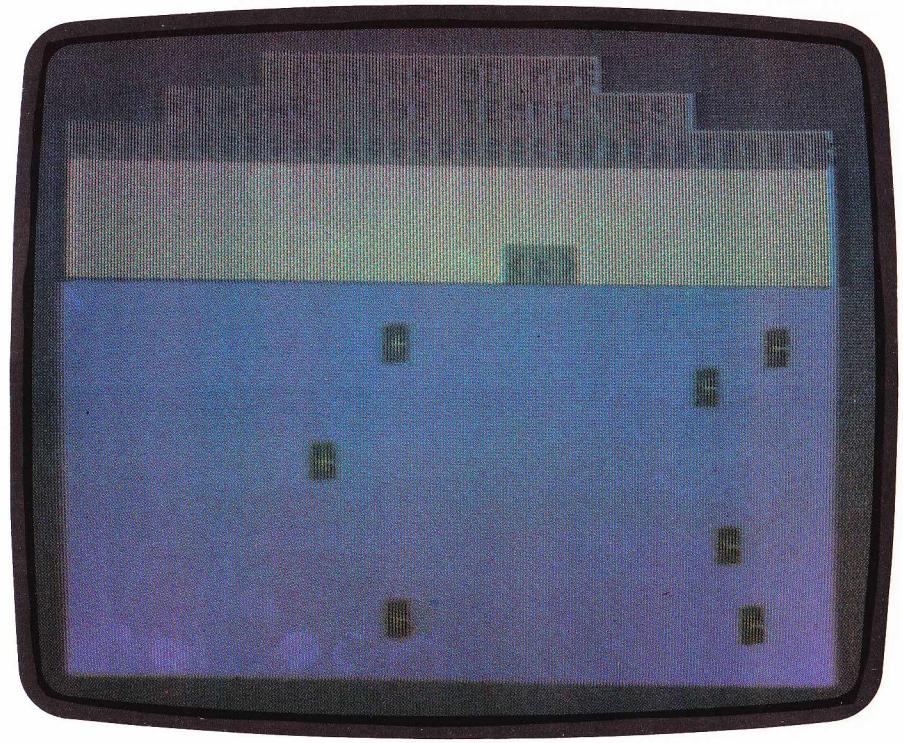



```
530 PRINT "QUANTAS TENTATIVAS VOCE
    QUER ";

540 INPUT V
550 GOTO 30
560 PRINT "JA FIZEMOS O NUMERO DE
    TENTATIVAS DESEJADAS";
570 IF S = T THEN PRINT " MEUS PARABENS"
580 PRINT " QUER TENTAR DE NOVO S/N"

590 INPUT V$
600 IF V$="S" THEN GOTO 1
610 PRINT
620 PRINT "TCHAU"
```


BATALHA NO MAR

PARA CP400 COLOR

Você está num navio, cercado por dez submarinos inimigos. Sua missão é torpedeá-los no menor tempo possível, porque outros virão em seguida e, quanto mais submarinos forem destruídos, mais pontos você terá. Cada submarino afundado vale dez pontos, mas, se ele passar pelo navio e atingir o canto superior esquerdo da água, reaparecerá no fundo do mar e você perderá cinco pontos. Na tela, um contador mostra o tempo de jogo (de 0 a 500). Quando a carga está a caminho, o relógio pára e qualquer tecla usada nesse intervalo será executada ao final da trajetória. Esgotado o tempo, aparece a contagem final. Tecle S se quiser jogar novamente.



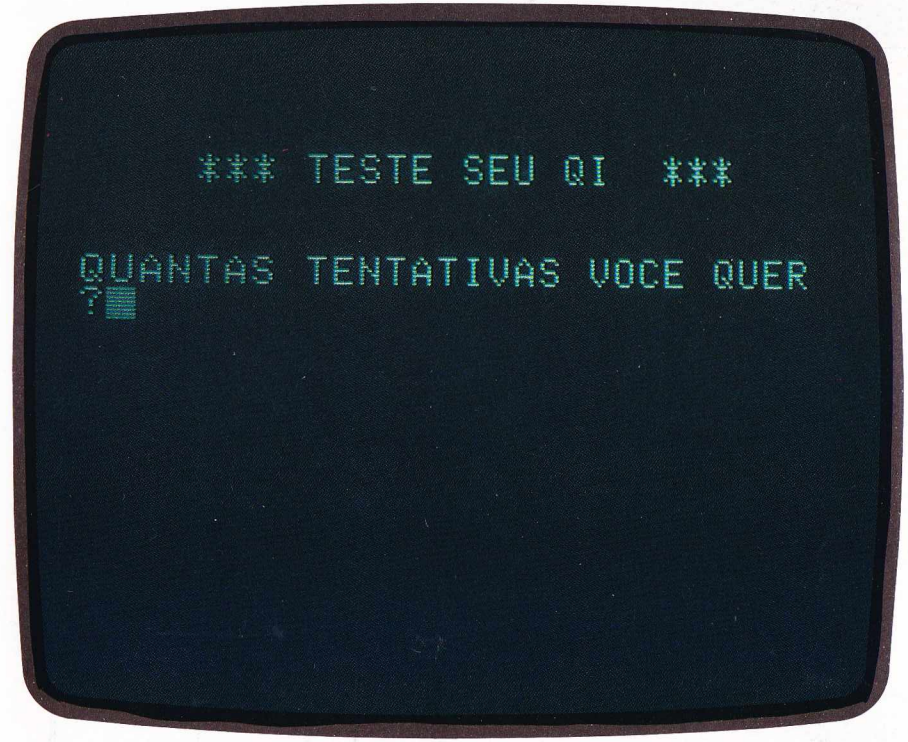
```
100 REM*********************************
105 REM
110 REM     BATALHA NO MAR
115 REM    PARA CP-400 COLOR
120 REM       16K OU 64K
125 REM
130 REM*********************************
135 REM
140 REM*********************************
145 REM     INICIALIZACAO
150 REM*********************************
155 REM
160 PMODE0,1: PCLEAR1: CLEAR300
165 K=176: OO=501: RN=RND(-TIMER)
170 DIM P(9)
175 REM
180 REM*********************************
185 REM  PREPARA A TELA DO JOGO
190 REM*********************************
195 REM
200 CLS3: PRINT STRING$(9,144)
205 PRINT @ 8,"BATALHA NO MAR";
```

```
210 PRINT STRING$(42,144);
215 PRINT STRING$(32,"#");
220 PRINT STRING$(96,159);
225 PRINT @ 160,STRING$(32,159);:
    PRINT @ K,"<X>";
230 REM
235 REM*******************************
240 REM   ROTINA PRINCIPAL
245 REM*******************************
250 REM
255 FOR X=0 TO 9
260 P(X)= RND(288) + 221
265 IF X=0 THEN 275
270 FOR Q=0 TO X-1: IF P(X)=>P(Q)+2 OR
    P(X)=< P(Q)-2 THEN NEXT Q ELSE 260
275 PRINT @ P(X), CHR$(127);
280 NEXTX
285 FOR X=0 TO 9
290 IF P(X)=0 THEN 315
295 P(X)=P(X)-1
300 IF P(X) < 192 THEN 325
305 PRINT @ P(X)+1, CHR$(175);
310 PRINT @ P(X), CHR$(127);
315 NEXT X
320 IF A$<>"F" THEN 330 ELSE RETURN
325 PRINT @ P(X)+1, CHR$(175);:
    XX=XX-5: P(X)=507: GOTO 305
330 A$=INKEY$
335 QQ=QQ-1: PRINT @ 36,"PLACAR "XX,
    "TEMPO "QQ;
340 IF QQ=0 THEN 580
345 REM
350 REM*******************************
355 REM     VERIFICA TECLADO
360 REM*******************************
365 REM
370 IF A$="G" THEN K=K-2
375 IF A$="H" THEN K=K+2
380 IF A$="F" THEN GOSUB 410
385 IF K<160 THEN K=160
390 IF K>189 THEN K=189
395 PRINT @ 160, STRING$(32,159);:
    PRINT @ K,"<X>";
400 GOSUB 285
405 GOTO 330
410 II=K
415 FOR I=0 TO 9
420 II=II+32
425 IF II>511 THEN GOTO  500
430 GOSUB 285
435 REM
440 REM*******************************
445 REM   VERIFICA SE ACERTOU
450 REM*******************************
455 REM
460 PRINT @ II,"I";
465 FOR T=0 TO 9
470 IF P(T)=II THEN 475 ELSE 485
475 P(T)=0: XX=XX+10: Z=Z+1: PRINT @ II,
    CHR$(175);: SOUND 10,1
480 IF Z=10 THEN 530 ELSE 500
485 NEXT T
490 PRINT @ II,CHR$(175);
495 NEXT I
500 A$="": GOTO 285
505 REM
510 REM*******************************
515 REM  PREPARA NOVA ETAPA
520 REM*******************************
525 REM
530 Z=0
535 CLS 0: PRINT @ 224,
    " TODOS OS SUBMARINOS DESTRUIDOS"
540 PRINT @ 6,"PLACAR "XX "TEMPO "QQ;
545 FOR TM=1TO10: PLAY"T30CDEFGAB":
    NEXT TM: FORZZ=0TO200: NEXT ZZ:
    GOTO200
550 REM
555 REM*******************************
560 REM  AVALIA RESULTADOS,
565 REM  RETORNA OU FINALIZA
570 REM*******************************
575 REM
580 PRINT @ 71,"TERMINOU O TEMPO";
585 PRINT @ 96,"      RESULTADO >> ";
590 IF XX<100 THEN PRINT"FRACO":GOTO 615
595 IF XX<200 THEN PRINT"MEDIO":GOTO 615
600 IF XX<300 THEN PRINT"LEGAL":GOTO 615
605 IF XX<400 THEN PRINT"OTIMO":GOTO 615
610 PRINT"MELHOR IMPOSSIVEL"
615 PRINT@486,"JOGA NOVAMENTE  (S/N)";:
    RE$=""
620 RE$=INKEY$:IF RE$="" THEN 620
625 IF RE$="N" THEN CLS:END ELSE
    IF RE$="S" THEN GOTO 110 ELSE 615
630 REM
635 REM*******************************
640 REM  FINAL DA LISTAGEM
645 REM*******************************
```


TESTE SEU Q.I.

PARA A LINHA SINCLAIR

Este programa apresenta seqüências lógicas de números, ora muito simples, como as progressões aritméticas, ora mais difíceis, como as geométricas ou as séries de razão variável. Seu desafio é encontrar o número seguinte da seqüência. Cada série é formada a partir do primeiro número, gerado de modo aleatório, juntamente com os fatores P, K e Q que a determinam. Na execução do programa, você decidirá quantos números da seqüência serão exibidos na tela e o total de tentativas permitidas. Ao término das possibilidades, o programa mostrará o número certo ou exibirá na tela o número de tentativas feitas, caso você acerte antes de esgotá-las.




```
1   REM *****PARA A LINHA SINCLAIR*****
2   REM ******   TESTE SEU QI   ******
3   REM *********************************
10  CLS
20  GOTO 500
30  LET S = 0
40  LET T = S
50  LET W = 0
60  CLS
70  LET W = W +1
80  PRINT "TENTATIVA NUMERO ";W
90  PRINT "A SEQUENCIA E': "
100 LET T = T + 1
110 LET K = INT (RND*(100))
120 LET Z = RND * (1)
130 REM *** 50  % DE POSSIBILIDADE DE K
    SER IGUAL A ZERO ***
140 IF Z > .5 THEN LET K = 0
150 REM *** 70  % DE POSSIBILIDADE DE K
    SER NEGATIVO ***
```