COLOR BASIC

COLOR BASIC

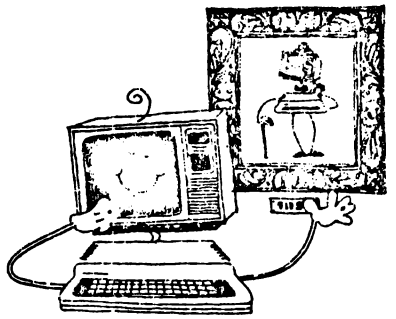
EXTENDIDO

COLOR BASIC

EXTENDIDO

INSTRUÇÃO PROGRAMADA

INTRODUÇÃO



Bem-vindo! Estamos de volta à família do COLOR.

Como você pode ver, nos crescemos e queremos ser os primeiros a apresentá-lo ao ... filho do COLOR BASIC. Embora seu nome seja COLOR BASIC EXTENDIDO, nós estamos certos que você vai achá-lo um ótimo complemento para suas atividades computacionais, não importa qual seja o seu nome.

O COLOR BASIC EXTENDIDO tem todas as características do COLOR BASIC, além de uma série de vantagens extras.

Por exemplo, com o COLOR BASIC EXTENDIDO, você pode:

- *Desenhar um círculo;*
- *Pintar uma casa;*

. *Tocar uma sinfonia;*
. *Criar um cubo;*
. *Editar uma linha;*
. *Elevar um número ao quadrado;*
. *E até mesmo tentar fazer um triângulo.*

Você já deve conhecer o Manual do COLOR BASIC e o Manual de Operação, portanto vamos tentar não repetir muito as informações desses livros.

Mostraremos, entretanto, a capacidade do COLOR BASIC EXTENDIDO , sugerindo meios que o ajudarão a usar e se divertir com o computador.

Para facilitar seu entendimento, dividimos este manual em três partes:

Seção I - "SOM E IMAGEM"
Ensina, entre outras coisas, como desenhar linhas e círculos, tocar músicas e traçar desenhos tridimensionais.

Seção II - "DE VOLTA AO BASIC"
Introduz as características do COLOR BASIC EXTENDIDO . Envolverá trabalhos com números e letras, em vez de formas e cores. Aqui, você aprenderá a usar o computador

para editar linhas, elevar números ao quadrado, e criar até mesmo suas próprias funções para o COLOR BASIC.

APÊNDICES - contendo:

- *Modelos de programas*
- *Soluções para exercícios que serão apresentados*
- *Tabelas úteis para se ter à mão*
- *Gabaritos de vídeo para facilitar seu trabalho com gráficos*
- *E muito mais!*

Nos apêndices incluímos uma seção para referência técnica sobre o COLOR BASIC e o COLOR BASIC EXTENDIDO.

Bem, então agora sente-se para conhecer seu novo amigo. Temos certeza que você concordará que ele é um membro bem-vindo, não apenas à nossa família de computadores, mas também à sua.

CONTEÚDO

APÊNDICES

SEÇÃO I

SOM E IMAGEM

CAPÍTULO I

VAMOS AO QUE INTERESSA

1

VAMOS AO QUE INTERESSA

( ) Seu computador está ligado e pronto para ser usado?
( ) Você já leu o Manual do COLOR BASIC?

Se respondeu sim às duas perguntas, você está pronto para começar.

Uma das características mais interessantes do COLOR BASIC EXTENDIDO é a sua capacidade de fazer gráficos (tais como linhas e círculos) mais precisos, mais variados e fáceis de usar do que os gráficos que você usou no Manual do COLOR BASIC. Nós os chamamos "gráficos de "alta resolução".

Vamos começar com o elemento gráfico básico — um simples ponto — e construir a partir daí.

Para o COLOR BASIC EXTENDIDO, colocar um ponto na tela é uma tarefa realmente muito simples. Digite o programa a seguir e você verá:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 PSET (10,20,8)
40 GOTO 40
```

Não se preocupe com nenhuma das palavras novas. PMODE e SCREEN, por exemplo, determinam o grau de detalhes e o conjunto de cores usadas. Estes comandos serão vistos mais tarde em outros capítulos.

Execute o programa.

A tela deve ficar de um tom bege e se você olhar cuidadosamente, verá um pequeno ponto laranja no canto superior esquerdo da tela. Este ponto foi posto lá pelo PSET da linha 30 do programa. Esta instrução permite colocar um ponto de uma determinada cor em qualquer parte da tela.

PSET tem a seguinte sintaxe:

PSET (x,y,c)

x	determina a posição no eixo X (horizontal) e é uma expressão numérica de 0 a 255.
y	determina a posição no eixo Y (vertical) e é uma expressão numérica de 0 a 191
c	determina a cor do ponto (código de cores) e é uma expressão numérica de 0 a 8.

NOTA : A coordenada X deve vir sempre antes da coordenada Y.

Nós chamamos estes quadros de "quadros resumo" e você os encontrará até o final do Manual.
De uma maneira geral, eles explicam instruções tais como o PSET. Preste muita atenção para sua organização e pontuação. Você substituirá os parâmetros, tais como, X, Y e C, por valores específicos.

Apesar de você não poder ver, o computador dividiu sua tela em aproximadamente 50.000 pontos. Cada um desses pontos pode ser colocado numa matriz se você imaginar sua tela formada por um retângulo com 256 pontos no sentido horizontal e 192 pontos no sentido vertical. Será necessário fazer referência a ambas as coordenadas (X,Y) para especificar qual é exatamente a posição que tem em mente.

É muito simples como o computador faz linhas grossas ou finas. Em alta resolução, ele usa uma pequena marca para situar um ponto. Em baixa resolução, ele combina quatro pequenos pontos para fazer um grande. Conseqüentemente, sempre que definir a cor de um dos quatro pequenos pontos, você define a cor dos outros. É preciso especificar sempre a coordenada X com valores de 0 a 255 e a coordenada Y, de 0 a 191, não importa qual a resolução usada.

Nos apêndices você encontrará um Gabarito para Telas de Gráficos de 256 X 192. Isto lhe será de grande ajuda, caso queira traçar gráficos usando qualquer uma das funções gráficas.

Por exemplo, a linha 30 do programa menciona o ponto laranja

```
30 PSET (10,20,8)
```

Esqueça o "8" dentro dos parênteses por enquanto, já que ele determina a cor do ponto e não sua localização.

Você pode verificar como o PSET determina a localização do ponto, contando horizontalmente no eixo X até 10 e, verticalmente, no eixo Y, até 20.

Caso quisesse colocar um ponto no centro da tela, você teria que contar até 128 no eixo X e até 96 no eixo Y.

A instrução ficaria assim:

```
PSET (128,96,8)
```

Você acha que poderia colocar um segundo ponto na tela, da mesma cor do primeiro? Que tal colocar um no canto inferior direito (255 no eixo X e 191 no eixo Y)?

Para colocar esse ponto, acrescente uma outra linha ao programa:

Nota muito importante:
Este computador é capaz de gerar 9(nove) cores distintas. Elas são nominalmente definidas como: preto, verde, amarelo, azul, vermelho, bege, azul esverdeado, carmim e laranja. Entretanto, os tons das cores que realmente serão produzidos pelo seu aparelho e o grau de diferença entre as tonalidades, vai depender da qualidade e do ajuste de cor do seu aparelho de televisão e não do computador. Sugerimos que execute o programa "Ajuste de cores", que se encontra no Manual de Operação deste computador e ajuste a cor conforme sua preferência antes de executar qualquer um desses programas.

```
35 PSET (255,191,8)
```

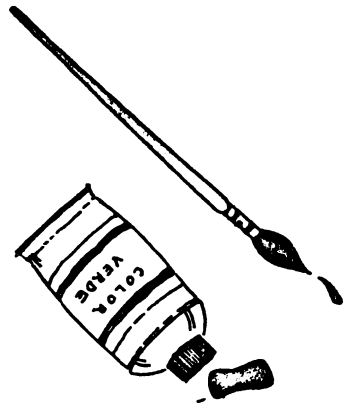
Execute o programa. Deve aparecer um outro ponto no canto inferior direito da tela.

Liste seu programa.

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 PSET (10,20,8)
35 PSET (255,191,8)
40 GOTO 40
```

Nada mal para um começo...

... MAS QUE TAL A COR?

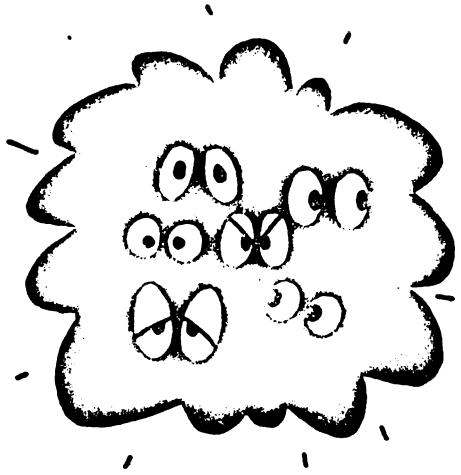


Até aqui nós falamos duas coisas sobre o "c":

1 - Ele determina a cor do ponto;

2 - Ele pode ser qualquer expressão numérica de 0 a 8.

Você deve se lembrar que no Manual do COLOR BASIC cada uma das nove cores disponível ti -



Para ajudá-lo a lembrar a diferença, tente ler PSET como "point set" e PRESET como "point reset" (e não como "pre-set").
Nós teremos exercícios do tipo "FAÇA VOCÊ MESMO O SEU PROGRAMA" até o final do Manual. Você encontrará algumas das soluções possíveis para os exercícios nos apêndices, mas não se esqueça, nossas sugestões geralmente não são as únicas soluções e às vezes, nem mesmo as melhores.

nha um código numérico (veja os apêndices). Nas linhas 30 e 35 do programa, especificamos laranja (código 8) para a cor do ponto. Dentro dos limites permitidos, você pode mudar esta cor mudando o seu código.

Caso não consiga a cor que escolheu, existe uma boa razão para isso. Discutiremos este assunto em capítulos mais adiante, quando falarmos sobre as diferentes "modalidades" gráficas, mas por enquanto, não se preocupe se não conseguir sempre a cor desejada.

ONDE ESTAVA VOCÊ QUANDO AS LUZES SE APAGARAM?

É tão fácil apagar um ponto quanto pô-lo na tela. Quem adivinha como? Lembra-se de como fez o "olhinho" do computador piscar no Manual do COLOR BASIC? Você usou RESET que apaga um ponto específico da tela. Agora você pode fazer o mesmo no COLOR BASIC EXTENDIDO usando o PRESET.

PRESET	(x,y)
x	determina a posição no eixo X e é uma expressão numérica de 0 a 255.
y	determina a posição no eixo Y e é uma expressão numérica de 0 a 191.

Note bem que com o PRESET não é preciso determinar a cor, já que o computador automaticamente apagará o ponto.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 1.1

Que tal se familiarizar com a localização dos vários pontos da tela de gráficos na televisão, usando o Gabarito de Tela de Gráficos? Selecione vários pontos no Gabarito, identifique-os em termos de suas coordenadas (x,y) e mostre-os na tela usando o programa que criamos no início. Não mude nenhuma linha do programa, exceto aquelas que contêm PSET (x, y,c).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 1.2

Você se lembra da função RND (randômica) do Manual do COLOR BASIC? Caso não se lembre, volte, dê uma olhada e então faça um pequeno programa que preencha a tela com pontos aleatórios de cores aleatórias.

VERIFICAÇÃO DE APRENDIZADO

Incluímos verificações de aprendizado como esta no final de cada capítulo. Elas o testarão a respeito dos conceitos importantes mostrados em cada capítulo. Se você tiver problemas em respondê-las, reveja o capítulo antes de passar adiante. Você encontrará a resposta de cada questão nos apêndices.

Escolha a resposta certa:

1) PSET (x,y,c) permite ao computador:
   a - mostrar um ponto
   b - tocar uma canção
   c - estabelecer a cor de fundo

2) A coordenada x pode ser qualquer número entre:
   a - 0 - 191
   b - 0 - 255
   c - 191 - 255
   d - 0 - 8

3) A coordenada y pode ser qualquer número entre:
   a - 0 - 191
   b - 0 - 255
   c - 191 - 255
   d - 0 - 8

4) O "c" em PSET (x,y,c) indica:
   a - uma expressão numérica entre 0 e 8
   b - o código de uma cor
   c - a especificação da cor de um ponto
   d - todas as respostas anteriores
   e - nenhuma das respostas anteriores

CAPÍTULO 2

ANDE NA LINHA

2

ANDE NA LINHA

Na verdade, nada mudou. Você continua usando blocos (ou pontos) para fazer uma linha, porém os gráficos de alta resolução usam pontos menores.

Lembra-se da "cara" do computador que você fez no Manaul do COLOR BASIC ? Um pouco rudimentar, não? Entre outras coisas, o COLOR BASIC EXTENDIDO permite aperfeiçoar a "cara" do computador de modo que ele não fique parecendo com uma coisa.

Muito bem, você já sabe como colocar um ponto na tela e até mesmo muitos pontos. Grande coisa!

DE VOLTA AO JARDIM DE INFÂNCIA

Se marcasse dois pontos em um pedaço de papel, qual seria a primeira coisa que você pensaria em fazer? Claro, ligá-los por uma linha reta. Que tal tenta isso?

Lembra-se que no Manual do COLOR BASIC você teve que usar uma série de blocos para fazer uma linha? Isto deu certo para o caso de uma linha horizontal ou vertical, mas uma linha em diagonal ficou parecendo mais uma escada do que uma linha reta (e neste caso, melhor nem pensar em fazer um círculo).

Seu computador agora dispõe de uma linha horizontal e vertical mais aperfeiçoada e uma ótima linha diagonal. Além disso, agora é possível variar a espessura da linha.

Uma maneira de fazer isto é usar a instrução "LINE" do COLOR BASIC EXTENDIDO. Usaremos o mesmo programa que mostra os pontos, mas com algumas modificações, e para facilitar, chamaremos este programa de "LINHAS".

Primeiro, mude a linha 30 do programa para :

```
30 LINE (0,0)-(255,191),PSET
```

e digite

```
35 (ENTER)
```

para cancelar a linha 35 do programa. Seu programa deve ficar assim:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 LINE (0,0)-(255,191),PSET
40 GOTO 40
```

Agora execute o programa. Você deve ter uma linha laranja indo da parte superior esquerda da tela para a parte inferior direita com um fundo de cor bege. Lindo, não?

Que tal mudar a direção da linha, digamos, da extremidade inferior esquerda para a superior direita?

Modifique a linha 30 para:

```
30 LINE (0,191)-(255,0),PSET
```

Isto já lhe dá uma idéia de como a instrução LINE funciona, não?

X DETERMINA O PONTO

Mais tarde falaremos mais a respeito de PSET,PRESET, B,BF. Por enquanto, concentraremos nossa atenção somente nas coordenadas (X,Y) porque elas representam um conceito muito importante para nós.
Seu programa deve ficar assim:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191) ,
   PSET
30 LINE (0,191)-(255,0) ,
   PSET
40 GOTO 40
```

Que tal fazer uma interseção de linhas?

Reponha a linha 30 original (antes,porém, renumere-a como linha 25) e execute o programa.

Apareceram na tela duas linhas laranjas cruzando-se no centro da tela? Na verdade, você pode colocar na tela quantas linhas quiser, basta aprender como. E talvez seja melhor fazer isto agora mesmo.

Já que uma linha reta é definida como a distância mais curta entre dois pontos, a primeira coisa que você tem que fazer é decidir onde colocar os dois pontos.

Os pontos onde nossa linha começava e terminava foram colocados entre parênteses.

```
25 LINE (0,0)-(255,191), PSET
```

Esses pontos são as coordenadas (X,Y).

Cada um desses pontos pode ser identificado no Gabarito da Tela de Gráficos.

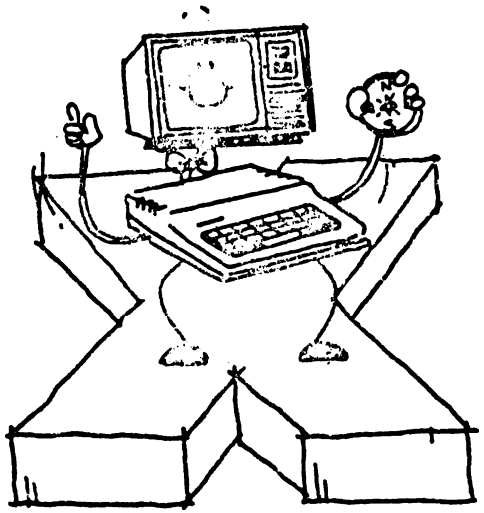
Esses pontos podem ser especificados no seu comando LINE da seguinte forma:

NOTA: Este PSET não é o mesmo que vimos no Capítulo 1. Este aqui não determina a posição de um ponto na tela ou um código de uma cor, mas tem uma função inteiramente diferente. Vá em frente, e verá que é muito mais fácil identificar este PSET do que você imagina.

LINE (x1,y1) - (x2,y2), a, b

(x1, y1)	determina o ponto de partida da linha. x1 é uma expressão numérica de 0 a 255; y1 é uma expressão numérica de 0 a 191.
(x2, y2)	determina o ponto final da linha. x2 é uma expressão numérica de 0 a 255; y2 é uma expressão numérica de 0 a 191.
a	é igual a PSET ou PRESET.
b	é igual a B ou BF. É opcional.

NOTA: Se o ponto de partida não for indicado, o computador começará no último ponto definido pelo programa. Caso não tenha sido definido anteriormente um ponto pelo programa, (128, 96) será usado como ponto de partida.



Tente usar o Gabarito da Tela de Gráficos para marcar os pontos que usamos para cruzar as linhas no programa "LINHAS".

Lembre-se que o primeiro par de coordenadas determina o ponto de partida da linha, e o segundo par, após o hífen, determina o ponto final.

Note também que nem sempre é necessário especificar o ponto de partida da instrução LINE (o primeiro par de coordenadas). Neste caso, o computador começará automaticamente no último ponto definido pelo programa. Caso você não tenha usado a instrução LINE anteriomen

te no programa, o computador começará a linha no ponto de coordenadas (128,96).

Por exemplo:

```
25 LINE (0,0)-(255,191),PSET
30 LINE -(191,0),PSET
```

A linha 20 do programa desenha uma linha que começa em (0,0) e se estende até (255,191). A linha 30 desenha uma linha que vai do ponto (255,191) até o ponto (191,0).

Se você não determinar um ponto de partida, o ponto final deverá ser precedido por um hifen (-).

PSET OU PRESET?

A especificação da cor está em outro lugar no programa e não no comando LINE, portanto, voltaremos a isto mais tarde. Por enquanto, concentre sua atenção apenas no comando LINE.

Vamos dar uma outra olhada nas instruções do programa que criaram as linhas intercruzadas.

```
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
```

Até agora falamos das coordenadas dentro dos parêntesis, mas se você olhar de novo aquele "quadro resumo", verá que existe uma alternativa para o próximo parâmetro: PSET ou PRESET.

Antes de explicarmos a diferença entre PSET e PRESET, tente trocar PSET na linha 25 por PRESET e execute o programa:

```
25 LINE(0,0)-(255,191),PRESET
```

Que aconteceu? Sua tela deve apresentar uma única linha laranja, criada pela linha 30 , que vai do canto inferior esquerdo da tela até o canto superior direito. Tente descobrir o que aconteceu quando PSET foi substituído por PRESET.

Troque o PSET na linha 30 por PRESET e veja o que acontece. E agora? Sua tela ficou em branco? Não! Na realidade, as linhas ainda estão lá, só que você não pode vê-las.

AGORA, O MOMENTO PELO QUAL VOCÊ TANTO ESPERAVA

.PSET mostra a linha, com a cor especificada para o primeiro plano.
.PRESET apaga a linha, com a cor especificada para o fundo.

Neste programa, em particular, usamos laranja como cor do primeiro plano (a linha laranja, certo?) e bege como cor do fundo (a cor da tela, tão logo você execute o programa).

Será mais fácil distinguir a cor do primeiro plano da cor do fundo, se você interpretar como uma linha de uma cor sobreposta a uma tela de outra cor.

Seria a mesma coisa que usar diferentes cores de giz para escrever num quadro-negro. Alguns quadros são verdes, outros pretos, etc. A cor do quadro seria a cor do fundo e o

giz seria a cor do primeiro plano.

Se você tentar escrever num quadro verde com um giz também verde, será impossível ver o que você está escrevendo. Foi exatamente isto que aconteceu quando, ao trocar PSET por PRESET na linha 25, você deu instruções ao computador para mudar a cor da linha criada pela linha 25, para a cor do fundo (da tela)! Então, é claro, você não poderia vê-la.

ENCAIXOTANDO

Estamos quase terminando com o assunto LINE, mas ainda há alguns itens que precisam ser vistos.

Com o COLOR BASIC EXTENDIDO é simples fazer uma caixa (retangular ou quadrada) sem ter que incluir no seu programa, quatro linhas separadas para formar os quatro lados da caixa.

Tudo o que você tem a fazer é determinar as extremidades de uma diagonal do quadrado e incluir B na instrução. Aí, quando você executar o programa, o computador criará um quadrado em vez de uma linha reta.

Numa simples definição, B significa "BOX" (caixa).

Para que você entenda melhor, vamos usar o programa "LINHAS" de novo:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

Esse programa cria duas linhas laranjas que se cruzam no centro da tela. Cancele a linha 30 e acrescente o item "B" à linha 25, após o PSET, e veja o que acontece quando você executa o programa.

```
25 LINE (0,0)-(255,191),PSET,B
```

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 2.1

Faça um programa que crie uma caixa com duas linhas se cruzando no centro.

PINTE ESTA CAIXA

Nós já estamos quase no fim do assunto LINE, portanto vamos tentar encerrá-lo de uma vez.

Se você der uma olhada no "quadro resumo" do comando LINE, verá que existe a opção de acrescentar "F" ao item opcional "B".

Esta opção lhe permite preencher a caixa com a cor do primeiro plano. Experimente. Temos certeza de que vai gostar.

Modifique a linha 25 para:

```
25 LINE (0,0)-(255,191),PSET,BF
```

Que tal? Você deve ter conseguido uma grande caixa laranja (255 X 192) em um fundo bege.

Lembre-se que você fez quase tudo isso com um único comando LINE. Era isso que queríamos dizer quando falamos no início da grande capacidade do COLOR BASIC EXTENDIDO.

PINTANDO O SETE

Já que estamos falando em cor do primeiro plano e cor do fundo há algum tempo, talvez seja hora de você aprender pelo menos algumas maneiras de controlar as cores.

Dentro das possibilidades, o comando gráfico COLOR lhe permite determinar as cores do

primeiro plano e do fundo (veja PMODE e SCREEN mais adiante neste Manual).

Sua sintaxe é:

COLOR primeiro plano, fundo	
primeiro plano	é uma expressão numérica de 0 a 8 e representa um código de cor.
fundo	é uma expressão numérica de 0 a 8 e representa um código de cor.

Explicaremos porque estas são as únicas cores disponíveis, quando discutirmos PMODE e SCREEN nos próximos capítulos.

Por enquanto você terá que aceitar nossa palavra de que as únicas cores disponíveis neste programa são bege(5), azul esverdeado(6), carmim(7) e laranja(8).

Se você não determinar as cores do primeiro plano e do fundo, o computador automaticamente escolherá o número mais alto dos códigos de cores disponíveis para a cor do primeiro plano e o mais baixo para a cor do fundo.

É por isso que as linhas cruzadas eram laranja(8) em fundo bege(5) no seu programa "LINHAS".

Com a instrução COLOR você pode mudar as cores do primeiro plano e do fundo (dentro das alternativas disponíveis).

Para ver como a instrução COLOR funciona, insira a linha 6 no seu programa e cancele a linha 25.

```
6 COLOR 5,7
```

Agora execute. Você não acha que as linhas cruzadas bege sobre um fundo carmim dão um ótimo visual?

Agora re-digite a linha 6 (ou, caso você tenha lido o capítulo 10, edite-a) de modo a inverter as cores de primeiro plano e de fundo.

```
6 COLOR 7,5
```

Linhas carmim em fundo bege também ficam bonitas. Nos próximos capítulos você aprenderá como dispor de mais cores.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 2.2

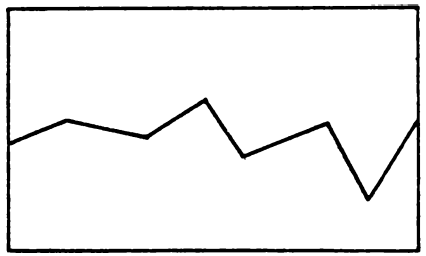
Está pronto para tentar fazer seu próprio programa usando LINE? Você poderia fazer uma casa? Comece com as linhas 5, 10 e 20 do programa "LINHAS" e continue a partir daí.Não se esqueça de acrescentar:

1 - Uma porta na frente, claro;
2 - Pelo menos uma janela (não esqueça de acender ou apagar as luzes);
3 - Uma chaminé (você não precisará de um limpador de chaminés, por enquanto).

O tipo de desenho é com você, mas, incluímos um programa modelo no Apêndice. Não se preocupe com as maçanetas da porta por enquanto. Faremos isto mais tarde.

NOTA: Será muito mais fácil desenhar e construir sua casa se você marcar previamente os pontos no Gabarito da Tela de Gráficos.

Guarde este programa em "cassete" já que você vai precisar dele mais tarde.



FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 2.3

Este deve ser um grande desafio para você. Se lembra que definimos uma linha reta como sendo a menor distância entre dois pontos? Bem, vamos colocar alguns metros extras entre os nossos dois pontos usando LINE (juntamente com algumas técnicas de programação que você aprendeu no Manual do COLOR BASIC) e desenhar uma linha tortuosa entre dois pontos.

Para começar, você pode usar as linhas 5, 10 e 20 do programa LINHAS.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Para desenhar uma linha (LINE) você tem que saber:

   a - onde ela começa
   b - onde ela termina
   c - seu comprimento
   d - todas as respostas anteriores
   e - nenhuma das respostas anteriores

2) Se você quiser que a linha seja da cor do fundo, acrescente o sufixo .....

3) A maneira mais fácil de obter um quadrado vazio usando LINE é:

   a - acrescentar BF

COLOR não é uma instrução de ação. Deve sempre preceder uma instrução de ação (como PCLS, DRAW, etc) para que as cores do primeiro plano e do fundo sejam realmente mudadas.

b - acrescentar B

c - usar LINE quatro vezes

4) A instrução COLOR permite:

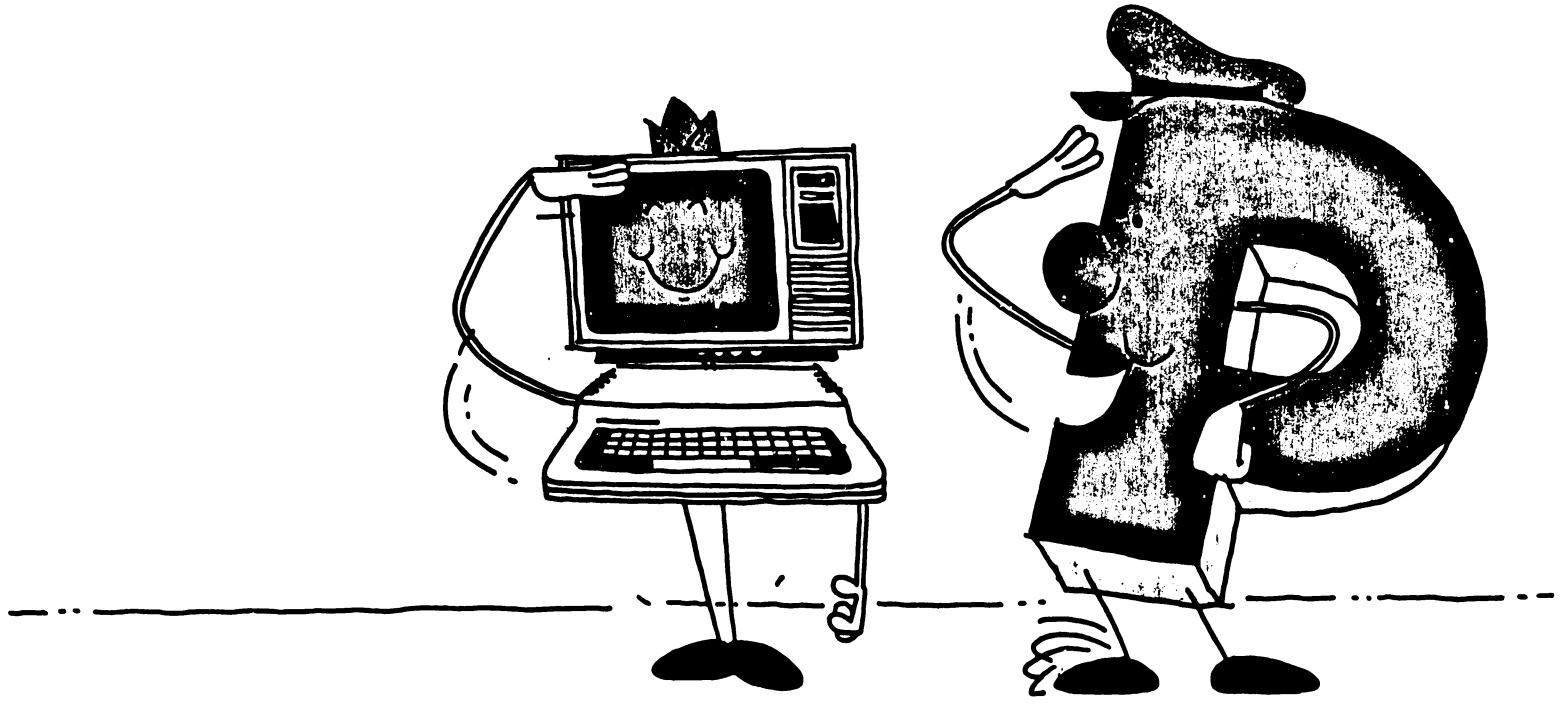
a - escolher a cor do fundo

b - colorir um quadrado

c - escolher a cor do primeiro plano e do fundo

5) A coordenada ..... deve sempre preceder a coordenada ..... quando você determinar um ponto no Gabarito da Tela de Gráficos.

CAPÍTULO 3



CONCEITOS IMPORTANTES SOBRE PÁGINAS

3

CONCEITOS IMPORTANTES SOBRE PÁGINAS

Quando você emite qualquer tipo de informação para a televisão, esta informação é armazenada numa parte da memória chamada de "Memória de Vídeo" (Vídeo RAM). O circuito de televisão do computador então lê a "Memória de Vídeo" e gera a representação visual na tela.

Pense em PCLEAR como "page clear" - páginas limpas.

A área de memória de uso normal é grande o suficiente para armazenar textos (letras e números) mas não para armazenar gráficos (círculos, linhas, etc). Conseqüentemente, o computador necessita de mais área de "Memória de Vídeo" para apresentação de gráficos.

Para ajudá-lo a lembrar-se, você poderia ler PMODE como "page-mode" - modalidade de páginas.

Esta outra parte da "Memória de Vídeo" pode ocupar até oito "páginas de memória" e, dependendo da necessidade, você pode reservar de 1 a 8 "páginas de memória" com o comando PCLEAR — uma "página" contém 1.536 posições de memória. PCLEAR determina o tamanho da "Memória de Vídeo" disponível para gráficos e, caso não seja especificado, o computador automaticamente assume PCLEAR igual a quatro "páginas".

Sua necessidade de "páginas" depende da quantidade de detalhes e cores requeridos pelo programa de gráficos — quanto maior o número de cores e detalhes, maior a memória necessária.

COMPUTADOR À LA PMODE

Mude o programa "LINHAS" para criar novamente linhas cruzadas:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

Vamos agora dar uma olhada na linha 5 que contém uma das funções gráficas mais importantes do seu computador - o PMODE.

```
5  PMODE 1,1
```

Esta instrução permite escolher o grau de resolução (quantidade de detalhes) e selecionar a "página de memória" pela qual você deseja começar a apresentação.

Antes de continuarmos, vamos dar uma olhada no "quadro resumo" de PMODE.

PMODE grau de resolução, página inicial

grau de resolução — é uma expressão numérica de 0 a 4 que especifica o grau de resolução da tela gráfica. É opcional, e se omitido, o computador usa o último valor especificado.

página inicial — é uma expressão numérica de 1 a 8 que especifica por qual "página de memória" de 1,5K você quer começar a apresentação. É opcional, e se omitido, o computador usa o conjunto de páginas anteriores.

NOTA: Ao ligar o computador, ele assume PMODE 2,1.

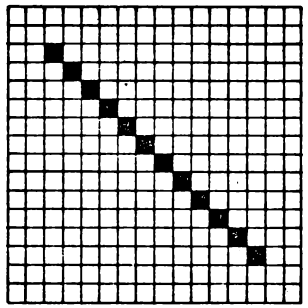
Dissemos anteriormente que ao trabalhar com gráficos, o computador divide a tela de vídeo em 256 pontos no eixo X (horizontal) e 192 no eixo Y (vertical). Esta matriz pode ser considerada "normal" para gráficos de alta resolução.

Não é necessário usar PMODE quando estiver usando a tela para programas de "texto".

Uma matriz de 256 X 192 dá a mais alta resolução (mais detalhes) disponível no COLOR BASIC EXTENDIDO. Por conveniência, a chamaremos de "PMODE 4".

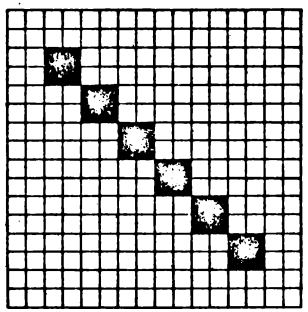
O computador tem 5 PMODEs disponíveis (0-4), cada um com uma capacidade diferente. Você já sabe que a única diferença entre a alta e a baixa resolução (PMODE 4 e PMODE 0, respectivamente) é o tamanho do ponto: a alta resolução usa uma única marca para formar um

ponto enquanto a baixa resolução usa quatro destas pequenas marcas para fazer um pequeno bloco.



ALTA RESOLUÇÃO

Apesar do tamanho da matriz variar (através da combinação de pequenos pontos para se fazer outros maiores), todos os PMODEs são mapeados em uma matriz de 256 X 192 e você tem que especificar as coordenadas dentro destes limites. Por exemplo, (128,96) é o centro da tela, não importa com que PMODE você esteja trabalhando. Do mesmo modo, (255,191) está na extremidade inferior direita da tela tanto para o PMODE 0 quanto para o PMODE 4.



BAIXA RESOLUÇÃO

TABELA DE RESOLUÇÃO DA TELA DE GRÁFICOS

PMODE Nº	TAMANHO DA MATRIZ	NÚMERO DE CORES	PÁGINAS USADAS	TAMANHO DO PONTO
4	256X192	2 cores	4	
3	128X192	4 cores	4	
2	128X192	2 cores	2	
1	128X96	4 cores	2	
0	128X96	2 cores	1	

TABELA 3.1

Lembre-se que a tela de gráficos está sempre cheia de "pontos". A questão é simplesmente quantos, de que tamanho e de que cor são esses pontos.
Apesar de nós a chamarmos de baixa resolução, ela ainda tem mais detalhes (mais alta resolução) do que os gráficos de 64 X 32 tipo SET/RESET na tela normal de textos.

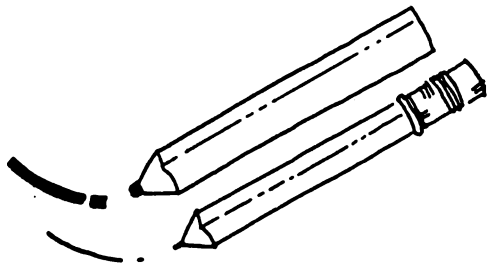
Na tabela 3.1 você pode ver facilmente porque o PMODE 4 dá a mais alta resolução(maiores detalhes). É porque ele tem os pontos menores, e portanto em maior número. Por isso, de agora em diante faremos referência ao PMODE 4 como de "alta resolução".

Abaixo, na tabela, vem os PMODEs 3 e 2. Já que os pontos destes PMODEs são duas vezes o tamanho de PMODE 4, a resolução não é tão alta. Nós os chamaremos de "média resolução".

Os pontos dos PMODEs 1 e 0 são 4 vezes maiores que o PMODE 4, e duas vezes maiores que os dos PMODEs 3 e 2. São, por isto, as modalidade de gráficos com menor resolução e nós os chamaremos de "baixa resolução".

Podemos considerar que a alta resolução precisa de 4 vezes mais pontos que a baixa resolução, para preencher a tela.

Os próximos exemplos mostram bem a diferença entre os PMODEs 4 e 0, com relação ao tamanho dos pontos e à resolução.



Você se lembra de quando estava na escola e usava aqueles lápis enormes que faziam linhas grossas no papel? Quando você finalmente passou para as séries mais adiantadas, passou também a usar lápis menores que faziam linhas mais finas podendo, então, acrescentar mais detalhes aos desenhos.

Com o seu computador acontece a mesma coisa. A baixa resolução pode ser comparada a um lápis enorme com um ponto duas vezes maior que o ponto delicado de alta resolução. É por isso que, com a alta resolução, você pode fazer linhas diagonais sem que elas fiquem parecendo degraus de escada. Porém, por enquanto, nada podemos dizer sobre círculos, já que são completamente redondos.

Chega de conversa. Vamos por o programa "LINHAS" para trabalhar. A linha 5 do programa ainda nos interessa.

```
5  PMODE 1,1
```

Mude-o para alta resolução (PMODE 4):

```
5  PMODE 4,1
```

Agora execute o programa. Você deve notar duas diferenças logo a primeira vista:

1 - A linha deve estar mais bem feita porque está em alta resolução.

2 - Deve também haver uma mudança de cor já que você mudou de uma modalidade de quatro cores para uma modalidade de duas cores (Mais adiante explicaremos isto melhor).

Era isso que queríamos dizer quando nós referíamos às "cores disponíveis", isto é, as cores disponíveis no PMODE que você estiver usando.

Como pode ver no programa acima, você deve prestar muita atenção no PMODE e verificar se a modalidade escolhida tem a seleção de cores e o detalhamento desejados.

Isto quer dizer que sempre que estiver em PMODE 4, por exemplo, você está preso a combinação de duas cores. Você só pode usar as combinações de preto e verde ou de preto e bege — e nenhuma outra.

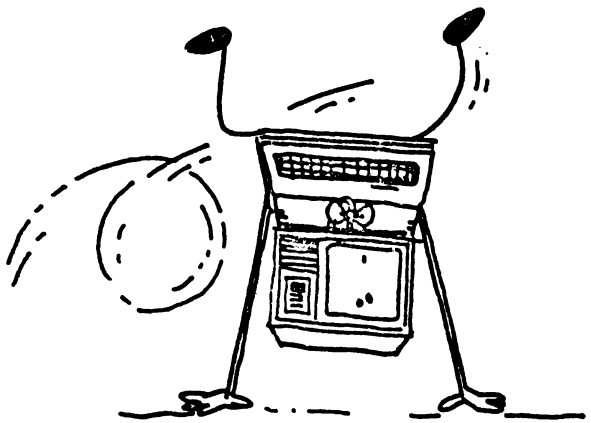
TABELA DE CONJUNTO DE CORES			
PMODE Nº	CONJUNTO DE CORES (Veja SCREEN)	COMBINAÇÃO DE DUAS CORES	COMBINAÇÃO DE QUATRO CORES
4	0	Preto / Verde	-
	1	Preto / Bege	-
3	0	-	Verde/amarelo/azul/vermelho
	1	-	Bege/azul esverdeado/carmim/ laranja
2	0	Preto / Verde	-
	1	Preto / Bege	-
1	0	-	Verde/amarelo/azul e vermelho
	1	-	Bege/azul esverdeado/carmim/ laranja
0	0	Preto / Verde	-
	1	Preto / Bege	-

TABELA 3.2

O comando PMODE 4 permite usar a alta resolução mas dá uma escolha limitada de cores. Se quiser usar a cor vermelha, por exemplo, você não pode usar o PMODE 4. Terá que passar para uma resolução mais baixa, digamos, PMODE 3 e fixar o conjunto de cores — estabeleça 0 (cores disponíveis: verde, amarelo, azul e vermelho). (Veja Tabela 3.2)

Temos aqui um programa que mostra um quadrado passando ciclicamente pelas diferentes modalidades. Repare como o "quadrado" passa de linhas grossas para linhas finas e muda de cores, conforme o programa assume diferentes valores de PMODE.

```
5  FOR MODE = 0 TO 4
10 PMODE MODE,1
20 PCLS
30 SCREEN 1,1
40 LINE (75,50)-(125,100),PSET,B
50 FOR Y = 0 TO 500: NEXT Y
60 NEXT MODE
70 GOTO 5
```



MUDANDO DE PÁGINA

Mencionamos anteriormente que o COLOR BASIC EXTENDIDO dividiu a "Memória de Vídeo" de gráficos do computador em 8 "páginas de memória".

O segundo parâmetro de PMODE, "página inicial", permite selecionar por qual destas páginas de memória você quer começar a apresentação de gráficos. Se a página inicial não for

especificada, o computador começará automaţicamente pela página 1, ou pela última página indicada. Isto será útil para muitos tipos de gráficos.

Antes de passarmos adiante, dê uma olhada na Tabela 3.3, a seguir:

	Para usar...	Reserve páginas com...
n = 1 a 5	PMODE 4,n	PCLEAR 4 + (n - 1)
	PMODE 3,n	PCLEAR 4 + (n - 1)
n = 1 a 7	PMODE 2,n	PCLEAR 2 + (n - 1)
	PMODE 1,n	PCLEAR 2 + (n - 1)
n = 1 a 8	PMODE 0,n	PCLEAR 1 + (n - 1)

TABELA 3.3

NOTA: Cada página de memória reservada ocupa 1.536 posições de memória (bytes).

Esta tabela detalha o número de páginas necessárias para cada PMODE. Repare que quanto maior a resolução e maior o número de cores disponíveis, mais o computador precisa de memória, e, conseqüentemente, sobra menos espaço para o seu programa.

Em outras palavras, o PMODE 0 precisa de uma página, o PMODE 2, de duas páginas, etc.

Após executar este programa, você entenderá melhor o que queremos dizer:

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 LINE (110,20)-(120,30),PSET,B
40 GOTO 40
```

Nada complicado, não? Um pequeno quadrado no alto da sua tela. Agora mude a linha 5 para:

```
5  PMODE 3,2
```

e execute o programa. De onde apareceu esse ?FC ERROR?

Ao mudar a página inicial de 1 para 2 você gerou um erro. O PMODE 3 exige quatro páginas de memória de gráficos. Quando você começa na página 2, isso significa que são necessárias as páginas 2, 3, 4 e 5. Mas você usou PCLEAR somente para as páginas 1-4. A página 5, portanto, não está disponível.

Na próxima seção, mostraremos como solucionar este problema para que você possa usar o

PMODE 3,2 (veja PCLEAR) mas por enquanto...

> FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 3.1
>
> Você já sabe que os PMODEs 4 e 3 exigem quatro páginas de memória e portanto, você só pode selecionar a página 1 como inicial. Mas o que aconteceria se você passasse para PMODE 2, 1 ou 0? Em que "páginas de memória" você poderia começar nesses PMODEs sem gerar um ?FC ERROR? Mude os parâmetros do PMODE para descobrir isto (por tentativa e erro) e compare suas respostas com as nossas.

PCLS

O seu programa "LINHAS" ainda está assim?

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

Vamos agora examinar o PCLS, que é a segunda instrução no programa.

PCLS simplesmente "limpa" a tela de gráficos. Em outras palavras, o PCLS tem a mesma função para a tela de gráficos que o CLS para a tela de textos.

O computador usa a cor de fundo para limpar a tela de gráficos, a menos que você determine uma outra cor (disponível).

PCLS	cor
cor	é uma expressão numérica de 0 a 8 e representa um dos códigos de cor disponível. "Cor" é opcional e se omitido, será usada cor de fundo.

No programa "LINHAS", a opção de cor do PCLS foi omitida. Neste caso, o computador usa automaticamente a cor do fundo atual, bege.

Re-digite a linha 10

```
10 PCLS 6
```

e execute o programa. A televisão deve mostrar linhas laranjas num fundo azul esverdeado (código de cor 6).

Não se esqueça que você só pode "limpar" a tela de gráficos usando cores disponíveis.

PCLEAR

Às vezes, quando você usar "páginas de memória", terá que usar a instrução PCLEAR para reservar um determinado número de páginas de gráficos de 1,5K.

Já que a instrução PCLEAR determina o tamanho da "Memória de Vídeo", ela deve ser a primeira ou segunda instrução no seu programa (depois de CLEAR, se CLEAR for usado).

PCLEAR n

n é uma expressão numérica de 1 a 8 e especifica o número de páginas de gráficos que serão reservadas.

NOTA: Quando você liga o computador, a instrução PCLEAR é inicializada automaticamente com 4. Você só precisa usar PCLEAR quando quiser reservar um número diferente de páginas.

Se fizer confusão entre as exigências de memória do programa e as exigências de memória de vídeo, você receberá s mensagem ?OM ERROR (Out of Memory -Falta de Memória).

Você já está pronto para ver o que o PCLEAR pode fazer por você? Você ainda tem aquele programa com o qual tivemos problemas anteriormente?

```
5  PMODE 3,2
10 PCLS
20 SCREEN 1,1
30 LINE (110,20)-(120,30),PSET,B
40 GOTO 40
```

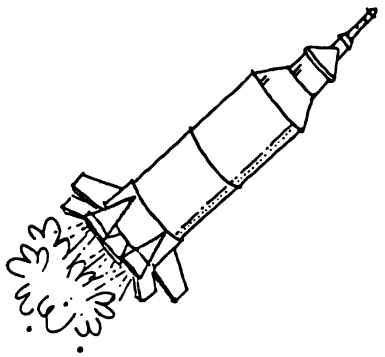
Este programa não apresentou um ?FC ERROR na linha 5 porque você começou na página 2 ? Para consertar isso, insira a linha 4

```
4 PCLEAR 8
```

e execute o programa. Perfeito! Agora você pode começar em qualquer página de 1 a 5.

Você provavelmente está imaginando porque nós simplesmente não deixamos o computador usar PCLEAR 8 o tempo todo. A razão é que PCLEAR 8 diminui a quantidade de memória disponível para seu uso. Algumas vezes você precisará de muito espaço de memória para o programa e outras vezes será necessário mais "páginas de memória" para gráficos. PCLEAR permite estabelecer esse equilíbrio de acordo com as suas necessidades.

Até agora, vimos o que podemos fazer com o PCLEAR. Mas como podemos usar todas aquelas páginas extras de que ele dispõe? A resposta está no segundo parâmetro do PMODE, a página inicial.



Suponhamos que você queira dar movimento a uma figura gráfica. Alterar a página inicial é uma das maneiras de fazer isso.

Pense em todos aqueles longos dias de verão quando você teve que sentar numa sala de aula e aturar todas aquelas infindáveis aulas de Estudos Socias e Aritmética. Com aquela brisa morna e o som de crianças brincando lá fora, como é que você aguentava? Bem, o jeito era ficar desenhando figuras no caderno (até o dia em que te pegaram!)

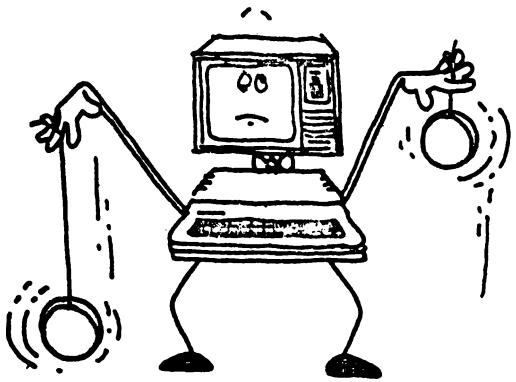
Alguns de seus desenhos preferidos talvez tenham sido foguetes espaciais em várias pági-

nas sucessivas, começando na parte de baixo da folha e subindo gradualmente. E aí, quando você folheava rapidamente as páginas, o foguete decolava rumo à estratosfera (ou a margem superior da página, o que viesse primeiro). Animação instantânea!

O mesmo princípio é aplicado nas páginas de memória. Você consegue simular movimentos na tela desenhando figuras com diferentes localizações e em diferentes páginas; então ao "folhear" as páginas gráficas, você consegue o efeito de movimento.

PARA CIMA E PARA BAIXO, PARA CIMA E PARA BAIXO...

Com uma exceção (Veja CIRCLE, capítulo 5), o programa a seguir incorpora todas as características que já vimos até aqui.



Você provavelmente deve estar pensando que seu computador é meio "pirado", mas agora vamos provar que o que ele é mesmo é um lô-iô. Na verdade, você pode chamar esse programa de lô-iô. Digite-o e execute-o:

```
10 PCLEAR 8
20 FOR P = 1 TO 8 'P=PAGINA, USA TODAS AS 8 PAGINAS
30 PMODE 0, P     'USA UMA PAGINA DE CADA VEZ
40 PCLS
50 LINE (128,0)-(138,10+(P-1)*15),PSET
60 CIRCLE (128,P*15),15
70 NEXT P
80 FOR P = 1 TO 8: GOSUB 110: NEXT P
90 FOR P = 7 TO 1 STEP -2: GOSUB 110: NEXT P
100 GOTO 80
110 PMODE 0,P
120 SCREEN 1,0
```

```
130 FOR T = 1 TO 10: NEXT T
140 RETURN
```

Nós colocamos movimento nos gráficos do computador. Este é o mesmo processo usado nos desenhos animados — desenhar uma série de imagens estáticas e passá-las rapidamente para dar a impressão de movimento.

PCOPY

Você sabia que são necessários mais de 12.000 desenhos individuais para fazer apenas sete minutos de desenho animado? Um computador seria uma grande ajuda, não?

Com PCOPY, você pode copiar o conteúdo dos gráficos de uma "página de memória" para outra. O "quadro resumo" para PCOPY é:

PCOPY origem TO destino

origem e destino	são expressões numéricas entre 1 e 8 especificando páginas de memória.

NOTA: Não use PCOPY em uma página que não esteja reservada. Por exemplo, depois de PCLEAR 4, você pode copiar apenas para as páginas 1-4.

Pense em PCOPY como "Page copy" - copiador de páginas.

Se você quiser copiar os gráficos da página 3 para a página 8, digite:

```
PCOPY 3 TO 8
```

Uma das vantagens é que o PCOPY pode encurtar seu programa eliminando a repetição de comandos.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 3.2

O programa a seguir mostra quatro quadrados (que estão em quatro "páginas de Memória" diferentes)na tela ao mesmo tempo. Execute e diminua o programa usando PCOPY.

```
4  PCLEAR 8
5  PMODE 3,4
10 PCLS
11 SCREEN 1,1
12 LINE (110,20)-(120,30),PSET,B
20 PMODE 3,3
21 SCREEN 1,1
22 LINE (110,20)-(120,30),PSET,B
30 PMODE 3,2
31 SCREEN 1,1
32 LINE (110,20)-(120,30),PSET,B
40 PMODE 3,1
41 SCREEN 1,1
42 LINE (110,20)-(120,30),PSET,B
50 GOTO 50
```

PPOINT

Pense em PPOINT como "Page point — ponto da página.

Ainda existe uma função que você precisa conhecer. O PPOINT testa a cor de um determinado ponto gráfico. Seu programa pode usar essa informação sempre que você quiser.

PPOINT	(x,y)
x	determina a coordenada x do ponto e é uma expressão numérica de 0 a 255.
y	determina a coordenada y do ponto e é uma expressão numérica de 0 a 191.
NOTA :	Use PPOINT no mesmo PMODE que você usou para estabelecer os pontos gráficos, caso contrário, pode ocorrer uma grande confusão.

O exemplo a seguir mostra como PPOINT pode ser uma instrução útil quando incluída num programa.

```
5   PMODE 3,1
10  PCLS
15  SCREEN 1,1
30  X = RND(10)
35  Y = RND(10)
40  C = RND(8)
50  PSET (X,Y,C)
```

```
60  IF PPOINT (5,5)=8 THEN GOTO 105
70  GOTO 30
105 CLS
110 PRINT@ 100,"AGORA A POSICAO(5,5) E' LARANJA"
```

O que acontece é que o computador preenche uma matriz de 10 X 10 (na extremidade superior esquerda da tela) com pontos aleatórios coloridos. Contudo quando o ponto (5,5) for laranja (código de cor 8), o computador vai para linha 105, limpa a tela e imprime a mensagem.

Na linha 60, testamos a cor do ponto (5,5). No programa a seguir, vamos imprimir os códigos de cores de quatro pontos.

Este programa mostra como o computador testa quatro pontos sucessivos.

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 PSET (10,11,5)
35 PSET (10,12,6)
40 PSET (10,13,7)
45 PSET (10,14,8)
50 PRINT PPOINT (10,11)
55 PRINT PPOINT (10,12)
60 PRINT PPOINT (10,13)
65 PRINT PPOINT (10,14)
```

Ao executar o programa, deve aparecer na tela:

```
5
6
7
8
```

que indica o código da cor dos pontos especificados.

Só por curiosidade, mude o PMODE na linha 5 para:

```
5 PMODE 1,1
```

e execute o programa. Você sabe porque o computador imprimiu

```
5
7
7
8
```

na tela?

Que resolução você está usando? Alta ou baixa? Você se lembra que dissemos anteriormente que a baixa resolução combina quatro pequenos pontos para fazer um grande?

O que aconteceu aqui é que (10,12) e (10,13) estão ambos no mesmo bloco. Já dissemos tam

bém que, quando um dos pontinhos no bloco fosse estabelecido, todos os outros pontos seriam da mesma cor. E foi exatamente isto que ocorreu aqui.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 3.3

Use LINE e "página inicial" para simular uma tempestade. (Ponha linhas desordenadas em posições aleatórias, em diferentes páginas). Em seguida, passe as páginas para frente e para trás.

VERIFICAÇÃO DE APRENDIZADO

1) Escolha a resposta certa:

   Qual dos PMODES a seguir lhe daria maior detalhamento?

   a - PMODE 3
   b - PMODE 2
   c - PMODE 4
   d - PMODE 0

2) CLS está para a tela de texto assim como ..... está para a tela de gráficos.

3) Uma única tela de gráficos pode necessitar de até ..... páginas de memória.

4) Relacione cada PMODE com a sua respectiva combinação de cores:

a - PMODE 4	1 - Modalidade de 3 cores
b - PMODE 3	2 - Modalidade de 2 cores
c - PMODE 2	3 - Modalidade de 4 cores
d - PMODE 1	4 - Modalidade de 1 cor
e - PMODE 0	

5) Faça um círculo na combinação de cores correta para a modalidade de quatro cores.

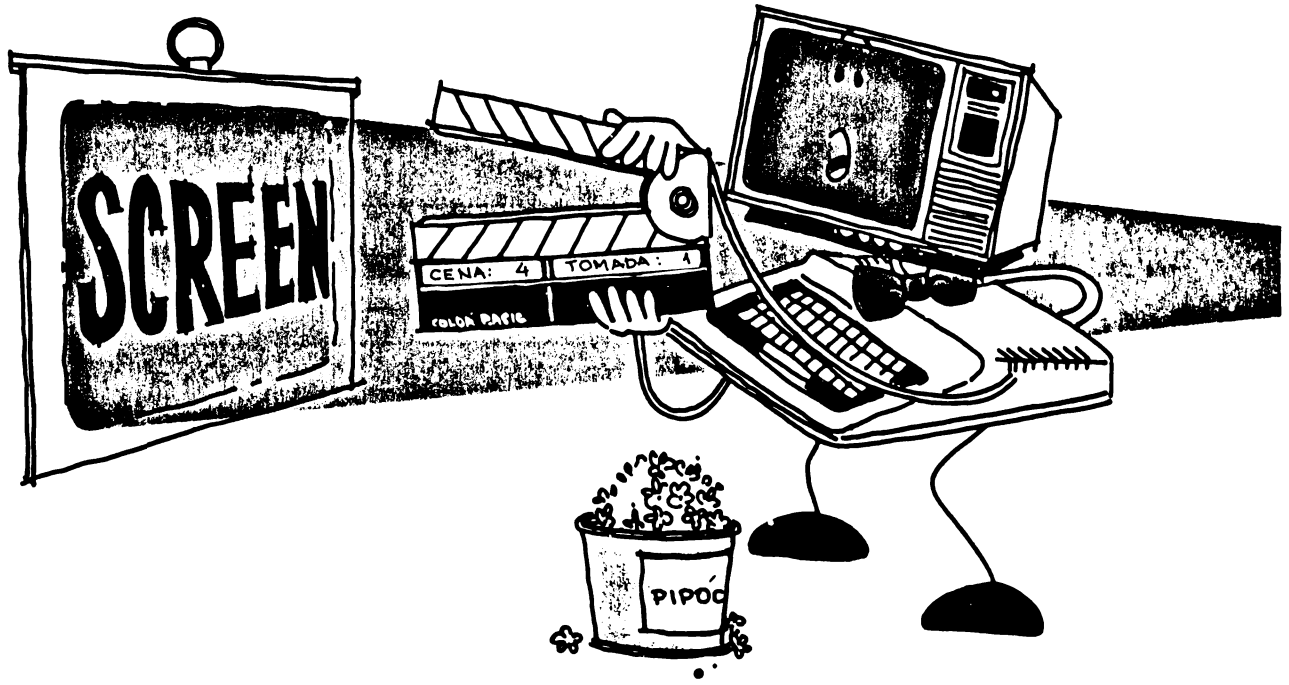
Vermelho Azul Verde Bege Amarelo Preto

6) Qual dos seguintes PMODES requer a maior memória de gráficos? E a menor memória de gráficos?

a - PMODE 0
b - PMODE 1
c - PMODE 2
d - PMODE 3
e - PMODE 4

NOTAS

CAPÍTULO 4



DE OLHO NA TELA...

4

DE OLHO NA TELA...

Você está pronto para estudar outra instrução? Se está, então prepare-se para conhecer a instrução SCREEN.

Dê uma olhada no programa "LINHAS".

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (10,10)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

No momento, vamos concentrar nossa atenção no comando SCREEN, da linha 20.

```
20 SCREEN 1,1
```

SCREEN dá duas instruções ao computador:

A primeira coisa que o SCREEN diz é se você quer usar a tela para gráficos (linhas, círculos, etc) ou texto (letras, números ou mesmo blocos de SET/RESET).

A segunda coisa é qual o conjunto de cores que você quer utilizar.

```
SCREEN tipo, conjunto de cores

Tipo é:   0 - para a tela de texto, ou
          1 - para a tela de gráficos

Conjunto de cores é: 0 ou 1.
```

NOTA: Se o tipo ou o conjunto de cores for um número positivo maior que 1, o computador interpreta este número como sendo 1.

Uma expressão numérica pode substituir 0 ou 1.

O primeiro parâmetro da instrução SCREEN diz ao computador que tipo de tela apresentar (se de gráficos ou de textos).

No programa "LINHAS", tente mudar a linha 20 para:

```
20 SCREEN 0,1
```

Agora execute o programa. O computador parece confuso? (Pressione a tecla (BREAK) para recuperar o controle). Na verdade, o que aconteceu, foi que o computador executou o programa

"LINHAS" mas não pode apresentar qualquer gráfico na tela, uma vez que você lhe deu instruções para usar uma tela de texto.

> NOTA: Sempre que for necessário imprimir um texto no vídeo (PRINT,INPUT), o computador automaticamente executará um comando SCREEN 0,0.

O segundo parâmetro diz ao computador qual o conjunto de cores que deve ser usado. Agora mude a linha 20 para:

```
20 SCREEN 1,0
```

Repare que você tem novamente a tela de gráficos mas o conjunto de cores foi mudado.

A primeira vista, parece que só existem duas opções de cores (0 ou 1). Na verdade, existe uma variedade muito maior, já que você está selecionando um conjunto de cores e não cores individuais.

	Modalidade de 2 cores	Modalidade de 4 cores
Se o conjunto de cores for 0, você tem...	Preto/Verde	Verde/Amarelo/ Azul/Vermelho
Se o conjunto de cores for 1, você tem...	Preto/Bege	Bege/Azul esverdeado/Carmim/Laranja.

Em outras palavras, se você está em uma das modalidades de duas cores (veja PMODE) e especifica 0 para o conjunto de cores, você só terá disponibilidade das cores preta e verde — nenhuma outra.

Nas modalidades de quatro cores, você tem como opção as combinações verde, amarelo,azul e vermelho ou bege, azul esverdeado, carmim e laranja.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 4.1

Você já entendeu a instrução SCREEN? Então faça um programa que mude da tela de textos para a tela de gráficos. Você talvez queira colocar um "loop" no programa de maneira que ele mude a combinação de cores a cada passagem completa pelos comandos do "loop". Desse modo, você pode ver todas as características da instrução SCREEN em ação.

Não cometa o erro de pensar que o SCREEN não pode afetar a tela de textos.

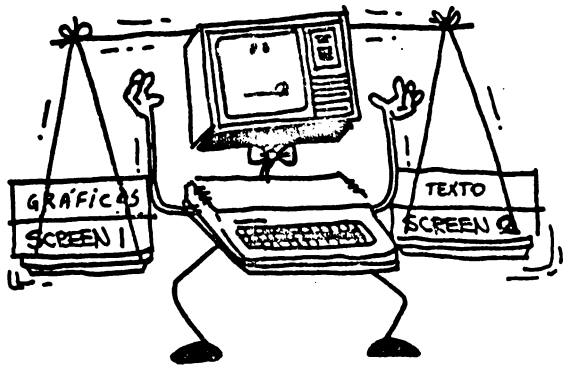
Sempre que você der entrada numa tela de textos,o computador presume que você quer o conjunto de cores 0 (preto/verde).

Mas você pode mudar a cor da tela de textos especificando:

```
SCREEN 0,1
```

Usando o SCREEN desta maneira, é possível mudar a apresentação visual de preto sobre verde para vermelho sobre laranja.

Contudo, o computador voltará automaticamente para preto sobre verde sempre que executar um comando de saída (como PRINT, INPUT ou LINE INPUT). O programa a seguir mostra isso:




```
5  CLS
10 PRINT@ 195, "ISTO E' VERMELHO SOBRE LARANJA"
20 SCREEN 0,1
30 FOR X = 1 TO 1000:NEXT X
40 CLS
50 PRINT@ 195, "ISTO E' PRETO SOBRE VERDE"
60 FOR X = 1 TO 1000:NEXT X
70 GOTO 5
```

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) SCREEN 1,1 diz ao computador para usar:

   a - Tela de gráficos com o conjunto de cores nº 1
   b - Tela de textos com o conjunto de cores nº 1

2) Qual das instruções a seguir resultaria em um erro:

a - SCREEN 0,1

b - SCREEN A,35

c - SCREEN 1,A

d - Todas as respostas anteriores

e - Nenhuma das respostas anteriores

3) A modalidade normal de texto é :

a - SCREEN 1,1

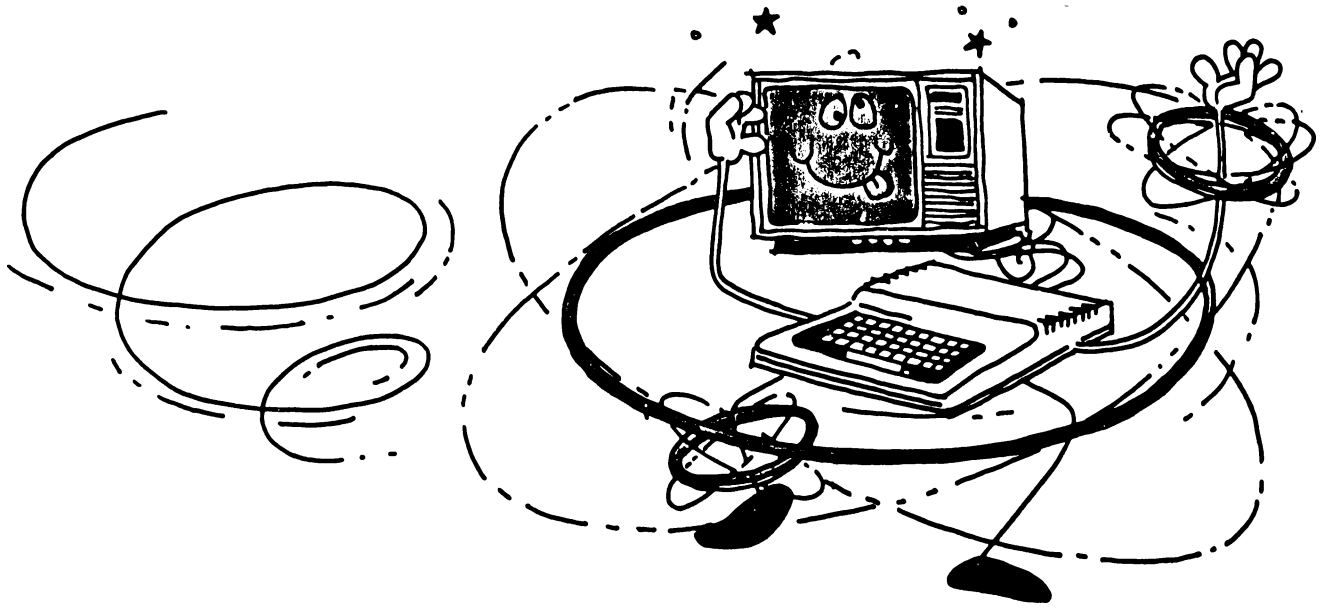
b - SCREEN 1,0

c - SCREEN 0,1

d - SCREEN 0,0

NOTAS

CAPÍTULO 5



ANDANDO EM CÍRCULOS

5

ANDANDO EM CÍRCULOS

Toda esta conversa sobre SCREEN, PMODE e PCLEAR deve ter deixado você meio zonzo! E olhe que você ainda não viu tudo!

Por exemplo, você pode criar um círculo completo, um arco de círculo ou uma elipse usando a instrução CIRCLE, cuja sintaxe é dada abaixo:

CIRCLE (x,y),r, c, al, início, fim

x	determina a coordenada X do centro do círculo e é uma expressão numérica de 0 a 255.
y	determina a coordenada Y do centro do círculo e é uma expressão numérica de 0 a 191.
r	determina o raio do círculo. Uma unidade de medida é igual a um ponto na tela (posição na tela).
c	determina o código da cor do círculo e é uma expressão numérica de 0 a 8. É opcional, e se for omitido, o computador assume a cor de fundo.
al	determina a relação altura/largura do círculo e é uma expressão numérica de 0 a 255. É opcional, e se for omitido, o computador assume 1.
início	determina o ponto inicial do arco de círculo e é uma expressão numérica de 0 a 1. É opcional, e se for omitido o computador assume 0.
fim	determina o ponto final do arco de círculo e é uma expressão numérica de 0 a 1. É opcional, e se for omitido o computador assume 1.

Para se desenhar um círculo são necessários apenas o seu centro e seu raio, e portanto começaremos por eles.

À semelhança das outras funções gráficas, a primeira coisa que você tem que determinar são as coordenadas (x,y).

Neste caso porém, você tem que se preocupar apenas com um conjunto de coordenadas. Elas indicam o centro do círculo, e para localizá-lo basta contar para a direita no eixo-X e para baixo no eixo-Y.

Após ter determinado o ponto, você tem que indicar o raio do círculo. O maior círculo que cabe na tela tem o centro em (129,96) e um raio de 95. Se o raio for maior que 95, o círculo se "achata" próximos às bordas da tela.

É hora de chamar o programa "LINHAS" de volta:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

Elimine a linha 25 e mude a linha 30 para:

```
30 CIRCLE (128,96),95
```

O programa deve ficar assim:

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 CIRCLE(128,96),95
40 GOTO 40
```

Execute o programa. A tela deve apresentar um círculo laranja, um tanto irregular, sobre um fundo bege. Você pensou que o COLOR BASIC EXTENDIDO fosse capaz de fazer um círculo realmente redondo, não? Bem, dê uma olhada na linha 5 do programa e verifique que você está em PMODE 1 (média resolução).

Mude o PMODE na linha 5 para "4" (alta resolução)

```
5  PMODE 4,1
10 PCLS
20 SCREEN 1,1
30 CIRCLE (128,96),95
40 GOTO 40
```

e execute o programa. Agora sim é um círculo! (Deve aparecer um círculo bege sobre um fundo preto).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 5.1

Usando o programa acima, desenhe um alvo. Você pode fazer isso de duas maneiras:

1 - Acrescente uma linha no programa para cada círculo concêntrico, mas use um centro comum (coordenadas x,y).

2 - Use um "loop" FOR... NEXT com STEP 10 para que o computador faça isto por você.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 5.2

Você ainda tem o programa daquela casa? Como é que você vai entrar nela sem uma maçaneta na porta? Use CIRCLE para colocar uma maçaneta na porta da frente. O Gabarito da Tela de Gráficos será muito útil para ajudar a localizar o ponto exato que você quer.

NOTA: Se você usar baixa ou média resolução, um círculo pequeno não terá muitos detalhes. Execute o programa em PMODE 4 e você conseguirá muito mais detalhes.

UM CÍRCULO DE COR DIFERENTE

Depois de determinar o raio do círculo, você pode escolher a cor. É claro que numa modalidade de duas cores você não terá muita escolha, mas numa modalidade de quatro cores (PMODE 1 ou 3) você achará estas opções de cores uma característica muito interessante.

Seu programa está assim?

```
5  PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 CIRCLE (128,96),95
40 GOTO 40
```

Primeiro vamos fazer um círculo de um tamanho razoável.

```
30 CIRCLE (128,96),30
```

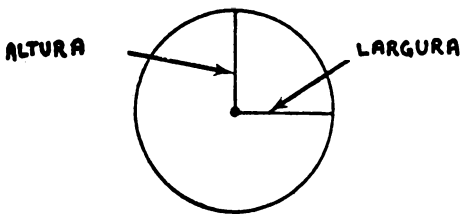
Agora, para variar um pouco, mude a cor para azul esverdeado.

```
30 CIRCLE (128,96),30,6
```

Isto é fácil! Na verdade você pode mudar a cor do círculo para qualquer uma das cores disponíveis.

APERTE O CÍRCULO...

Você já segurou um bambolê, ou pneu de bicicleta com as duas mãos e apertou? Qual a forma que ele tomou? Elíptica, talvez?



Você pode fazer a mesma coisa com o círculo feito pelo computador usando a opção de relação altura/largura(al). É a proporção entre a largura do círculo e a sua altura.

A largura do círculo será sempre igual ao raio especificado . A altura é determinada por al. Se al for maior que 1, o círculo será mais alto do que largo. Se al for menor que 1, o círculo será mais largo do que alto.

Por exemplo:

Repare que omitimos o código da cor. Isso diz ao computador para usar a cor de primeiro plano. Mesmo assim, temos que incluir a vírgula.

```
5  PMODE 4,1
10 PCLS
20 SCREEN 1,1
30 CIRCLE (128,96),30,,1
40 GOTO 40
```

Este programa faz um círculo perfeito, mas se você mudar a1 para:

```
30 CIRCLE (128,96),30,,3
```

o círculo será mais alto do que largo.

e se você mudar a1 para:

```
30 CIRCLE (128,96),30,,.25
```

o círculo será mais largo do que alto.

Quando você usa 0, imagine uma moeda vista de perfil e você terá uma boa idéia a respeito do que queremos dizer.

Se a1 for igual a 0, o círculo se torna "infinitamente" mais largo do que alto. Em outras palavras, o círculo se torna uma linha horizontal. À medida que a1 aumenta a partir de 1, o círculo se aproxima de uma linha vertical.

Mude a linha 30 para as duas formas abaixo e execute o programa:

```
30 CIRCLE (128,96),30,,0
```

Depois:

```
30 CIRCLE (128,96),30,,100
```

DO INÍCIO AO FIM

As últimas opções da instrução CIRCLE permitem desenhar apenas parte de um círculo (um arco).

Para usar esta opção, especifique o ponto onde o arco deve começar (qualquer número de 0 a 1). Coloque uma vírgula e indique onde ele deve terminar (qualquer número de 0 a 1).

O ponto inicial (0) para qualquer ação circular é equivalente a posição de 3 horas em um relógio. Uma vez que o círculo esteja começado, a ação do desenho parte do ponto inicial no sentido dos ponteiros do relógio.

Por exemplo, se você quiser apenas meio círculo, digamos de 6 horas (ponto 0.25) a 12 horas (ponto 0.75), digite:

```
30 CIRCLE (128,96),30,1,1,.25,.75
```

Sempre que o ponto inicial for igual ou maior que o ponto final ou quando o início/fim for omitido, o computador desenha um círculo completo.

NOTA: Para usar as opções início/fim, deve-se especificar a relação altura/largura (al). Para um arco normal, use al = 1.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 5.3

Já caiu a noite na casa que você construiu? Se já, você bem que poderia iluminá-la um pouco, colocando uma lua crescente. Isso vai exigir dois arcos intercruzados e muitas tentativas e erros da sua parte.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 5.4

Talvez não esteja apenas escuro em volta da casa, mas frio também. Então acenda a lareira e faça uma fumaça sair da chaminé. (Use CIRCLE para gerar uma espiral que simule a fumaça).

VERIFICAÇÃO DE APRENDIZADO

Escolha e resposta certa:

1) Qual é a sintaxe correta do comando CIRCLE?

   a - CIRCLE (x,y), r, al, c, inicio, fim
   b - CIRCLE (y,x), r, c, al, inicio, fim
   c - CIRCLE (x,y), r, c, al, inicio, fim
   d - CIRCLE (x,y), r, al, c, fim, inicio

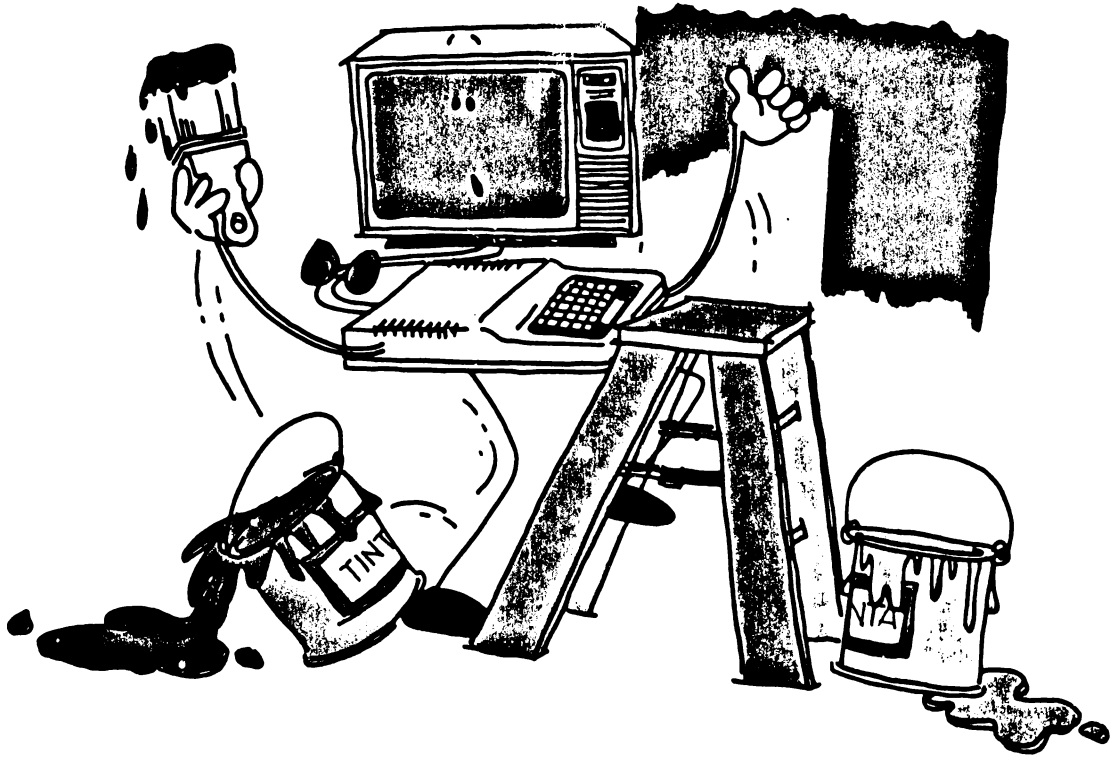
2) Os pontos inicial e final de um arco de círculo são números entre:

   a - 0 - 360
   b - 0 - 10
   c - 0 - 1

3) Se você especificar a relação altura/largura igual a 0, seu círculo será:

   a - Redondo
   b - Elíptico
   c - Uma linha reta

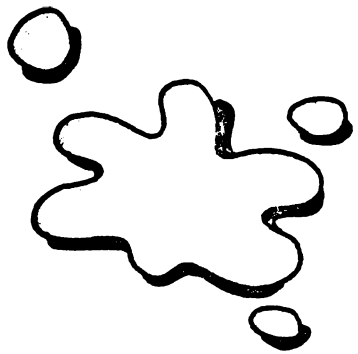
CAPÍTULO 6



UM TOQUE ARTÍSTICO

6

UM TOQUE ARTÍSTICO



Você provavelmente já deve estar pensando que nos esquecemos que este computador é colorido. Até aqui, tudo que tem sido feito é dar uma pinceladinha aqui, um ou dois borrões ali ... assim você nunca poderá criar uma obra-prima.

Bem, é hora de relaxar um pouco e pintar o sete, se não de vermelho, pelo menos de laranja vivo.

A função gráfica PAINT do COLOR BASIC EXTENDIDO permite "pintar" algo de qualquer forma to com qualquer uma das cores disponíveis, e sua sintaxe é a seguinte:

```
PAINT (x,y),c,b

x  determina a coordenada x e é uma expressão numérica de 0 a 255.
y  determina a coordenada y e é uma expressão numérica de 0 a 191.
c  determina o código da cor e é uma expressão numérica de 0 a 8.
b  determina a cor da borda (margem), que delimita a pintura. É uma expressão numéri-
   ca de 0 a 8.
```

Os números dentro dos parênteses especificam onde a pintura deve começar. O primeiro parâmetro depois dos parênteses, "c", especifica o código da cor da pintura.

O parâmetro final, "b", indica a cor da borda que delimita a pintura. Se o computador atinge uma borda de uma outra cor que não a especificada , ele continuará pintando além dela.

Vamos mudar o programa "LINHAS" para:

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 LINE (0,0)-(255,191),PSET
40 LINE (0,191)-(255,0),PSET
50 CIRCLE (128,96),90
60 PAINT (135,125),8,8
70 GOTO 70
```

Antes de executá-lo, você gostaria de arriscar um palpite sobre o que vai acontecer? Se você olhar as linhas 30 e 40, verá as nossas linhas cruzadas.

A linha 50 gera um círculo cujo centro é no ponto onde as duas linhas se cruzam. Bem, até aqui tudo fácil mas o que me diz do PAINT na linha 60?

Se o seu palpite foi que o computador iria para posição (135,125) da tela e usaria a cor laranja até que a pintura atingisse uma borda laranja, você está absolutamente certo.

Elimine a linha 30 e execute-o. Veja como o computador pinta a metade do círculo depois das bordas serem redefinidas.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 6.1

Você pode pintar um círculo inteiro? Existem duas maneiras de se fazer isso: Uma seria acrescentando uma linha e a outra seria eliminando uma linha.

Lembre-se: você só pode pintar com as cores disponíveis no conjunto de cores do PMODE em uso.

Mas você não especificou linhas vermelhas e pintura vermelha, não é? Você tem alguma idéia a respeito do que aconteceu?

Quando o computador está numa modalidade de quatro cores e você especifica uma cor não disponível, ele subtrai 4 dos códigos 5 a 8 (zero é interpretado como 3).

A propósito, você reparou em que PMODE e em que conjunto de cores você estava? O PMODE 3 é uma modalidade de quatro cores e o conjunto de cores 1 dá as cores bege, azul esverdeado, carmim e laranja.

Continue em PMODE 3, mudando porém o conjunto de cores (SCREEN 1,0) e execute o programa.

Sem haver mudado qualquer outra linha, você deve ter conseguido um círculo vermelho (borda) sobre um fundo verde.

Para evitar confusão a respeito da cor, mude a cor do PAINT para ajustá-lo ao conjunto de cores que você está usando.

```
60 PAINT (135,125),2,4
```

Agora, quando você executar o programa, o semi-círculo deverá ser pintado de amarelo (código 2) até que o computador chegue à borda vermelha (código 4).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 6.2

Você ainda tem o programa da "casa"? Talvez ela esteja parecendo muito modesta, então que tal enfeitá-la um pouco com alguma pintura?

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 6.3

Aqui temos uma variação na instrução PAINT. Adicione uma garagem e use PAINT para levantar e abaixar a porta. Isso exigirá algum esforço da sua parte.

Acrescente um período de espera antes e depois de abrir a porta. Aproveite e coloque também um sol na tela usando CIRCLE.

VERIFICAÇÃO DE APRENDIZADO

1) Onde está o erro nesta instrução? PAINT (128,96), 8, 2

2) Diga se são falsas ou verdadeiras as afirmações abaixo:

a - PAINT pode usar todas as oito cores ao mesmo tempo.

b - A menos que você especifique uma borda em particular, o PAINT se deterá na primeira borda que alcançar.

CAPÍTULO 7

DESENHE A LINHA

EM QUALQUER LUGAR

7

DESENHE UMA LINHA EM QUALQUER LUGAR

Até agora mostramos a vocês como criar linhas, círculos e quadrados. Mostramos até mesmo como preenchê-los ou pintá-los, se necessário. Uma coisa que ainda não fizemos, porém, é mostrar-lhes como "desenhar" uma linha, um quadrado ou um triângulo.

Sim, acredite ou não, existe uma maneira mais fácil de fazer algumas das coisas que mostramos a vocês. Esta "maneira mais fácil" chama-se DRAW e nos permite desenhar uma linha (ou uma série de linhas) especificando sua direção, ângulo e cor — tudo no mesmo comando do programa.

```
DRAW  linha

linha     é uma expressão string e pode incluir:
          Comandos de movimento
          M = Move a posição de desenho
          U = Para cima
          D = Para baixo
          L = Para a esquerda
```

```
 5 PMODE 3,1
10 PCLS
20 SCREEN 1,1
25 DRAW "BM128,96;U25;R25
   ;D25;L25"
40 GOTO 40
```

Separamos cada instrução de movimento com um ponto e vírgula (;) mas você não tem que necessariamente fazer o mesmo — isso simplesmente facilita a leitura do programa. Você deve, no entanto, separar sempre as coordenadas (X, Y) com uma vírgula (,).

R = Para a direita
E = Ângulo de 45 graus
F = Ângulo de 135 graus
G = Ângulo de 225 graus
H = Ângulo de 315 graus
X = Executa um substring e retorna

Modalidades
C = Cor
A = Ângulo
S = Escala

Opções
N = Nenhuma atualização da posição DRAW
B = Em branco (não desenha, apenas move)

NOTA: Se "linha" for uma constante tipo STRING, ela deve ser colocada entre aspas. Insira sempre a opção B imediatamente antes do comando de movimento M; caso contrário podem aparecer linhas indesejáveis.

No início deste manual mostramos como criar um quadrado usando LINE. Você teve que dar os pontos de partida e fim desejados e o computador se incumbiu de ligá-los. Para isto você precisou de apenas uma linha de programa, mas provavelmente teve dificuldades em determinar no Gabarito da Tela de Gráficos o quadrado que queria.

Com DRAW, você tem apenas que dar o ponto de partida e dizer ao computador em qual direção, e até onde a linha deve ir; ou então, pode omitir o ponto de início e o computador

começará na última posição DRAW (o centro da tela, caso você não tenha usado anteriormente a instrução DRAW no programa).

Vamos experimentar, usando o velho e fiel programa "LINHAS".

Cancele as linhas 25 e 30 do programa e inclua a seguinte:

```
25 DRAW "BM128,96;U25;R25;D25;L25"
```

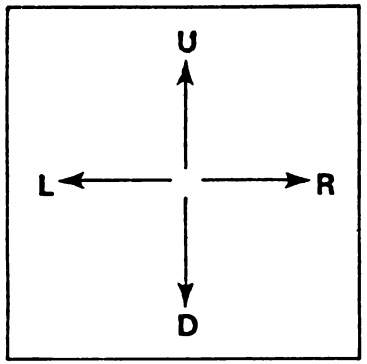
Pronto! Você pode imaginar porque o canto inferior esquerdo do quadrado está em (128,96)? Observe cuidadosamente os dois primeiros caracteres entre aspas.

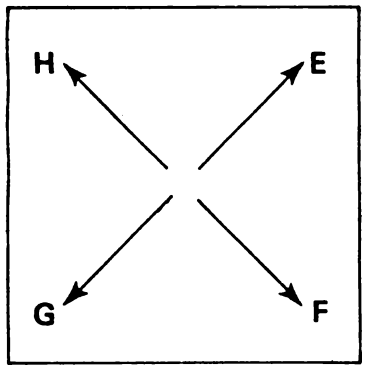
O comando de movimento M posiciona o primeiro ponto do desenho em um local especificado.

Mx,y
x é a coordenada x e é uma expressão numérica de 0 a 255.
y é a coordenada y e é uma expressão numérica de 0 a 191.
NOTA: M deve sempre ser precedido pela letra B, ou aparecerão linhas indesejáveis.

No programa acima, instruímos o computador para começar o desenho em (128,96), desenhar 25 pontos para cima (U), mais 25 pontos para a direita (R), mais 25 pontos para baixo (D) e, finalmente, mais 25 pontos para a esquerda (L).

NOTA: Se você não especificar o comprimento da linha, o computador usará "1".





LINHAS DIAGONAIS

Em vez de desenhar linhas horizontais e verticais, desenhe um quadrado inclinado. Para fazer isto, coloque E, F, G e H no lugar de U, R, L e D na linha 25 do programa:

```
25 DRAW "BM128,96;E25;F25;G25;H25"
```

Este DRAW também começa em (128,96). Porém, em vez de ir em linha reta para cima, a primeira linha é traçada a 45 graus e as próximas 3 linhas são desenhadas nos ângulos designados.

Se você está em PMODE 0 ou 1 e usa E, F, G ou H para gerar uma linha que tenha de comprimento um número ímpar e pelo menos um número ímpar como coordenada (x,y), as linhas F e H sofrerão um ligeiro desvio no ponto médio; se ambas as coordenadas (x,y) forem números pares, as linhas E e G sofrerão este desvio. Isto é normal.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.1

Você já sabe que seu computador é a estrela do show, mas você pode provar isso "desenhando" uma estrela? Use o comando de movimento DRAW tanto para as linhas perpendiculares como para as diagonais.

MOVIMENTO ABSOLUTO X MOVIMENTO RELATIVO

Movimento Absoluto: "Vá até a esquina da Av. Rio Branco com a rua 7 de Setembro".
Movimento Relativo: " Siga duas quadras abaixo e dobre à direita".

Suponhamos que você tenha desenhado um quadrado mas queira desenhar um outro perto do primeiro. O problema é que você sabe exatamente a que distância você quer que o quadrado fique, mas não quer ter que localizar as coordenadas (x,y).

Existe uma outra forma de comando M que permite especificar movimento "relativo" em vez de "absoluto". Até aqui trabalhamos com movimento absoluto o que significa que tivemos que especificar pontos em termos de suas coordenadas reais (x,y).

Com o comando relativo, nós especificamos pontos em relação ao ponto atual (último a ser desenhado).

Aqui está um resumo para movimento relativo:

M sinal deslocamento - x, sinal deslocamento - y	
deslocamento - x	é um número que especifica a distância a ser movida a partir da posição x atual. Se o deslocamento-x for precedido por um sinal "+", a posição x será aumentada do valor especificado. Se o deslocamento-x for precedido por um sinal "-", a posição x será diminuída do valor especificado. O sinal "+" ou o sinal "-" devem ser usados já que é o seu uso que determina ao computador que use o movimento relativo e não o absoluto.

deslocamento - y	é um número que especifica a distância a ser movida a partir da posição y atual. O sinal do número determina se a posição atual deverá ser aumentada (+) ou diminuída (-). Para deslocamentos positivos, o sinal pode ser omitido apenas no deslocamento-y.

Por exemplo, se quiser criar um outro quadrado numa posição relativa à posição do quadrado original do seu programa "LINHAS"(redefinido), você pode acrescentar esta linha:

```
30 DRAW "BM+15,15;U25;R25;D25;L25"
```

Quando o computador executa a linha 30, a posição do desenho atual está em (128,96)que é a última posição de desenho na linha 25. Portanto, o canto esquerdo inferior do novo quadrado está localizado em (X=128+15=143, Y=96+15=111)

Se você mudar a linha 30 para

```
30 DRAW "BM+15,-15;U25;R25;D25;L25"
```

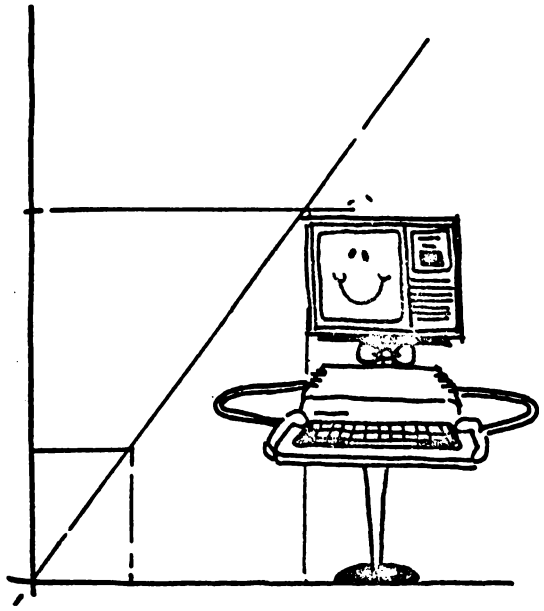
e executar o programa, o ponto de partida do novo quadrado ficará em (X=128+15=143 , Y=96-15=81).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.2

Depois de toda esta atividade, você provavelmente já está pronto para uma "esfriada". Então porque não usar DRAW para criar um cubo de gelo?

O cubo inteiro pode ser desenhado com DRAW, ou você pode querer incorporar um par de comandos LINE dentro do programa. Tente usar ambos, movimento absoluto e relativo.

É HORA DE USAR AS ESCALAS



Você desenhou linhas, quadrados e cubos. Mas e se os quadrados ficarem muito grandes ou muito pequenos?

Bem, seu computador tem uma função que permite reduzir ou ampliar qualquer representação visual gerada por DRAW. Tudo o que você tem a fazer é inserir o seguinte no string:

Sx

x é um número de 1 a 62 que indica a escala adotada no desenho em unidades de 1/4.

1 = escala 1/4	5 = escala 5/4 (125%)
2 = escala 2/4	8 = escala 8/4 (dupla)
3 = escala 3/4	12 = escala 12/4 (tripla)
4 = escala 4/4 (unitária)	etc.

NOTA: Se você não especificar um número para S, o computador assume S4 (4/4=1).

Quando usar a opção de redução do desenho, se a linha resultante não foi um número inteiro, o computador arredondará o comprimento da linha para o número inteiro mais próximo. Por exemplo, "S2U25R25D25 L25" resultaria em um quadrado de 12,5X12,5. O computador desenha um quadrado de 13X13.

Você não tem usado muito os string e funções de strings neste manual. Para uma rápida revisão de STR$, veja o Manual do COLOR BASIC. Veremos strings novamente na seção II deste Manual.

Depois de um comando Sx, todos os comandos de movimento absolutos e relativos, serão gerados nesta escala até a ocorrência do próximo comando Sx.

Sim, é hora de usar o programa "LINHAS" novamente, mas vamos fazer um único quadrado de novo. Cancele a linha 30 do programa e mude a linha 25;

```
25 DRAW "S2;BM128;9G;U25;R25;D25;L25"
```

Execute o programa. Você deverá ter um quadrado com o canto inferior esquerdo em (128,96) porém da metade do tamanho que você especificou.

Execute o seguinte programa para ver como um pequeno quadrado pode ser reduzido e ampliado para preencher a tela por completo.

```
5  PMODE 4,1
10 PCLS
20 SCREEN 1,1
25 FOR SCALE = 1 TO 62
30 S$= "S" + STR$(SCALE) + ";"
35 DRAW S$ + "BM10,100U20R20D20L20"
40 NEXT SCALE
50 GOTO 50
```

Não cometa o erro de pensar que o menor quadrado é o que foi especificado na linha 35. O que nós especificamos é o quarto a partir do menor.

PINTE-ME

A opção C da instrução DRAW permite especificar a cor de uma linha em particular.

Primeiro façamos uma atualização da listagem do programa "LINHAS":

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 DRAW "S2;BM128,96;U25;R25;D25;L25"
```

Volte para a escala de 4/4 (completa) — mude S2 para S4 ou cancele S2 — e insira o seguinte, imediatamente após o primeiro par de aspas na linha 30:

```
C6
```

Execute o programa. Conseguiu um quadrado azul esverdeado num fundo amarelado?

Agora, substitua o C6, na linha 30 do programa, por C8 e execute-o. O quadrado ficou laranja?

C tem que ter a seguinte sintaxe:

Cx	
x	é uma expressão numérica de 0 a 8 e determina um dos nove códigos de cores. É opcional e, se omitido, a cor do primeiro plano será usada.

Seu programa deveria ficar assim:

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 DRAW "C6;BM128,96;U25
   ;R25;D25;L25"
40 GOTO 40
```

Se você quiser apagar uma linha, desenhe uma outra linha em cima desta usando a cor do fundo.

Você pode inserir Cx em qualquer lugar dentro da instrução DRAW e todas as ações que se seguem serão da cor que você especificar. Por exemplo, mude a linha 30 do programa para:

```
30 DRAW "C8;BM128,96;U25;R25;C6;D25;L25"
```

Execute o programa. Você deverá ter um quadrado de duas cores em sua tela. As duas primeiras linhas devem ser laranja, mas quando a linha começar a descer, por exemplo, antes de D25, deverá tornar-se azul esverdeada.

QUAL É O ÂNGULO ?

Outra opção que pode ser utilizada no DRAW é A. "A" permite especificar o ângulo no qual a linha deve ser desenhada. Depois que a opção for incluída no comando DRAW, todas as linhas subsequentes serão desenhadas com um deslocamento angular especificado por Ax, cuja sintaxe é:

Ax

x é uma expressão numérica de 0 a 3 e determina um dos ângulos a seguir:

0 = 0 graus
1 = 90 graus (no sentido dos ponteiros do relógio)
2 = 180 graus (no sentido dos ponteiros do relógio)
3 = 270 graus (no sentido dos ponteiros do relógio)

NOTA: Se você não especificar um ângulo, o computador usará A0.

Para melhor ilustrar isso, mude a linha 30 do programa para:

```
30 DRAW "A0;BM128,96;U25"
```

Execute o programa. A tela deverá mostrar uma linha (com um tamanho de 25 pontos) indo direto para cima. Agora mude a linha para:

```
30 DRAW "A1;BM128,96;U25"
```

Execute o programa. A linha agora deverá estar apontando para a direita. A diferença no comprimento da linha é porque os pontos em PMODE 3 são retângulos mais altos do que largos. Portanto, as linhas horizontais são ligeiramente mais longas que as linhas verti - cais.

ATIRANDO NO "NADA"

Suponhamos que você queira fazer um desenho em branco ou uma linha invisível. Você pode fazer isso com a opção B que determina que a linha a seguir (apenas a linha a seguir) deverá estar em branco.

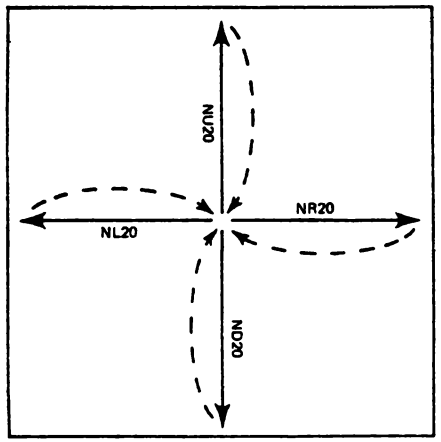
Por exemplo, digamos que você está desenhando as letras do alfabeto e vai desenhar a letra "C". Ela não é nada mais que um quadrado com um lado em branco.

Mude seu programa para que ele gere um quadrado, mas deixe o lado direito, gerado por D25, em branco:

```
30 DRAW "BM128,96;U25;R25;B;D25;L25"
```

Execute, então, o programa. Está vendo? Mas lembre-se, apenas a linha imediatamente após o B estará em branco.

> FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.3
>
> Imprima seu nome na tela usando DRAW. Isto significa que você deverá permanecer na tela de gráficos. Claro, seria mais fácil escrever seu nome na tela de texto, mas você não pode ter texto e gráfico ao mesmo tempo.



O QUE ? MAIS OPÇÕES ?

Outra das muitas características do DRAW é a opção N de "não atualização". "N" diz ao computador para retornar à posição inicial após desenhar a linha seguinte (linha 2).

Vamos usar esta opção numa linha do programa e então discutir isso. Mude a linha 30 do programa para:

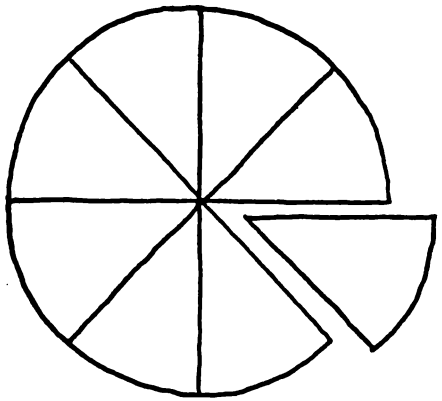
```
30 DRAW "M128,96;N;U25;N;R25;N;D25;N;L25;"
```

Execute o programa. Deverão aparecer quatro linhas partindo do centro da tela em quatro direções diferentes. Para cima, para a direita, para baixo e para a esquerda.

O que aconteceu foi que você disse ao computador para desenhar uma linha começando em

(128,96) e que fosse 25 pontos para cima, mas que retornasse a sua posição inicial antes de começar uma outra linha.

O computador então desenhou uma linha com 25 pontos para a direita e retornou à posição original. A seguir, ele desenhou uma linha de 25 pontos para baixo e novamente retornou à sua posição original. Finalmente, o computador desenhou uma linha de 25 pontos para a esquerda e, novamente, voltou à posição original.



FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.4

Usando a opção N do DRAW (e CIRCLE) faça o computador desenhar uma torta com oito pedaços. Após fazer isso, corte um pedaço da torta e separe-o para um lado.

CONSTANTES STRING X VARIÁVEIS STRING

Você provavelmente deve ter pensado que nós o estávamos "enrolando" quando dissemos que o string após o DRAW tanto poderia ser uma constante (o que temos usado em nossos exemplos) como uma variável.

O sinal dolar ($) deve ser sempre seguido por um ponto e vírgula, mesmo que os outros pontos e vírgulas não sejam necessários: XA$; XX$; XC$.

Se você desejar usar uma variável string, simplesmente identifique-a como tal em uma linha precedente à DRAW e use-a no lugar do que está entre aspas, na instrução DRAW. Por exemplo, mude seu programa para

```
25 A$="BM128,96;C8;U25;R25;D25;L25"
30 DRAW A$
```

e execute o programa. O programa deverá gerar um quadrado laranja (25 X 25) cuja extremidade inferior esquerda estará no centro da tela.

Entretanto, o COLOR BASIC EXTENDIDO oferece uma variação disto. Chamamos a isto de ação de execução (X). Enquanto você estiver executando a rotina DRAW, a ação de execução permite executar um outro string DRAW, e então retornar e completar a primeira operação. Mude a linha 30 do programa para:

```
30 DRAW "BM95;50;U25;R25;XA$;D25;L25"
```

Execute o programa. O computador deverá começar com uma linha em (95,50) que sobe, vira à direita e desenha uma outra linha. Neste ponto, o A$ será executado e será desenhado um quadrado de 25 X 25 começando em (128,96). Após executar o A$, o computador retorna ao string original e completa a sua execução.

NOTA: A posição de desenho não retorna à sua posição anterior (por exemplo, o primeiro quadrado), mas continua onde estava após ter completado o A$.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.5

Como você viu, caso tenha completado o "FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.3", você pode simular texto (letras) na tela de gráficos, desenhando as letras. Use DRAW para criar todas as 26 letras do alfabeto. Armazene o comando de desenho em strings. Use então a opção "execute" arrumando as letras para imprimir mensagens ou palavras.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 7.6

Você ainda tem aquela casa? Se tem, retire-a de seu arquivo. Você agora provavelmente já pode ver como o DRAW pode facilitar suas construções.

Sua principal tarefa neste exercício é fechar e abrir a porta da frente. Para isto, você empregará quase todas as características das quais falamos até aqui.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) (Mx,y) (M deslocamento-x, deslocamento-y) geram movimentos absolutos.

2) Caso você não especifique onde o DRAW deve começar, o computador começará:

   a - No canto inferior esquerdo
   b - Em (255,191)
   c - Na posição atual (última)
   d - Em (75,163)

3) Para evitar linhas indesejadas, a instrução DRAW deve ser sempre precedida de: .....

4) Qual dos comandos à seguir desenha em um ângulo de 45 graus?

   a - U
   b - A1
   c - E
   d - H2

5) Para executar um substring, usa-se:

a - F
b - A
c - G
d - X

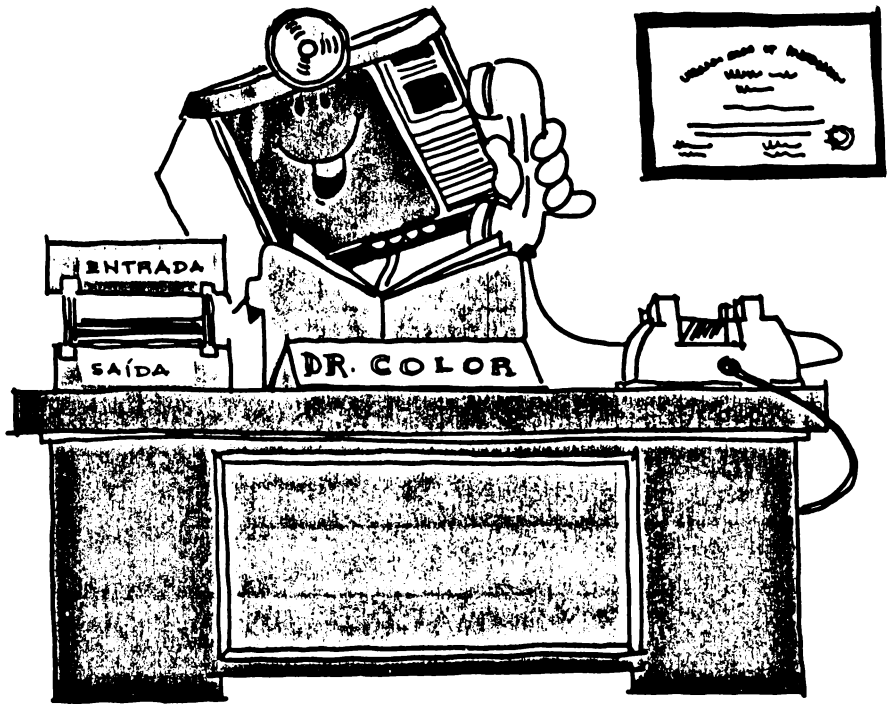
6) Com DRAW, você pode especificar a cor de uma linha (Falso ou Verdadeiro).

7) Qual dos comandos a seguir é uma escala unitária (4/4)?

a - S1
b - S2
c - S4
d - S10

NOTAS

CAPÍTULO 8



COMO SIMULAR MOVIMENTOS

8

COMO SIMULAR MOVIMENTOS

Quem? O que? Falamos sobre matrizes no Manual do COLOR BASIC portanto, se estiver um pouquinho esquecido, volte e faça uma pequena revisão antes de continuar.

Mesmo que o computador venha fazendo verdadeiras "mágicas" com o programa "LINHAS", você pode achar que ele ainda não está apto para o que der e vier. Porém, você provavelmente mudará de opinião quando comprovar o que o computador faz com "GET" e "PUT".

Estes comandos permitem pegar (GET) uma área retangular contendo um gráfico, armazená-la uma matriz e depois colocar (PUT) a matriz de volta na tela.

Quando se trata de simular movimento, GET e PUT podem mover objetos mais rapidamente do que qualquer outro método.

As sintaxes para GET e PUT são:

GET ponto inicial - ponto final, destino, G	
ponto inicial	é a coordenada (x1,y1) do canto superior esquerdo do retângulo do gráfico.
ponto final	é a coordenada (x2,y2) do canto inferior direito do mesmo retânulo.
destino	é o nome de uma matriz pré-definida que armazenará o conteúdo do retângulo.
G	indica que é para armazenar o conteúdo do retângulo com todos os detalhes gráficos. É opcional em alguns casos.

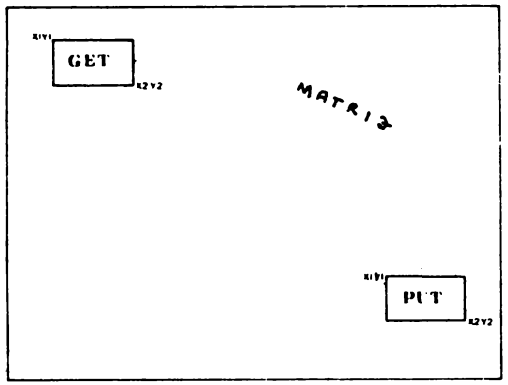
PUT ponto inicial- ponto final, fonte, ação.

ponto inicial - ponto final	conforme explicados anteriormente.
fonte	é o nome de uma matriz pré-definida que contém os dados a serem escritos no retângulo a ser mostrado na tela.
ação	determina como os dados serão escritos no retângulo.
PSET	ativa cada ponto que está armazenado na matriz fonte.
PRESET	apaga cada ponto que está armazenado na matriz fonte.
AND,OR,NOT	Veja a seguir.

ação é opcional em alguns casos.

NOTA IMPORTANTE: Certifique-se de estar usando o mesmo PMODE para GET e PUT. Caso contrário você não conseguirá mostrar (PUT) o retângulo armazenado pela instrução GET.

Este programa lhe dará uma idéia do que fazem GET e PUT. Mais tarde explicaremos como funcionam estas funções. Digite e execute:




```
5  PCLEAR 4
10 PMODE 3,1
15 PCLS
20 SCREEN 1,1
25 DIM V(20,20)
30 CIRCLE (20,20),10
35 GET (10,10)-(30,30),V
```

```
40 PCLS
42 FOR  DLAY = 1 TO 300: NEXT  DLAY
45 PUT (110,110)-(130,130),V
50 FOR  DLAY = 1 TO 300: NEXT  DLAY
60 GOTO 60
```

Deverá aparecer um círculo laranja na parte superior esquerda da tela (posição (10,10) - linha 30 do programa). Depois de um ou dois segundos, o círculo deverá desaparecer e, então, de repente, reaparecerá em algum outro lugar na tela (posição (110,110) – linha 45 do programa).

O que o computador fez foi:

- Criar uma matriz (linha 25)
- Criar um círculo (linha 30)
- Armazenar o círculo que estava dentro do retângulo fonte, especificado na instrução GET, numa matriz (linha 35).
- Limpar a tela (linha 40)
- Colocar o círculo da matriz no retângulo de destino, especificado na instrução PUT (linha 45)

A ilustração a seguir deve ajudá-lo a entender exatamente o que aconteceu.

ANATOMIA DE UM RETÂNGULO

A posição e o tamanho do retângulo da tela são determinados pelas duas coordenadas (x,y) , usadas em GET/PUT.

POSIÇÃO	Canto superior esquerdo = ponto inicial = (x1,y1) Canto inferior direito = ponto final = (x2,y2)
TAMANHO	Largura = x2 - x1 Comprimento = y2 - y1

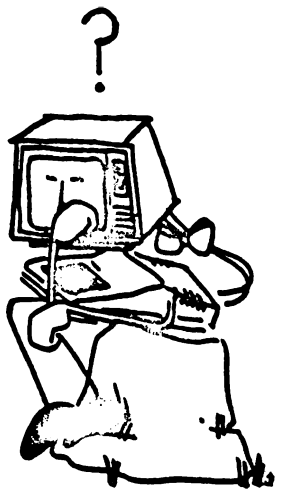
ESCOLHENDO A MATRIZ CERTA

Note bem que antes de usarmos as instruções GET ou PUT temos que criar uma matriz bi-dimensional para armazenar o retângulo da tela (linha 25). Como podemos determinar o tamanho de cada matriz? O tamanho de cada matriz deve ser comparável ao do retângulo da tela.

Primeira dimensão da matriz = largura do retângulo
Segunda dimensão da matriz = comprimento do retângulo

Por exemplo, um retângulo de 40 X 20 necessita de uma matriz de 40 X 20. Já que a matriz do COLOR BASIC tem o elemento zero, a instrução de dimensão a seguir cria uma matriz adequada:

DIM V(39,19)



DE QUE TAMANHO PODE SER O SEU RETÂNGULO ?

O tamanho do retângulo é limitado pela quantidade de memória disponível para uso no programa. Cada elemento da matriz de armazenamento consome cinco posições de memória (bytes) , e é óbvio que se o seu programa for muito longo, você não terá muito espaço para a matriz .

Naturalmente o retângulo deve ter também tamanho suficiente para conter os gráficos e, sua posição na tela, deve ser escolhida de modo que inclua todos os pontos que você quer armazenar.

Se você usar a opção G da instrução GET, é preciso usar uma das opções disponíveis para o PUT (PSET, PRESET, AND, OR, NOT), caso contrário, aparecerá algum "lixo" quando você colocar o retângulo de volta na tela. Nem sempre é necessário usar as opções.

- Não use as opções em PMODE 0, 1 ou 3
- Use as opções em PMODE 4 ou 2

As opções para o PUT desempenham funções de acordo com a tabela a seguir:

OPÇÃO	FUNÇÃO
PSET	Ativa os pontos que estavam acesos no retângulo original.
PRESET	Desativa os pontos que estavam acesos no retângulo original.
AND	Um operador lógico. Compara os pontos armazenados no retângulo original com os do retângulo de destino. Se ambos estiverem ativados, então o ponto da tela será ativado; senão, o ponto da tela será apagado.
OR	Operador lógico. Compara os pontos (como acima). Se pelo menos um estiver ativado, o ponto da tela permanecerá ativado.
NOT	Operador lógico. Inverte o estado de cada ponto no retângulo de destino independentemente do conteúdo da matriz PUT.

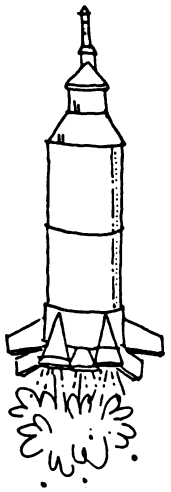
ESPALHE-SE

Veja como o programa a seguir usa GET e PUT para simular movimento. Para dar velocidade à ação, você pode aumentar o incremento do STEP. Preste muita atenção na maneira como o programa usa as funções que você já estudou (LINE, CIRCLE, PAINT, etc).

```
5  PCLEAR 4
10 DIM V(30,30)
15 PMODE 3,1
20 PCLS
25 SCREEN 1,1
30 CIRCLE (128,96),30
35 PAINT (128,95),2,4
```

```
40 PAINT (128,97),3,4
45 GET (98,81)-(128,111),V,G
50 PCLS
55 FOR I = 150 TO 1 STEP -1
60 PUT (I,81-I/5)-(I+60,111-I/5),V,PSET
65 NEXT I
70 GOTO 70
```

O único truque que usamos foi mudar (atualizar) a posição do retângulo do PUT pelo valor de I.



FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 8.1

Se lembra do foguete do qual falamos no capítulo 3? Use GET e PUT para fazer uma nave espacial atravessar os limites superiores da tela. Você talvez queira acrescentar alguns obstáculos (um ou dois marcianos, por exemplo) para a nossa nave "Entreprise" se desviar.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Se usar a opção G com GET, o que deve ser usado com PUT?

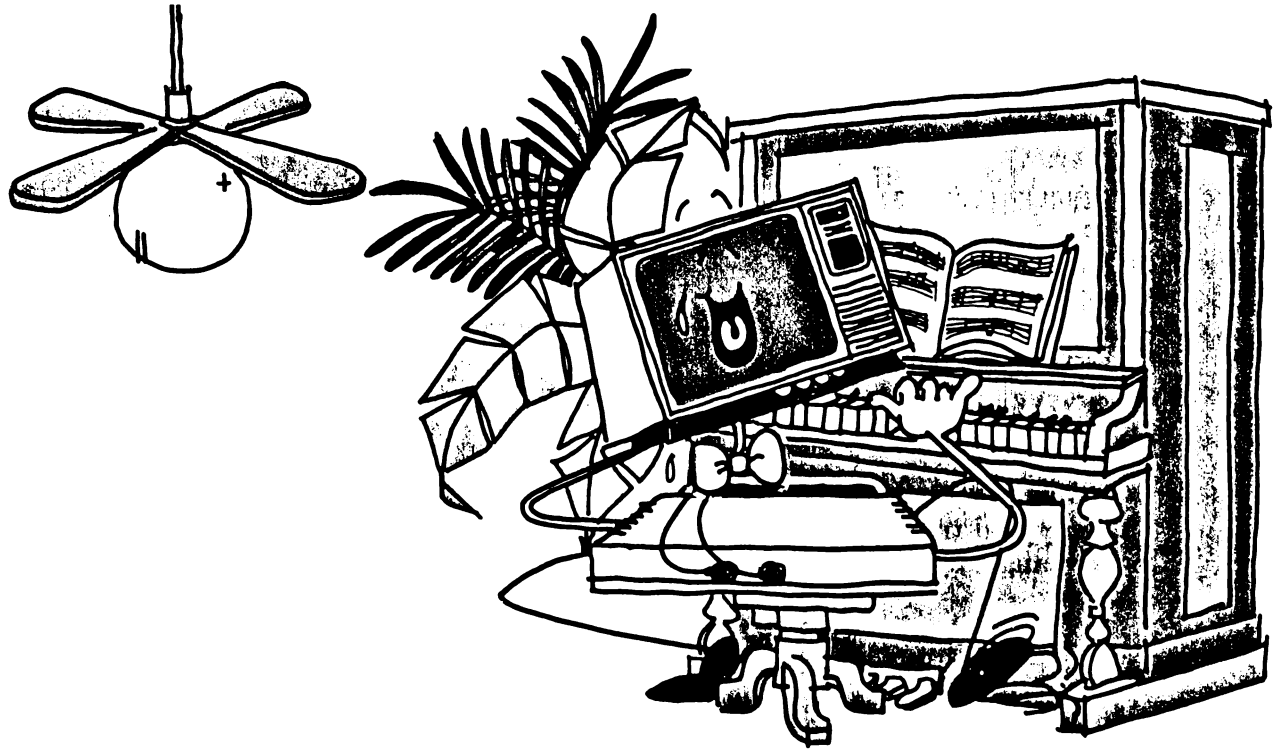
   a - PSET
   b - PRESET
   c - NOT
   d - Qualquer das respostas anteriores
   e - Nenhuma das respostas anteriores

2) GET armazena gráficos em uma:

   a - Tela
   b - Matriz
   c - Página de memória

NOTAS

CAPÍTULO 9



TOCA MAIS UMA...

9

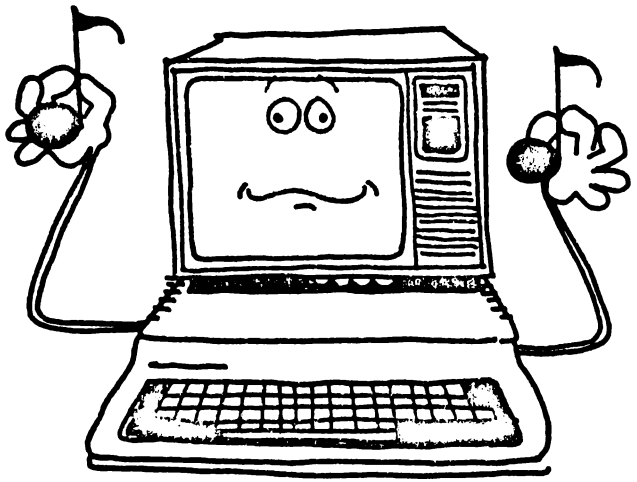
TOCA MAIS UMA . . .

Seus olhos não estão ficando meio embaçados? Você não anda sonhando com pontos, linhas, quadrados, triângulos, círculos e conjuntos de cores?

Bem, talvez seja hora de dar um descanso aos seus olhos e sacrificar os seus ouvidos com a função PLAY do COLOR BASIC EXTENDIDO. Essa função permite não só tocar músicas, como também compor.

Antes de irmos adiante talvez seja melhor explicarmos que PLAY não é uma função gráfica. Consequentemente, não será necessário começar os programas com PMODE, PCLS e SCREEN. Bem, seus olhos provavelmente já devem estar muito cansados, portanto, vamos fazer menos barulho e mais música.

OUÇA COM ATENÇÃO



PLAY música

música	é uma expressão string que especifica:
Nota	uma letra de "A" a "G" ou um número de 1 a 12.
Oitava	"O" seguido por um número de 1 a 5. Se omitida, a oitava "2" será usada.
Duração da Nota	"L" seguido por um número de 1 a 255. Se omitida, será usada a duração em curso.
Andamento	"T" seguido por um número de 1 a 255. Se omitido, T2 será usado.
Volume	"V" seguido por um número de 1 a 31. Se omitido, V15 será usado.
Duração da Pausa	"P" seguido por um número de 1 a 255.
Execução de Substrings	substrings devem ser precedidos pelo prefixo "X" seguido por um ponto e vírgula, por exemplo, XA$;

VAMOS ACOMPANHAR AS NOTAS

É óbvio que você não pode ter música sem notas musicais. O comando PLAY dá duas maneiras de determinar a nota exata que você precisa.

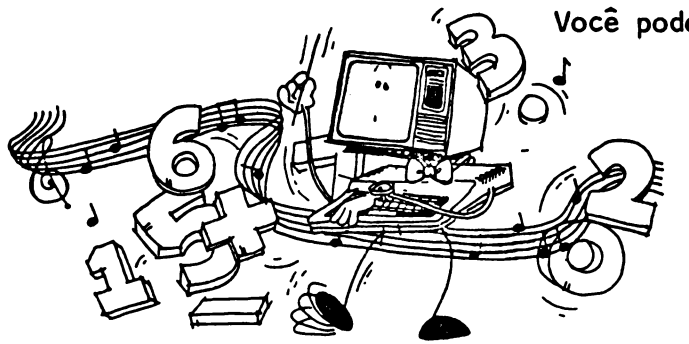
A primeira, e provavelmente a mais fácil maneira de tocar a nota que você quer é entrando com uma das notas musicais padrão — "A,B,C,D,E,F ou G". Para indicar um "sustenido" ou um "bemol", use "+" ou "#" para sustenidos e "-" para bemóis.

Por exemplo, "A" representa A natural, "A#" representa A sustenido e "A-" representa A bemol. Digite o seguinte exemplo para ver(ou ouvir?) o que queremos dizer:

PLAY "A" (ENTER)

Para ouvir a alteração causada por um sustenido e um bemol, digite:

PLAY "A;A#" (ENTER)
PLAY "A-;A;A#;A;A-" (ENTER)



Você pode fazer a mesma coisa com todas as sete notas (A-G) da escala.

NOTA: Existe apenas um semitom separando B e C ; E e F. Portanto, B#=C, C-=B, E#=F e F-=E .

UMA NOVA "NOTA" - AÇÃO

Outra maneira de se especificar uma nota musical é usar um número entre 1 e 12, precedido pela letra N. O N é opcional, e se for omitido, o número sozinho indicará a nota.

Os números 1 a 12 representam cada uma das notas da escala musical incluindo todos os sustenidos e bemóis. Esta é uma notação mais resumida, embora seja mais difícil que a notação padrão.

NOTA: O comando PLAY não reconhece a notação "B#" ou "C-". Use os números 1 e 12 respectivamente ou substitua "B#" por "C" e "C-" por "B". Você conseguirá a nota que deseja já que B# = C e C- = B.

Eis aqui um pequeno programa que toca notas, depois de digitar uma letra ou um número e pressionar a tecla (ENTER). Com este programa, você pode tocar uma única nota ou uma canção inteira (até 249 caracteres).

```
5  CLS
10 INPUT "DIGITE UMA NOTA
   (A-G,#,-) OU (1-12)  E
   PRESSIONE <ENTER> ";
   A$
20 PLAY A$
30 GOTO 5
```

Execute o programa a seguir para ouvir a escala completa de 12 tons. De agora em diante, chamaremos este programa de "ESCALA".

```
5  CLS
10 FOR N=1 TO 12 'N=NOTA
15 PRINT "NOTA ";N
20 PLAY STR$(N)
30 NEXT N
```

Acrescente uma pausa no programa para poder comparar os números com as notas à medida em que a escala sobe de 1 a 12 (C a B).

```
25 FOR I=1 TO 500: NEXT I
```

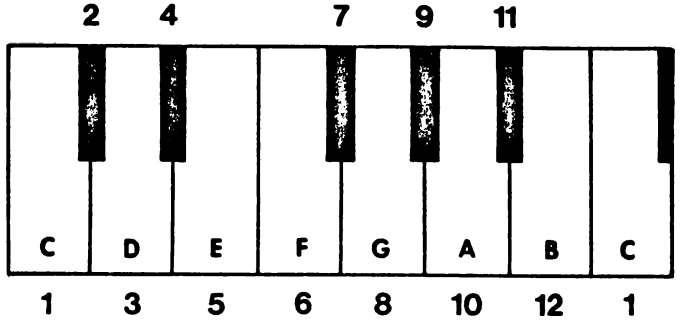


TABELA DE NÚMEROS/NOTA MUSICAL		
NÚMERO	NOTA	
1	C	(DÓ)
2	C#/D-	(DÓ#/RÉb)
3	D	(RÉ)
4	D#/E-	(RÉ#/MIb)
5	E/F-	(MI/FAb)
6	F/E#	(FA/MI#)
7	F#/G-	(FA#/SOLb)
8	G	(SOL)
9	G#/A-	(SOL#/LÁb)
10	A	(LÁ)
11	A#/B-	(LÁ#/SIb)
12	B	(SI)
NOTA: "b" = bemol		

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 9.1

Modifique o programa "ESCALA" para que ele execute a escala descendente, em vez de ascendente.

SEMIBREVES, MÍNIMAS, SEMÍNIMAS...
(DURAÇÃO DAS NOTAS)

As notas no programa "ESCALAS" eram notas semínimas. Como não especificamos a duração da nota (isto é, se era uma semibreve, mínima ou semínima, etc) o computador automaticamente usou notas semínimas que é o valor inicial.

Para escolher a duração da nota, use L seguido de um número entre 1 e 255. O número 1 , por exemplo, representa uma semibreve, 2, uma mínima, 4, uma semínima, 8, uma colcheia, 16, uma semicolcheia, etc.

Na verdade, você pode usar qualquer número de 1 a 255 (quem já ouviu falar de 1/15 de nota?)

Vamos variar a duração das notas para produzir o rufar de um tambor. Digite esta linha e aperte (ENTER).

```
PLAY "L2;A;L4;A;A;L2;A;A"
```

Você cronometrou a duração das notas para ter certeza que elas são quatro vezes mais longas? Isto não é realmente necessário já que o computador possui um relógio interno que faz isso por você.

TABELA DE DURAÇÃO DE NOTAS	
L - número	Duração da nota
L1	Semibreve
L2	Mínima
L3	Quiáltera de 3
L4	Semínima
L8	Colcheia
L16	Semicolcheia
L32	Fusa
L64	Semifusa
.	-
.	-
.	-
L255	1/255 de semibreve

L2 indica uma mínima, L4, uma semínima. Portanto, nós tocamos assim: "mínima, semínima , semínima, mínima, mínima".

PLAY "L1; A, A#; A-" (ENTER)

Repare que há mais alguma coisa sobre a opção L: ela não precisa ser repetida para toda nota que for tocada. PLAY usa o último valor da nota até que você dê uma instrução modificando o comando N. Foi por isso que não tivemos que repetir L4 no último exemplo.

Só para ver, tente tocar três notas A com duração 1/255.

PLAY "L255;A;A+;A-" (ENTER)

Isto é o que chamamos staccato.

NOTAS PONTUADAS

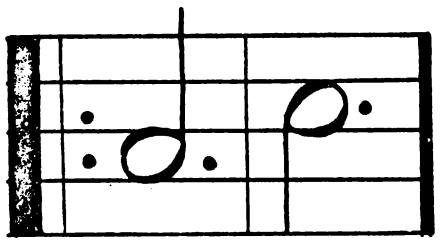
*Existe uma fórmula matemática que você poderá usar para calcular a duração da nota apesar de que, provavelmente, não será usada com muita freqüência. É a seguinte: (valor da duração da nota + valor da duração da nota * número de pontos)/2.*

Se você lê música já ouviu falar de notas pontuadas. O ponto à direita de uma nota musical significa que o valor da duração da nota é acrescido da metade do seu valor normal. Você sabe que existem outras durações de notas além de mínimas e semínimas. Por exemplo, existe uma semínima pontuada, equivalente à 3/8 de uma semibreve.

Você pode tocar tal nota acrescentando um ponto (.) ou uma série de pontos (...) ao número L. Cada ponto acrescentado (e pode-se acrescentar quantos quiser) aumenta a duração da nota em metade do seu valor normal. Por exemplo:

L4. = 1/4 + 1/8 = 3/8 da duração de uma semibreve

Tente isso:



PLAY "L4.;A;L8;C;L4.;E;L8;C;E;C;E;C;L4;A" (ENTER)

MUDANDO DE OITAVAS

Nossa oitava simples (nº 2) soa bem, mas como variedade é o tempero da música, torna-se monótono tocar a mesma coisa várias vezes seguidas (como um piano com apenas 12 teclas).

"O" - oitava - é a opção do PLAY que permite selecionar outras oitavas.

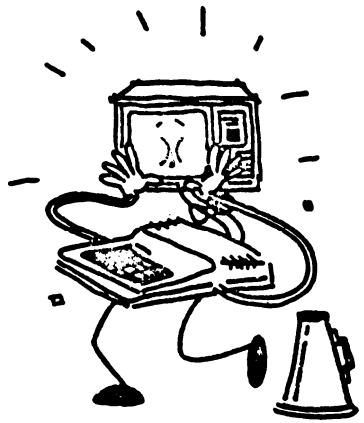
Para mudar a oitava, use "O" seguido por um número entre 1 e 5 (qualquer número fora desses limites resultará em um erro).

Seu computador automaticamente usa a oitava 2, que inclui o Dó central, se você não especificar a oitava desejada. Vamos tentar tocar a escala fundamental:

```
PLAY "CDEFGABAGFEDCBA" (ENTER)
```

O que aconteceu? G é a maior nota na oitava 2, então quando o computador alcançou A, ele começou de novo a partir do início da oitava. Tente isto:

```
PLAY "CDEFG;O3;ABAO2;FEDCBA" (ENTER)
```



Apostamos como você já ou<u>viu falar em "abaixar o rádio" mas não em "abaixar o computador" !

TOQUE MAIS ALTO
(VOLUME)

É claro, você pode ajustar o volume da sua música aumentando ou diminuindo o volume da televisão, mas você não vai querer ficar sentado ao lado do aparelho variando o volume de uma música, especialmente quando o computador pode fazer isso por você.

Seu computador faz isso com a característica V (volume). Tudo o que você tem que fazer é usar V seguido por um número entre 0 e 31. Se você não especificar o valor de V, o computador automaticamente assume V15.

NOTA: O computador usará o último valor de V, até que você o mude.

Ajuste o volume da televisão em uma tonalidade média normal e execute este programa.

```
5 CLS
10 PLAY "V15;A;V10;A;V15;A;V20;A;V25;A;V30;A"
```

Pressione a tecla (BREAK) para sair do loop ou você ficará com uma grande dor de cabeça em muito pouco tempo.

UM MOMENTO DE SILÊNCIO, POR FAVOR
(PAUSA)

Talvez esse último programinha fosse bem melhor de ser ouvido se as notas não fossem tocadas todas juntas. Vamos usar a característica P (pausa) para conseguirmos uns poucos momentos de silêncio entre as notas e ver se não soa melhor.

Para colocar uma pausa entre as notas, use P seguido por um número entre 1 e 255. Estes números correspondem aos mesmos números usados em L (duração da nota), com uma diferença muito importante — os pontos não podem ser usados com P. Em compensação, você pode usar uma série de pausas substituindo os pontos.

Por exemplo, para conseguir uma pausa de 3/8 da duração de uma semibreve, digite P4P8.

Deixamos espaços livres entre cada combinação de volume/nota para que você possa ler a linha sem dificuldade. Porém, esses espaços são desnecessários.

Mude a linha 10 do último programa para:

```
10 PLAY "V5;A; P2; V10;A; P2; V15;A; P2; V20;A; P2; V25;A; P2; V30;A; P2"
```

Na verdade, uma pausa de mínima (P2) entre todos aqueles A's não faz grande diferença mas dá para ter uma idéia de como o P funciona.

É HORA DE MARCARMOS OS TEMPOS
(ANDAMENTO)

Agora, o programa deve estar assim:

```
5  CLS
10 PLAY "V5;A;P2; V10;A;P2; V15;A;P2; V20;A;P2; V25;A;P2; V30;A;P2"
20 GOTO 10
```

Se não está agradável, pelo menos é tolerável, porém o andamento está um pouco lento. Você pode aumentar ou diminuir este andamento com T e um número entre 1 e 255. Se você não especificar o número T, o computador automaticamente usa T2. Comece diminuindo o andamento do nosso programa.

```
10 PLAY "T1; V5;A;P2; V10;A;P2; V15;A;P2; V20;A;P2; V25;A;P2; V30;A;P2"
```

Aumente a velocidade mudando de T1 para T15. Agora está melhor.

Que tal aumentar ao máximo a velocidade (255) e executar o programa? Não levou muito tempo, levou?

EXECUTANDO UM SUBSTRING
(OPÇÃO X)

Você se lembra como executou um substring usando a opção (X) com DRAW? PLAY tem uma característica parecida que permite executar um substring, retornar ao string original e completá-lo.

A função execute (X) tem a seguinte forma:

```
XA$;
```

A$ contém uma seqüência de comandos e funções normais do PLAY. X diz ao computador para usar PLAY A$. É sempre necessário usar o ponto e vírgula (;) depois de A$.

Reorganize o programa de demonstrações de maneira que ele execute um substring.

```
5  CLS
10 A$= "A;A#;A-"
20 B$= "05;XA$;"
30 C$= "01;XA$;XB$;"
40 PLAY C$
```

Execute o programa e acompanhe-o.

NOTA: Um sinal dólar ($) deve vir sempre acompanhado de um ponto e vírgula toda vez que você usar a função de execução. Neste exemplo, você pode retirar todos os pontos e vírgulas com exceção dos que vêm depois do $.

MAIS UMA NOTA ...

(+ , - , < , >)

Não, nós não vamos inventar novas notas, com H ou J. Queremos apenas mostrar uma outra maneira de usar algumas das opções que descrevemos.

Com O (oitava), V (volume), T (andamento) e L (duração da nota), você pode usar qualquer um dos sufixos abaixos, em vez de acrescentar um número.

SUFIXO	OBJETIVO
+	Adiciona 1 ao valor atual
-	Subtrai 1 do valor atual
<	Multiplica o valor atual por 2
>	Divide o valor atual por 2

Vamos demonstrar essas características usando um dos programas modelos:

```
5  CLS
10 PLAY "T2"
20 PLAY "A;A#;A-"
30 GOTO 20
```

Repare que estabelecemos o valor de T na linha 10. Execute o programa apenas para ver como ele está. Não mudou nada, é o mesmo de sempre. Agora insira T+ na linha 20.

```
20 PLAY "T+; A;A#;A-"
```

Execute o programa. O sinal de mais (+) automaticamente adiciona 1 ao valor de T a cada vez que a linha 20 é "tocada". Percebeu a mudança por volta de T100?

Partindo de um início lento você começa realmente a voar.

Agora use o T (andamento) de uma outra maneira: com um sinal menos (-).

Execute este programa:

```
5  CLS
10 PLAY "T255"
20 PLAY "T-; A;A#;A-"
30 GOTO 20
```

Depois de um início rápido, o computador vai reduzindo o andamento até 1 (de um em um).

A multiplicação não é mais rápida do que a adição? Na linha 10, reponha o valor 2 para o andamento, mude T na linha 20 para T> e execute.

```
10 PLAY "T2"
20 PLAY "T>; A;A#;A-"
```

Você começou com T2, certo? O computador multiplicou esse valor (2) por 2=4, 4X2=8, 8X2==16 e assim por diante até alcançar 255.

Você também pode reduzir o andamento com a mesma rapidez se dividir o valor atual por 2 usando "<".

```
10 PLAY "T255"
20 PLAY "T<; A;A#;A-"
```

Lembre-se, você pode fazer a mesma coisa com L para diminuir ou aumentar a duração da nota, V para o volume e O para aumentar ou abaixar a oitava.

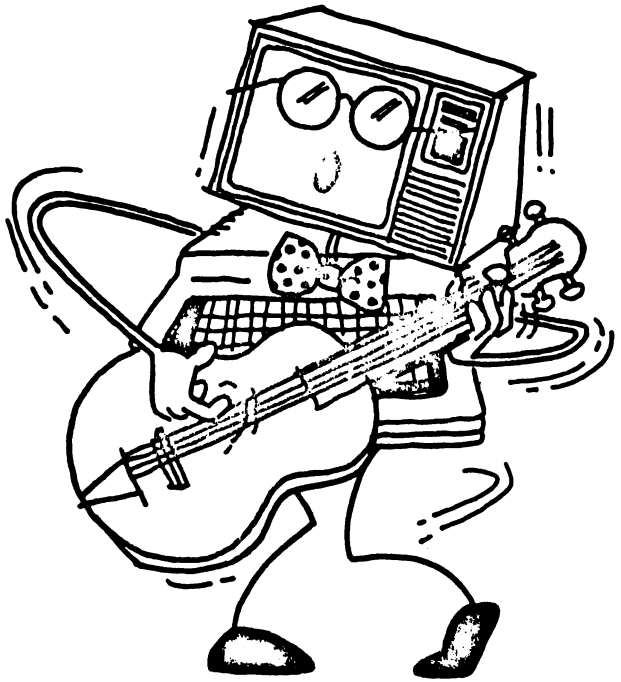
RELEMBRANDO VINÍCIUS DE MORAIS

Ok; você já está familiarizado com as funções do PLAY? Se já, observe-as em ação no seguinte programa e tente identificar essa melodia:

Você provavelmente se lembra do PRINT @ do Manual COLOR BASIC, mas você não verá nada sobre STRING$ até a seção II deste manual.

```
5   CLS
100 A$="V31;T2;O2;L4.;G;L8;E;L4;E;L8;D;L4.;G;L8;E;L4;E;L8;E;D;L4.;G;L4;E;E;
    L8;D;L4;G"
105 B$="L8;G;E;L4;E;L8;E;D;L4;F;D;D;L8;D;C;L4;E;C;C;L8;C;O1;L4;B-"
110 C$="O2;L1;C;P2;L4.;G;L8;E;L4;E;L8;D;L4.;G"
115 D$="L8;E;L4;E;L8;E;L4;D;G;E;E;L8;D;L4.;G;L8;E;L4;E;L8;E;D;L4;F"
120 E$="D;D;L8;D;C;L4;E;C;C;L8;C;O1;L4;B-;O2;L1;C;P2"
125 X$="XA$;XB$;XC$;XD$;XE$;"
130 PLAY X$
```

Reconhece essa canção? O que você diria de incrementá-la um pouco? Acrescente essas linhas:



```
10 PRINT @ 96,STRING$ (32,"*")
20 PRINT @ 166,"GAROTA DE IPANEMA"
30 PRINT @ 225,"TOM JOBIM E VINICIUS DE MORAIS"
35 PRINT @ 288,STRING$ (32,"*")
40 FOR X = 1 TO 500: NEXT X
45 CLS
50 PRINT @ 32,"OLHA QUE COISA MAIS LINDA"
55 PRINT @ 64,"MAIS CHEIA DE GRACA"
60 PRINT @ 96,"E' ELA MENINA QUE VEM E QUE PASSA"
65 PRINT @ 128,"NUM DOCE BALANCO, A CAMINHO DO MAR"
70 PRINT @ 224,"MOCA DO CORPO DOURADO"
75 PRINT @ 256,"DO SOL DE IPANEMA"
80 PRINT @ 288,"O SEU BALANCADO E' MAIS QUE UM"
```

```
85 PRINT @ 326, "POEMA"
90 PRINT @ 352,"E' A COISA MAIS LINDA"
95 PRINT @ 390,"QUE EU JA' VI PASSAR"
```

Execute o programa de novo e cante junto com o computador. Você adorou não foi? Tanto que quer ouví-la de novo. OK. Acrescente mais estas linhas:

```
150 CLS
160 PRINT @ 233,"TOQUE OUTRA VEZ"
165 FOR X = 1 TO 500: NEXT X
170 CLS
175 PRINT @ 233,"COM TODO O PRAZER"
180 FOR I = 1 TO 500: NEXT I
185 GOTO 5
```

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 9.2

Nossa versão de GAROTA DE IPANEMA soa muito bem, mas ainda não tem o balanço do Rio de Janeiro. Acrescente mais balanço ao programa básico para adaptá-lo ao seu próprio gosto musical.

Tente completar a música.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 9 3

Tente alguns arranjos musicais próprios. Nós incluímos vários deles no Apêndice C.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) O computador não designará automaticamente um valor para:

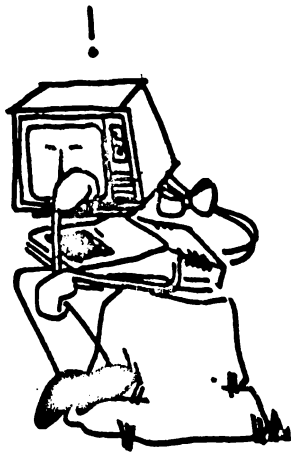
   a - oitava
   b - pausa
   c - duração da nota
   d - andamento

2) Qual das opções a seguir é válida somente para valores de 0 a 31?

   a - volume
   b - andamento
   c - nota
   d - oitava

3) Faça um círculo em volta do(s) ponto e vírgula(s) que podem ser omitidos:

```
PLAY "L4;XA$;8;3;A"
```



4) O comando PLAY exige qual dos PMODE's a seguir?

a - 0
b - 1
c - 2
d - 3
e - 4
f - nenhuma das respostas anteriores
g - qualquer uma das respostas anteriores

NOTAS

SEÇÃO II

DE VOLTA AO BASIC

CAPÍTULO 10

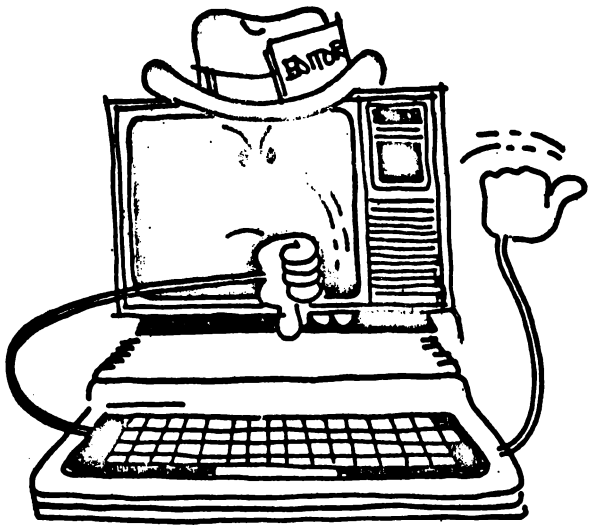
LENDO

E

ESCREVENDO

10

LENDO E ESCREVENDO



Neste Manual, bem como no Manual COLOR BASIC , você provavelmente perdeu muito tempo re digitando linhas de programa sempre que teve que fazer alguma alteração.

O COLOR BASIC EXTENDIDO oferece uma maneira bem mais fácil de fazer mudanças nos programas. Chamamos esta característica de "edição" porque ela permite mudar, eliminar, inserir, pesquisar, destruir e ampliar as linhas de um programa.

NÃO JOGUE FORA ESTA LINHA, EDITE-A! (EDIT)

Vamos fazer de conta que você cometeu um erro quando digitava o programa "LINHAS" no início deste Manual e que por alguma razão, ocorreu um erro na linha 30.

```
30 LINF (0,191)-(255O,0), PSET
```

Para editar a linha 30, digite:

```
EDIT 30 (ENTER)
```

A sintaxe do comando é a seguinte: EDIT número da linha (ENTER).

e a tela mostrará:

```
30▒
```

Agora você pode fazer mudanças na linha 30.

> NOTA: Quando estiver na modalidade de edição, pressione a tecla (ESPAÇO) para mover o cursor à frente e (←) para retroceder.

Em primeiro lugar, é preciso mudar o F para E na palavra LINF. Para fazer isso na modalidade de edição, você deve:

- Posicionar o cursor "em cima" da letra que foi digitada errada (F)
- Pressionar (C) (change - mudança)
- Pressionar a tecla da letra correta (E).

e então a alteração estará feita. A linha ficará assim:

```
30 LINE ▒
```

Para dar uma olhada na linha inteira, pressione (L) (list) e a linha será mostrada na sua forma real, pronta para novas edições.

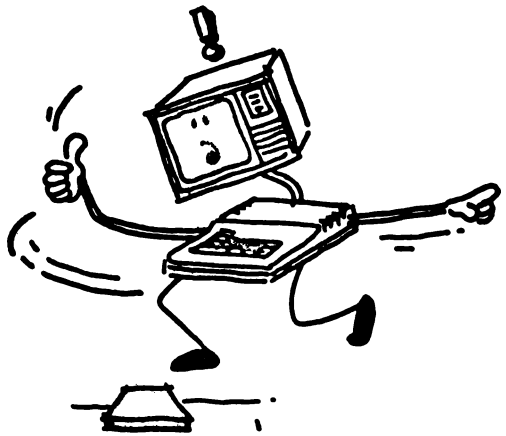
Para mudar mais de um caráter na modalidade de edição, você deve:

Quando falamos em "caracteres", estamos nos referindo também a "espaços".

- Posicionar o cursor em cima da primeira letra que quer mudar;
- Digitar o número de caracteres que serão mudados;
- Pressionar (C) ;
- Digitar os novos caracteres.

NOTA: Depois de digitar o número e a letra (C) (para mudança), é preciso pressionar n teclas antes de poder sair deste subcomando. Não adianta pressionar (ENTER) (BREAK) ou (SHIFT) (↑) para sair do subcomando é preciso completar toda a "mudança".

VOCE ESTÁ FORA (DELETE)



Bem, isto resolveu o problema com o primeiro erro mas ainda há um 0 a mais dentro do segundo conjunto de parênteses. Ele precisa ser eliminado. O primeiro método de eliminação é parecido com o primeiro método de mudança.

- Posicione o cursor em cima do caráter indevido
- Pressione (D) (delete-eliminar)

e a operação está completa. Por exemplo:

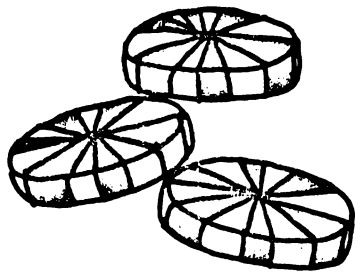
30 LINE (0,191)-(255*

pressione (D) para eliminar o 0. Parece não ter acontecido nada na tela. Para certificar-se que o caráter foi cancelado, pressione (L) para ver a linha e continuar editando ou pressione (ENTER) para terminar a operação de edição.

Existe um outro processo de eliminação que envolve a eliminação de um número específico de caracteres.

- Posicione o cursor sobre o primeiro caráter a ser eliminado;
- Digite o número indicando quantos caracteres devem ser cancelados;
- Pressione (D).

INCLUA O QUE QUISER (INSERT)



A esta altura, a linha 30 já deve estar assim:

30 LINE (0,191)-(255,0),PET

Ela está correta com excessão de um S que está faltando entre o P e o E no último parâmetro (deveria ser PSET e não PET). Você deve:

- Posicionar o cursor sobre o caráter onde será feita a inserção;

- Pressionar (I); (insert-inserir)
- Digitar os caracteres a serem inseridos

Há porém, uma grande diferença entre inserir e as características de edição descritas anteriormente. Após fazer a inserção você continuará na modalidade de inserção. Por exemplo, se tentar avançar o cursor — pressionando (ESPAÇO) após ter feito a inserção, você irá incluir novos espaços.

Primeiro foi o cachorrinho de estimação, depois uma pedra especial e agora, um computador de estimação. E depois?

Uma vez dada entrada na modalidade de edição, não é preciso pressionar (ENTER) depois de subcomandos como mudança, inserção, listagem, etc.

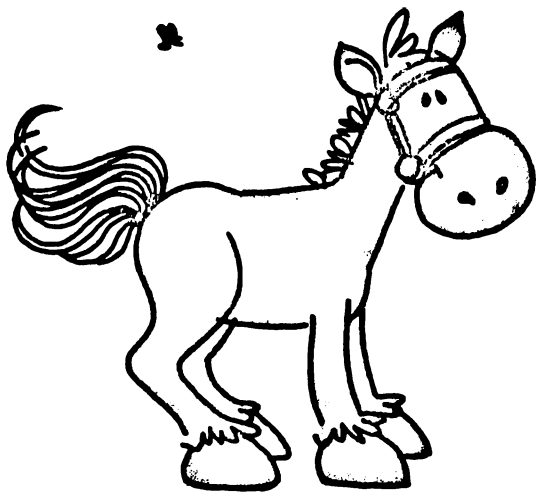
Para sair da modalidade de inserção, é preciso pressionar (ENTER) para encerrar a operação de edição ou pressionar as teclas (SHIFT) e (↑) para interromper a inserção mas continuar editando.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 10.1

Mude PSET para PRESET usando a modalidade de inserção (Depois faça o contrário usando a modalidade de eliminação).

CORTE O QUE É DESNECESSÁRIO

O COLOR BASIC EXTENDIDO tem uma função que combina as funções de eliminação com a de inserção. Ela é chamada "hack" porque permite alterar uma linha cortando o final e inserindo novos caracteres. Para usar esta característica, você tem que:



- Posicionar o cursor sobre o primeiro caráter que será cortado;
- Pressionar (H);
- Digitar os novos caracteres;
- Pressionar (ENTER) para interromper, ou (SHIFT) (↑) para continuar editando.

Suponhamos que a linha 30 esteja assim:

```
30 LINE (0,191)-PRESET, (255,0)
```

e que você queira mudar tudo, após o sinal menos (-). Você teria que posicionar o cursor no P de PRESET.

```
30 LINE (0,191)-■
```

pressione (H) e digite a nova informação:

```
30 LINE (0,191)-(255,0),PSET■
```

Você ainda está na modalidade de inserção. Pressione (ENTER) para completar a edição ou pressione (SHIFT) (↑) para interromper a inserção e continuar editando.

O COLOR BASIC EXTENDIDO ACERTA NOVAMENTE
(EXTEND)

Talvez tenha alcançado um estágio em que seja necessário usar o programa "LINHAS" acrescentando a opção B na linha 30.

Uma maneira de fazer isso é re-digitar a linha toda. Isto fará o trabalho mas um tanto quanto mal feito.

Uma outra maneira poderia ser dando entrada na modalidade de edição, mover o cursor até o fim da linha e então usar a característica de inserção — menos grosseira mas ainda um pouco complicada.

O COLOR BASIC EXTENDIDO tem uma solução muito simples para este problema. Você pode acrescentar novos caracteres no final da linha na modalidade de edição.

- Pressione (X) (extend-extender)
- Digite os caracteres adicionais
- Pressione (ENTER) para interromper a edição ou (SHIFT) (↑) para continuar editando

Por exemplo, para extender a linha 30, dê entrada na modalidade de edição e a tela mostrará:

30 ▒

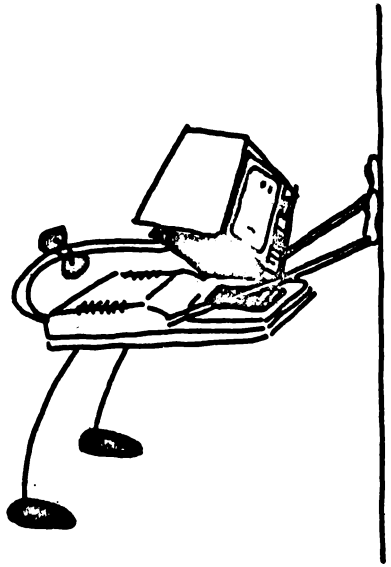
Pressione (X) (para extensão); o cursor pula imediatamente para o final da linha e permanece na modalidade de inserção:

```
30 LINE (0,191)-(255,0),PSET■
```

Você está na modalidade de inserção e pode portanto, começar a acrescentar caracteres:

```
30 LINE (0,191)-(255,0),PSET,B■
```

> FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 10.2
>
> Use esta característica "X" não apenas para gerar uma caixa, mas também para preenchê-la com uma cor específica. (Isto é, edite nossa versão da linha 30).



PROCURE-ME (SEARCH)

Uma outra maneira de fazer o cursor mover-se rapidamente para frente é obrigar o computador a buscar (pesquisar) um determinado caráter através da linha do programa.

Se, por exemplo, você está na modalidade de edição e decide eliminar o B da linha 30, como é que você pode chegar até o B sem ter que pressionar (ESPAÇO) toda vida até o fim da linha? Assim:

- Pressione (S) (search-pesquisa)
- Pressione a tecla do caráter até o qual você quer chegar (B, neste caso).

O cursor pesquisa a linha até encontrar o caráter desejado e aí então, você pode cancelar a letra ou usar qualquer uma das características de edição.

Isso funciona muito bem numa linha como a linha 30, que tem apenas uma única ocorrência do caráter especificado (por exemplo, apenas um B) mas e se fosse uma linha que tivesse mais de uma ocorrência? Por exemplo, se você inadvertidamente colocasse um parênteses a mais no segundo conjunto de parênteses.

```
30 LINE (0,191)-((255,0),PSET
```

Se você simplesmente disser ao computador para procurar por um'('ele irá parar no primeiro'('.

Você pode pedir ao computador para procurar por um segundo '(' da seguinte maneira: dê entrada na modalidade de edição na linha 30, pressione (2) (significa a segunda ocorrência) seguido por (S) (para pesquisa).

Após ter feito isso, pressione ((−) o caráter que deve ser encontrado.

Deve aparecer o seguinte na tela:

```
30 LINE (0,191)-▒
```

Você pode então efetuar qualquer operação de edição. Neste caso, queremos eliminar um caráter. Se lembra como fazer isso? Ótimo, então faça!

DESTRUA O ADVERSÁRIO ... AH ... ERRO (KILL)

KILL é quase o oposto de Hack e permite cancelar tudo até a enésima ocorrência do caráter especificado. Na modalidade de edição, você deve:

- Digitar o número de ocorrência do primeiro caráter a ser salvo (não destruído);
- Pressionar (K);
- Pressionar a tecla do primeiro caráter a ser salvo.

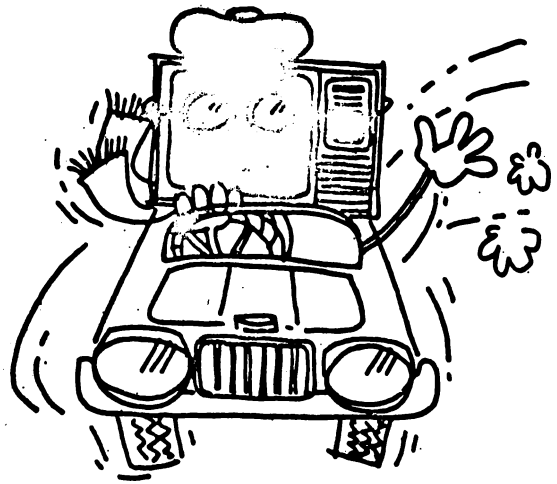
Suponhamos que você queira destruir a primeira metade da linha 30 (tudo até o sinal menos).

Primeiro dê entrada na modalidade de edição para a linha 30, digite (1) (para primeira ocorrência) e (K) (para KILL) e em seguida pressione (-) (o primeiro caráter a ser salvo).

Se você listar a linha agora, a tela mostrará o seguinte:

```
30-(255,0),PSET
```

VIVENDO A TODA VELOCIDADE
(MOVIMENTO DO CURSOR)



Já mostramos como movimentar o cursor para frente e para trás num ritmo relativamente lento e também explicamos como movê-lo rapidamente para frente numa operação de edição.

Algumas vezes, porém, você precisará movimentar o cursor rapidamente mas não para pesquisar, eliminar ou extender a linha. Neste caso:

- Digite o número de espaços que você quer avançar.
- Pressione (ESPAÇO)

Por exemplo, se pressionar (5) (ESPAÇO), o cursor avançará 5 caracteres. Você pode então efetuar qualquer operação de edição.

Para retroceder rapidamente, siga o mesmo procedimento substuindo porém (ESPAÇO) por (←)

Suponhamos, por exemplo, que o cursor esteja posicionado no final da linha 30 e você quer retroceder 6 espaços:

```
30 LINE (0,191)-(255,0),PSET■
```

Pressione (6) (o número de espaços) seguido por (←) e aparecerá na tela:

```
30 LINE (0,191)-(255,0#
```

Você pode então editar a linha a partir deste posição.

CARACTERÍSTICAS DO COMANDO EDIT	
Sintaxe de comando	**Operação desenvolvida**
EDIT número da linha	Edita a linha especificada.
(L)	Lista a linha que está sendo editada.
(C) novo caráter	Muda o caráter atual.
n(C) novos caracteres	Muda os próximos n caracteres.
(I)	Insere caracteres.
(D)	Cancela o caráter atual.
n(D)	Cancela n caracteres.
(H)	Elimina a linha a partir da posição atual e permite inserção de novos caracteres.
(X)	Extende a linha (o cursor é movido para o final da linha e a modalidade de inserção é ativada).
(S) caráter	Pesquisa a primeira ocorrência do caráter especificado.
n(S) caráter	Procura a enésima ocorrência do caráter especificado.
(K)	Elimina (cancela) o resto da linha.
n(K) caráter	Elimina (cancela) até a enésima ocorrência do caráter especificado.
n (ESPAÇO)	Avança o cursor n espaços. Se n for omitido, o computador usará 1.
n (←)	Retrocede o cursor n espaços. Se n for omitido , será usado 1.

Se você pressionar (L) para ver a linha enquanto estiver usando a modalidade de inserção, você irá inserir a letra "L" na linha do programa, em vez de listá-la.

MAIS LEITURA E ESCRITA

(DEL)

A única maneira de eliminar uma linha num programa em linguagem COLOR BASIC é digitar o seu número e pressionar (ENTER).

Isto não é problema no caso de uma ou duas linhas mas e se você quiser eliminar 50 ou 60 linhas de um programa? Você tem que digitar todos aqueles números e pressionar (ENTER) para cada um deles. (Seria mais fácil limpar a memória (NEW) e re-digitar o programa).

A linguagem COLOR BASIC EXTENDIDO é novamente a sua salvação, desta vez oferecendo uma maneira mais fácil de cancelar linhas do programa. Use o comando DEL.

Não esqueça de incluir um hífen(-) depois do comando, caso contrário, ocorrerá um erro.

Por exemplo, se você quer apagar todo o programa "LINHAS", digite:

```
DEL - (ENTER)
```

Se você precisa eliminar a linha 15, digite:

```
DEL 15
```

e assim por diante.

CARACTERÍSTICAS DO COMANDO DEL	
Sintaxe de comando	Operação desenvolvid:
DEL -	Elimina o programa inteiro.
DEL n	Elimina a linha n.
DEL n -	Elimina todas as linhas a partir da linha n.
DEL n1,n2	Elimina todas as linhas entre n1 e n2 (inclusive).
DEL - n	Elimina todas as linhas até n (inclusive).

VOCÊ ESGOTOU TODAS AS LINHAS (RENUM)

Até aqui foi possível mudar qualquer coisa na linha de um programa, exceto o próprio número da linha.

Bem, não se desespere mais, você também pode fazer isso com o comando RENUM. RENUM permite alterar todos os números das linhas numa forma especificada bem como todos os números de linhas que apareçam depois de GOTO, GOSUB, THEN, ON...GOTO, e ON...GOSUB.

A sintaxe do comando é a seguinte:

RENUM nova linha, linha inicial, incremento

nova linha	especifica o novo número da primeira linha a ser renumerada. É opcional e se omitido, é usado o número 10.
linha inicial	especifica o número da linha no programa original onde você quer começar a renumeração. É opcional e se omitido o programa inteiro é renumerado.
incremento	especifica o incremento a ser usado entre cada linha numerada. É opcional e se omitido é usado o número 10.

NOTA: RENUM não altera a ordem das linhas do programa, ele apenas designa novos números na ordem já existente.

Use o comando RENUM no nosso famoso programa "LINHAS".

```
5  PMODE 3,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191), PSET
30 LINE (0,191)-(255,0), PSET
40 GOTO 40
```

Se você quiser que o programa começe na linha 100, com um espaçamento 5 entre uma linha e outra, digite:

```
RENUM 100,5,5 (ENTER)
```

Agora, a primeira linha do programa terá o número 100 e cada linha que vier terá um acréscimo de 5 sobre a anterior. Liste o programa renumerado:

```
100 PMODE 3,1
105 PCLS
110 SCREEN 1,1
115 LINE (0,0)-(255,191), PSET
120 LINE (0,191)-(255,0), PSET
```

Repare que o computador também renumerou o número de linhas após a instrução GOTO na linha 125.

Lembre-se que nós dissemos que todos os parâmetros do comando RENUM são opcionais. Consequentemente, existem diversas variações de RENUM. Por exemplo, se você digitar:

```
RENUM (ENTER)
```

O computador renumera o programa inteiro começando por 10 e usando um incremento de 10.

E se você quiser renumerar todas as linhas após a linha 5? Tente:

```
RENUM 100, 10, 100 (ENTER)
```

e liste o programa. Você verá que o programa conserva a linha 5 sem qualquer mudança mas renumera todas as outras linhas.

Se digitar:

RENUM 10000, 100 (ENTER)

O computador renumera a linha 100 e todas as linhas numeradas acima de 100. A primeira linha renumerada passará a ser linha 10000 e será usado um incremento de 10 entre as linhas subsequentes.

Outra maneira de usar RENUM é digitar:

RENUM 100,,100

O programa inteiro será renumerado, começando com um novo número de linha 100 e usando um acréscimo de 100 entre elas.

A última maneira como você talvez queira usar RENUM é:

RENUM,,5

Neste caso o programa inteiro será renumerado, começando com um novo número de linha 10 e usando um incremento de 5.

CONDIÇÕES DE ERRO DO COMANDO RENUM

RENUM não pode ser usado para mudar a ordem das linhas do programa. Por exemplo, se o seu programa original tem linhas numeradas 10, 20 e 30, então o comando:

```
RENUM 15,30
```

não é válido já que o resultado seria mover a linha 30 para adiante da linha 20. Neste caso ocorrerá um erro (FC) e o programa original não sofrerá qualquer alteração.

RENUM não cria número de linhas maiores que 63999; se os parâmetros fornecidos gerarem um número de linha maior que 63999 ocorrerá um erro e não haverá mudança no programa original.

Se for usado um número de linha indefinido dentro do seu programa original, RENUM imprime uma mensagem de advertência.

```
UL xxxx IN yyyy
```

onde xxxx é o número da linha original e yyyy é o novo número da linha xxxx.

NOTA: RENUM renumera o programa apesar dessa mensagem de advertência e embora a linha indefinida não seja renumerada.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 10.3

Tente cada um dos exemplos já citados no programa "LINHAS" até que esteja bem acostumado com suas variações.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Indique a instrução incorreta:

   a - DELETE 30
   b - EDIT 30
   c - RENUM 30,5,30

2) Na modalidade de edição você pode listar uma linha na sua forma corrente e continuar editando se:

   a - Digitar LIST número da linha

b - Pressionar (L) (ENTER)

c - Pressionar (L)

3) Diga se é falsa ou verdadeira a afirmação abaixo:

RENUM permite mudar a ordem dos números das linhas de um programa.

4) Qual das características a seguir não é um subcomando de edição?

a - Inserir

b - Alterar

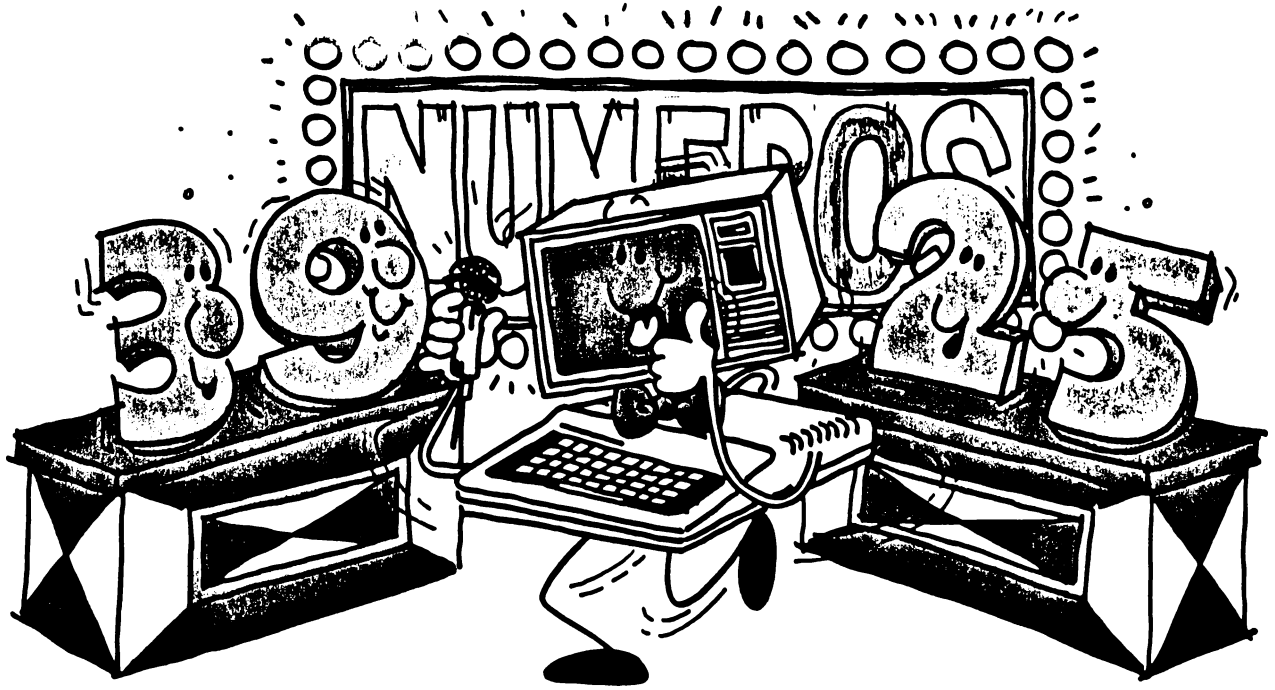
c - Eliminar

d - Pesquisar

e - Construir

NOTAS

CAPÍTULO II



O JOGO DOS NÚMEROS

11

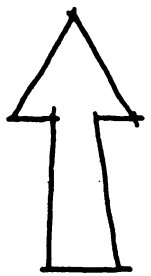
O JOGO DOS NÚMEROS

Você encontrará nos apêndices, uma tabela com muitas fórmulas matemáticas (geometria simples, trigonometria e álgebra).

Em adição às funções matemáticas simples descritas no Manual COLOR BASIC o repertório do COLOR BASIC EXTENDIDO inclui várias funções avançadas.

Daremos uma noção de cada função e mostraremos como usá-las para efetuar qualquer tipo de operação matemática de nível superior.

Antes de fazermos isto, porém, seria melhor se familiarizar com algumas definições. De qualquer maneira, ponha uma música suave no computador por alguns minutos, relaxe e vamos em frente.



POTENCIAÇÃO (↑)

Rápido! Quanto é 1.5 ao quadrado? e 77 ao cubo? Se você não sabe pergunte ao computador. É para isso que ele está aí, não é? Sempre que quiser elevar um número a uma potência n, siga este modelo.

Número ↑ potência

número — indica o número que você quer elevar a uma potência. Pode ser qualquer expressão numérica.

↑ — é gerado quando você pressiona (↑).

potência — é o exponente ao qual o número é elevado. Pode ser qualquer expressão numérica.

NOTA: A potenciação tem precedência sobre os outros operadores. Por exemplo, se tentar — 2 ↑ 2, o resultado será um número negativo. Para elevar corretamente -2 a segunda potência (resultando em um número positivo) coloque o -2 entre parênteses.

Não se preocupe com o "002". Isto é chamado "erro de arredondamento" e é inevitável porque (sentimos ter que dizer isso) o computador não é um calculista perfeito. Mas afinal, nenhuma máquina o é.

Vamos começar com o mais difícil, 77 ao cubo. Olhando a sintaxe do comando, você pode efetuar a operação? A resposta é 456533.002.

Se apareceu este resultado na tela, você começou bem.

```
PRINT 77^3
456533.002
OK
```

Tente elevar 10 a décima potência. Foi isto que apareceu na tela?

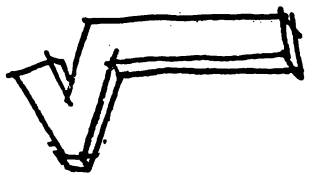
```
1.00000001E+10
```

Já que 10.000.000.000 tem mais que 9 dígitos significativos, o computador foi para a notação científica (Isto foi visto no Manual COLOR BASIC).

Tente 100 elevado à centésima potência. E então? Em vez do resultado apareceu um ?OV ERROR (overflow = estouro de capacidade)? Isto significa que a resposta está além da capacidade do computador. Este computador foi projetado para lidar com números de -10^{38} a $+10^{38}$ e qualquer exigência além disto resultará num erro de estouro de capacidade.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.1

Faça um programa que mostre o quadrado dos números inteiros de 1 a 10.



RAIZ QUADRADA (SQR)

A função SQR dá a raiz quadrada de um número:

SQR (número)

número é uma expressão numérica nunca menor que zero.

Por exemplo, se você quer a raiz quadrada de 100, digite:

PRINT SQR(100) (ENTER)

e você descobrirá (se é que ainda não sabe) que a resposta é 10.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.2

Faça outro pequeno programa que dê a raiz quadrada dos números de 10 em 10, de 100 a 0.

ESTUDO DO TRIÂNGULO

Queremos que dê uma olhada nesse triângulo. Vamos usá-lo durante toda nossa discussão a respeito das funções trigonométricas, portanto é preciso conhecê-lo bem.

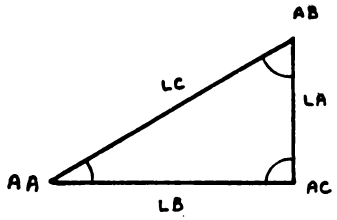
Você verá que a trigonometria tem muitas aplicações práticas. Por exemplo, imagine que o triângulo com o qual está trabalhando neste capítulo é, na verdade, o telhado de uma casa que você está construindo (talvez até a casa construída na seção I deste Manual). Estas funções podem ajudá-lo a determinar tanto o comprimento das vigas como a inclinação do telhado. Portanto, mesmo que não goste de matemática (mas goste de construção) este é provavelmente o capítulo que você estava procurando.

Repare que rotulamos os ângulos como o prefixo A (ângulo A = AA, etc). Para uma melhor

distinção entre ângulos e lados, marcamos os lados opostos aos seus respectivos ângulos com o previxo L, (por exemplo, lado LA é oposto ao ângulo AA, etc...)

Em trigonometria, LB é chamado "adjacente", lado LA, "oposto" e, lado LC, "hipotenusa".

Usando o nosso triângulo, podemos definir as funções trigonométricas comuns da seguinte maneira:



SENO de AA = SIN (AA) = oposto/hipotenusa = LA/LC
COSSENO de AA = COS (AA) = adjacente /hipotenusa = LB/LC
TANGENTE de AA = TAN (AA) = oposto/adjacente = LA/LB

GRAUS VS RADIANOS

Para definir um ângulo, pode-se usar qualquer uma das unidades de medida, mas a unidade mais comum é o grau e a unidade mais técnica é o radiano.

Este computador aceita qualquer ângulo medido em radiano. Já que radiano pode ser um pouco estranho para você, vamos convertê-los em graus (e vice-versa) desta maneira:

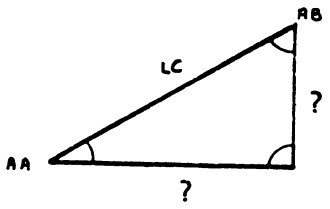
```
GRAUS PARA RADIANO : GRAUS/57.29577951
RADIANO PARA GRAUS : RADIANO *57.29577951
```

Nos programas simples deste capítulo incluímos um "conversor". Com isso, basta fornecer os ângulos em graus e o computador automaticamente os converte para radiano (e vice-versa).

SENO DE UM ÂNGULO (SIN)

A sua sintaxe é:

SIN (ângulo)	
ângulo	é uma expressão numérica para um ângulo em radiano.



Uma das aplicações de SIN e para determinar os dois lados desconhecidos de um triângulo se dois ângulos e o terceiro lado forem conhecidos.

Digite e execute o programa abaixo. Você pode fornecer qualquer valor.

```
5 CLS
10 INPUT "FORNECA O ANGULO A (AA)"; AA: IF AA<=0 OR AA>=180 THEN 100
20 INPUT "FORNECA O ANGULO B (AB)"; AB: IF AB<=0 OR AB>=180 THEN 100
30 INPUT "FORNECA O LADO   C (LC)"; LC: IF LC<=0 THEN 100
40 AC = 180-(AA+AB): REM  VALOR DO ANGULO AC
50 IF (AA+AB+AC) <> 180 THEN 100 : REM  TRIANGULO=180 GRAUS
```

No apêndice de "Programas Simples", você encontrará um programa que usa as funções gráficas de uma maneira bem parecida. O programa, chamado " Desenhos Triangulares ", desenha triângulos quando você especifica combinações de lados e ângulos.

```
60 AA=AA/57.29577951: AB=AB/57.29577951: AC=AC/57.29577951: REM
   CONVERTE GRAUS EM RADIANOS
70 LA=((SIN(AA))/(SIN(AC))) * LC: IF LA < 0 THEN 100
80 LB=((SIN(AB))/(SIN(AC))) * LC: IF LB < 0 THEN 100
90 PRINT "O LADO A (LA) TEM UM COMPRIMENTO DE" LA "E O LADO B (LB)
   TEM UM COMPRIMENTO DE " LB : GOTO 10
100 PRINT "DESCULPE, ISTO NAO E' UM TRIANGULO, TENTE NOVAMENTE" : GOTO 10
```

Quando o computador solicitar os ângulos AB e AC, você deve fornecê-los em graus. Se tentar usar ângulos negativos ou ângulos maiores ou iguais a 180 graus, o computador vai para a linha 100, imprime a mensagem, e fica pronto para outra tentativa. Se fornecer um número negativo para o lado LC, ocorrerá o mesmo resultado.

Já que você não sabe o tamanho do ângulo AC, o computador automaticamente determina isto na linha 40. Se a soma dos três ângulos não for igual a 180 graus, o computador toma a ação apropriada na linha 50. A linha 60 converte os "seus" graus nos radianos "do computador", de maneira que os cálculos dos senos possam ser efetuados.

ADVERTÊNCIA DE SENO

Você provavelmente já viu curvas de seno. Elas são usadas para indicar a potência AC e outras grandezas elétricas. Execute o programa e veja uma curva de seno "enrolada" horizontalmente (e dê uma olhada nos programas simples dos apêndices onde foi feito uma curva de seno mais elaborada).

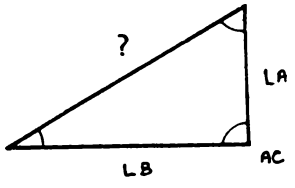
```
5  CLS
10 FOR A=180 TO -179 STEP -10
20 RD=A/57.29577951 : REM RADIANOS
30 PC=SIN(RD)*14+16.5 : REM PC=POSICAO NA COLUNA
40 PRINT TAB(PC); "S" : REM MARCACAO DE SENO (RD)
50 NEXT A
60 GOTO 60
```

COSSENO DE UM ÂNGULO (COS)

A função cosseno é relacionada com a função seno e abaixo damos a sua sintaxe:

COS (ângulo)
ângulo é uma expressão numérica para um ângulo em radianos.

Uma das aplicações de cosseno é para determinar o comprimento do terceiro lado de um triângulo se você já conhece dois lados e um ângulo. Para identificação do ângulo e lado, refira-se ao triângulo modelo.



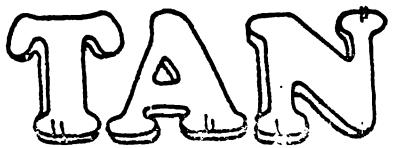

```
5 CLS
10 INPUT "FORNECA O ANGULO C (AC)"; AC: IF AC<=0 OR AC>=180 THEN 100
20 AC=AC/57.29577951: REM CONVERTE GRAUS PARA RADIANOS
30 INPUT "FORNECA O LADO   A (LA)"; LA: IF LA<=0 THEN 100
40 INPUT "FORNECA O LADO   B (LB)"; LB: IF LB<=0 THEN 100
```

```
50 LC=((LA^2)+(LB^2))-(2 * (LA*LB*COS(AC))): IF LC<0 THEN 100
60 PRINT "O LADO C (LC) TEM UM COMPRIMENTO DE" SQR(LC) : GOTO 10
100 PRINT "DESCULPE, ISTO NAO E' UM TRIANGULO, TENTE NOVAMENTE" : GOTO 10
```

Repare que o programa trabalha quase da mesma forma que o programa SIN exceto pelo uso da exponenciação (↑) na linha 50 e SQR na linha 60.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.3

Cosseno pode fazer curvas por conta própria. Reescreva o programa "curvas de seno" de maneira que ele trace COS(RD) em vez de SIN(RD). Use a letra C (cosseno) para marcar a curva feita por COS. Qual é a diferença entre os dois?



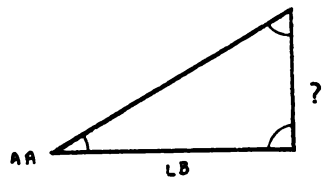
TANGENTE DE UM ÂNGULO (TAN)

A terceira função trigonométrica que você pode usar é TAN. Ela permite calcular a tangente de um ângulo.

TAN (ângulo)

Ângulo é uma expressão numérica para um ângulo em radianos.

A função tangente pode ser usada entre outras coisas, para determinar o comprimento desconhecido de um lado de um triângulo se você souber as dimensões de outro lado e um ângulo.




```
5 CLS
10 INPUT "FORNECA O LADO   B (LB)"; LB: IF LB<=0 THEN 100
20 INPUT "FORNECA O ANGULO A (AA)"; AA: IF AA<=0 OR AA>=180 THEN 100
30 AA=AA/57.29577951 : REM CONVERTE GRAUS EM RADIANOS
40 LA=LB*(TAN(AA)) : IF LA<=0 THEN 100
50 PRINT "O LADO A (LA) TEM UM COMPRIMENTO DE" LA : GOTO 10
100 PRINT "DESCULPE, ISTO NAO E' UM TRIANGULO, TENTE NOVAMENTE" : GOTO 10
```

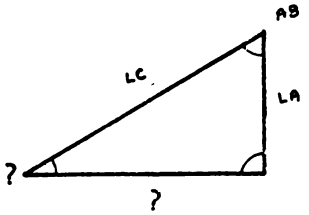
A chave para este problema está, é claro, na linha 40, onde a tangente do ângulo AA é multiplicada pelo comprimento do lado LB para determinar o comprimento do lado LA.

ARCOTANGENTE (ATN)

ATN (arcotangente) é o inverso da tangente e sua sintaxe é assim:

ATN (ângulo)
ângulo é uma expressão numérica.

O programa abaixo usa ATN e TAN para calcular dois ângulos desconhecidos de um triângulo quando dois lados e um ângulos são conhecidos.




```
10 CLS
20 INPUT "FORNECA O LADO A    (LA)"; LA: IF LA<=0 THEN 150
30 INPUT "FORNECA O LADO C    (LC)"; LC: IF LC<=0 THEN 150
40 INPUT "FORNECA O ANGULO B  (AB)"; AB: IF AB<=0 OR AB>=180 THEN 150
50 X=(180-AB) : REM AA+AC=180-AB
60 X=X/57.29577951 : REM CONVERTE GRAUS EM RADIANOS
70 Y=((LA-LC)/(LA+LC))*TAN(X/2)
80 Z=ATN(Y)
90 AA=(X/2)+Z
100 AC=(X/2)-Z
110 AA=AA*57.29577951 : REM CONVERTE RADIANOS EM GRAUS
120 AC=AC*57.29577951 : REM CONVERTE RADIANOS EM GRAUS
130 PRINT "O ANGULO A (AA) TEM" AA "GRAUS"
140 PRINT "O ANGULO C (AC) TEM" AC "GRAUS" : GOTO 20
150 PRINT "DESCULPE, ISTO NAO E' UM TRIANGULO, TENTE NOVAMENTE" : GOTO 20
```

TAN ((AA-AC)/2) é igual a (LA-LC)/(LA+LC) * TAN ((AA+AC)/2). Repare também que foi necessário converter os radianos do computador para os seus graus (linhas 110 e 120).

LOG

LOG determina o logaritmo natural de um número. Isto é o inverso de EXP, portanto X = LOG (EXP(X)).

LOG (número)
número é uma expressão numérica maior que zero.

O logaritmo de um número é a potência a qual uma "base" deve ser elevada para resultar no número. Os logaritmos são muito úteis na resolução de problemas científicos e matemáti - cos. Na função LOG, a base é e = 2.718281828.

Para mudar o logaritmo de um número para uma outra base B, use a fórmula log base B (x)= $\log_e(x)/\log_e(B)$. Por exemplo, LOG (32768)/LOG (2) determina o logaritmo na base 2 de 32768 (isto é, a potência a qual 2 é elevado para se ter 32768).

Tente isto:

```
PRINT LOG(1)
PRINT LOG(100)
PRINT LOG(2,718281828)
```

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.4

Calcule o LOG para os seguintes números:

a) 1003 b) 74.9865 c) 3.354285

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.5

Determine o logaritmo na base 10 dos seguintes números:

a) 1 b) 10 c) 100 d) 500 e) 0.1 f) 1001

Sugestão: $\log_{10} x = \frac{\log_e x}{\log_e 10}$

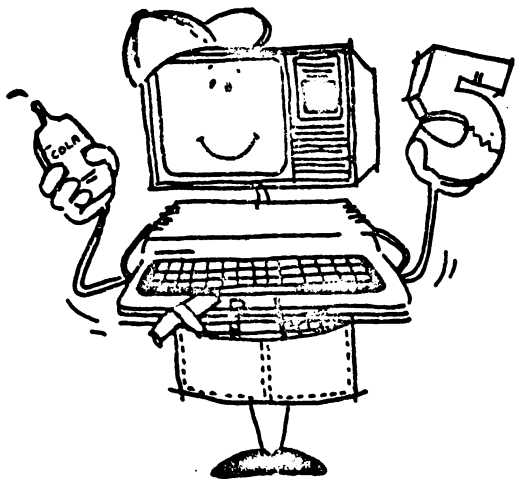
EXP

A função EXP calcula a exponencial natural de um número, isto é, $e^{\text{número}}$. Esta função é o inverso de LOG, portanto, X = EXP(LOG(X)).

EXP (número)
número é uma expressão numérica menor de 87.3365

Execute esse programa para ver EXP em ação:

```
10 CLS
20 INPUT "FORNECA UM VALOR X"; X
30 PRINT "EXP(X)=" EXP(X)
40 GOTO 20
```



USE O COMPUTADOR COMO UM QUEBRA-GALHOS MATEMÁTICO (FIX)

É impressionante quando o seu computador desenvolve um número até nove dígitos significativos, especialmente quando oito desses números estão à direita do ponto decimal.

Algumas vezes, entretanto, você pode não querer usar todos esses números; você pode precisar apenas da parte inteira do número (por exemplo, o número à esquerda do ponto decimal). A função FIX permite chegar ao número inteiro simplesmente cortando todos os dígitos à direita do ponto decimal.

FIX (número)
número é uma expressão numérica.

Por exemplo:

```
PRINT FIX(2.2643951)
2
OK
```

Temos aqui um programa que separa a parte inteira da parte fracionária de um número:

```
10 CLS
20 INPUT "FORNECA UM NUMERO COMO X,YZ"; X
30 W=FIX(X)
40 F=ABS(X)-ABS(W)
50 PRINT "PARTE INTEIRA    ="; W
60 PRINT "PARTE FRACIONARIA="; F
70.GOTO 20
```

Quando utilizar esta característica não se esqueça de usar a instrução DEF FN antes de tentar executar a função definida por ela. Caso contrário, ocorrerá um ?UF ERROR (função indefinida).

DEFINA SUAS PRÓPRIAS FUNÇÕES (DEF FN)

O COLOR BASIC EXTENDIDO tem uma função numérica DEF FN, que é diferente de qualquer outra que falamos até aqui. Ela lhe permite criar suas próprias funções matemáticas. Você

pode usar esta nova função exatamente da mesma maneira que qualquer uma das outras funções disponíveis (tais como SIN, COS, etc). Uma vez usada a instrução DEF FN para definir uma função, você pode pô-la em ação no seu programa usando o prefixo FN na frente do nome designado para a nova função.

DEF FN nome (lista de variáveis) = fórmula

nome	é o nome dado a função criada por você.
lista de variáveis	indica uma variável fictícia para cada variável a ser usada pela função.
fórmula	define a operação em termos das variáveis dadas na lista de variáveis.

NOTA: As variáveis que aparecem na fórmula servem apenas para definir a fórmula ; eles não afetam as variáveis do programa que têm o mesmo nome.
Apenas um argumento é permitido na chamada de fórmulas, conseqüentemente DEF FN deve conter apenas uma variável.

NOTA: A função DEF FN só pode ser usada em um programa, nunca num comando imediato.

Por exemplo, uma operação matemática que você teve que usar várias vezes neste capítulo foi a conversão de graus para radianos. Não seria ótimo se o computador tivesse uma função integrada que fizesse isso para você?

Se você mudar o programa simples que usamos para SIN, verá como criar um DEF FN que converta graus em radianos.

```
7 DEF FNR(X)=X/57.29577951
60 AA=FNR(AA) : AB=FNR(AB) : AC=FNR(AC)
```

Você pode ver imediatamente quanto trabalho de digitação isto lhe poupa, já que você teve que fornecer 57.2957751 apenas uma vez. Sempre que chamar FNR, o computador automaticamente insere o valor fornecido na função e efetua a operação designada.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 11.6

Use o DEF FN para:

1) Converter radianos para graus
2) Criar uma função matemática que eleve números ao cubo.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Qual das instruções abaixo sempre retorna um número inteiro?

a) SIN
b) COS
c) FIX
d) ATN

2) Você pode criar suas próprias funções matemáticas com:

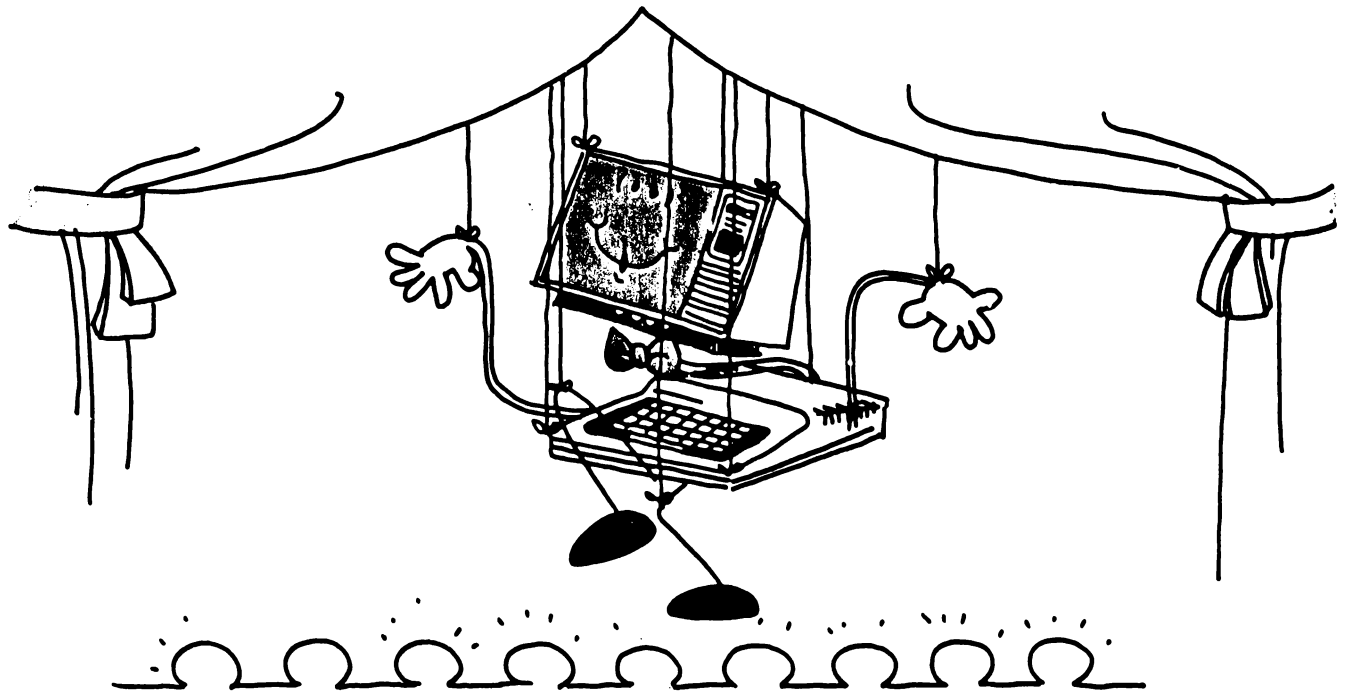
a) SIN
b) COS
c) FIX
d) DEF FN

3) Qual das afirmativas abaixo está errada?

a) SIN (ângulo)
b) LOG (ângulo)
c) COS (ângulo)
d) TAN (ângulo)

NOTAS

CAPÍTULO 12



MANIPULANDO STRINGS

12

MANIPULANDO STRINGS

Não queremos prendê-lo a repetições, mas o COLOR BASIC EXTENDIDO possui outras instruções de tratamento de strings. O assunto "strings" foi extensamente discutido no Manual COLOR BASIC e caso precise relembrar um pouco, consulte este manual.

STRING$

A função STRING$ é muito útil para se fazer gráficos, tabelas ou qualquer outra repre - sentação visual similar, porque ela cria uma série de caracteres.

STRING$ (extensão, caráter)	
extensão	é uma expressão numérica de 0 a 255.
caráter	é uma expressão "string" para um caráter (neste caso deve estar escrito entre aspas) ou uma expressão numérica para um código ASCII.

O número de caracteres exibidos depende do número que você especificar para extensão. Os

caracteres a serem usados dependem do caráter ou do código ASCII que você especificar. Veja no Apêndice E a lista completa de códigos dos caracteres ASCII.

Por exemplo, você pode querer "enfeitar" um pouco o nosso pobre e surrado programa "LINHAS":

```
5  CLS
6  X$ = STRING$ (13,"*")
7  PRINT @ 96, X$; "LINHAS"; X$
8  FOR X = 1 TO 1000: NEXT X
10 PMODE 3,1
15 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
40 GOTO 40
```

Na linha 6, por exemplo, atribuímos o valor STRING$ (13,"*"), que é uma série de treze asteriscos, à variável X$.

A linha 7 diz ao computador para imprimir X$ (começando na posição 96 da tela) seguida da palavra LINHAS, seguida novamente por X$. (Veja o Gabarito da Tela no Apêndice M). Como X$ é igual à treze asteriscos (*), estes caracteres são impressos antes e depois de LINHAS.

O que? Você quer enfeitar ainda mais o programa? Tudo bem! Acrescente essas linhas:

```
16 Y$= STRING$ (31,42)
17 PRINT@ 384,Y$
```

Desta vez você não determinou um asterisco(*) para ser mostrado na parte inferior da tela; em vez disso, você disse ao computador para mostrar o caráter representado pelo código ASCII 42. E, como você provavelmente já deve ter adivinhado, o código ASCII 42 por coincidência representa um asterisco.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 12.1

Você alguma vez já fez listas para controlar os trabalhos que você ou outras pessoas têm que fazer?

Faça um programa que crie uma lista de controle usando STRING$.

INSTR

Se você quiser procurar por um string contido em outro string, use INSTR, que tem a seguinte sintaxe:

INSTR (posição, string1, string2)

posição determina a posição em string1 onde a busca deve começar. É uma expressão numérica de 0 a 255 e é opcional. Se for omitida, a busca começa automaticamente pelo primeiro caráter em string1.

string 1	é o string no qual se quer pesquisar.
string 2	é o string que se está procurando.

Você tem a opção de especificar em "string1" onde a procura deve começar. Caso você decida não usar a opção, a busca começará automaticamente pelo primeiro caráter em string1.

INSTR retorna um 0 se:

- a posição for maior que o número de caracteres em string1
- string1 for nulo
- string2 não puder ser encontrado

Vejamos como INSTR trabalha no programa a seguir:

```
5  CLEAR 500
10 CLS
15 INPUT "STRING1";S1$
20 INPUT "STRING2";S2$
25 C=0: P=1  'P=POSICAO
30 F  =  INSTR (P,S1$,S2$)
35 IF F = 0 THEN 60
40 C=C+1
45 PRINT LEFT$ (S1$,F-1)+STRING$(LEN(S2$),CHR$(12
   8))+RIGHT$(S1$,LEN(S1$)-F-LEN(S2$)+1)
50 P=F+LEN (S2$)
```

```
55 IF P <= LEN(S1$)-LEN(S2$)+1 THEN 30
60 PRINT "NUMERO DE OCORRENCIAS: ";C
```

O programa a seguir é um modelo e você pode fornecer qualquer texto.

```
STRING1? VOCE DEVERIA TENTAR USAR SEU
COMPUTADOR O MAXIMO POSSIVEL.
STRING2? CO
VOCE DEVERIA USAR SEU --MPUTADOR O
MAXIMO POSSIVEL
VOCE DEVERIA TENTAR USAR SEU COMPUTADOR O
MAXIMO POSSIVEL
NUMERO DE OCORRENCIAS: 1
OK
```

O que aconteceu foi o seguinte:

- Você atribuiu o valor VOCE DEVERIA TENTAR USAR O SEU COMPUTADOR O MAXIMO POSSIVEL (linha 15) a S1$(string 1).

- Você designou o valor CO (linha 20) para S2$.

- Você disse ao computador para iniciar na primeira posição (P) em S1$ e começar a buscar S2$ (linha 30).

- Quando a função INSTR localizou o S2$ ela imprimiu e separou o S2$(CHRS(128)). Partiu

então a procura da próxima ocorrência de S2$.

- Finalmente, o computador mostrou quantas ocorrências de S2$ ele encontrou em S1$(linha 60).

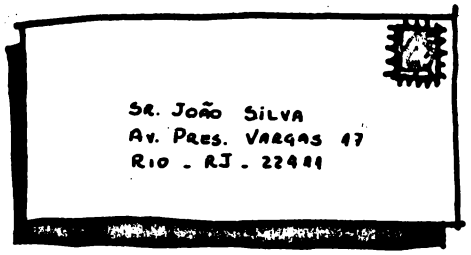
FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 12.2
Faça um programa que imprima a primeira e a segunda ocorrências de B em ABCDEB.

O programa de armazenamento de dados a seguir contém uma lista de correspondência com nomes e endereços. Esta é uma maneira muito fácil de se armazenar informações. Repare como poupamos espaço de armazenamento não deixando espaço entre as palavras. Isto dificulta a sua leitura mas não faz a menor diferença para o computador.

Note também como identificamos os códigos de endereçamento postal (CEP) simplesmente colocando um asterisco (*) antes deles. Desse modo o computador não os confundirá com os números das ruas.

Neste caso estamos procurando pelos nomes e endereços de todas as pessoas que moram na área especificada pelo CEP 22411. Consequentemente,*22411 será atribuído à variável A$ (string2).

```
5  CLS
10 A$="*22411"
```




```
20 X$="JOAO SILVA,PRES. VARGAS 17,RIORJ*22411"
30 Y$="LUIS SILVEIRA,B. DÁ TORRE 32,RIORJ*22400"
40 Z$="MARIA TEIXEIRA,NS COPACABANA 45,RIORJ*22412"
```

Para que o computador possa procurar por X$, acrescente:

```
50 PRINT INSTR(X$,A$)
```

e execute o programa. A tela deveria mostrar

```
33
OK
```

que significa que X$ contém o nome e endereço que você precisa.

E o Y$? Edite a linha 50 para que o computador procure através daqueles endereços. Ele lhe disse se encontrou o nome requisitado?

Agora tente Z$. Um zero é justamente a maneira do computador dizer "Não existe nesta lista nenhum dos nomes que você procura".

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 12.3

Modifique o programa lista de correspondência de duas maneiras:
1 - De maneira que X$ contenha dois endereços que tenham o CEP 22411
2 - E que o computador procure por toda ocorrência de *22411,e não apenas a primeira.

MID$

A instrução MID$ é uma instrução de edição de strings muito potente, pois permite substituir uma parte do string por outra. Esta instrução não é a mesma instrução MID$ do COLOR BASIC.

MID$ (string original, posição, comprimento)= string substituto

string original	é o nome da variável string que você quer mudar;
posição	é uma expressão numérica especificando a posição do primeiro caráter a ser mudado.
comprimento	é uma expressão numérica especificando o número de caracteres a serem substituídos. É opcional e se for omitido, todo o string substituto será usado.
string substituto	é um string que substitui a parte especificada da série antiga.

NOTA: Se "string substituto" for menor que "comprimento" ele será completamente usado na substituição. A resultante terá sempre o mesmo comprimento do string original.

Para ver o que queremos dizer, execute este programa:

```
5  CLS
10 A$="SAO PAULO,RJ"
20 MID$ (A$,12)="SP"
30 PRINT A$
```

Na linha 10, foi designado SAO PAULO,RJ para o valor de A$. Na linha 20 então, você disse ao computador que usasse MID$ para substituir parte do string original (A$) por SP, começando na posição 11.

Mude a posição na linha 20 de 11 para 8 e execute o programa. Você deve conseguir este resultado:

```
SAO PAUSP,RJ
```

Agora acrescente a opção "comprimento" na linha 20:

```
20 MID$ (A$,12,2)="SP"
```

e repare como isto não afeta o resultado já que o string substituto e o string original tinham ambos dois caracteres de comprimento. Mude o comprimento para 1:

```
20 MID$ (A$,12,1)="SP"
```

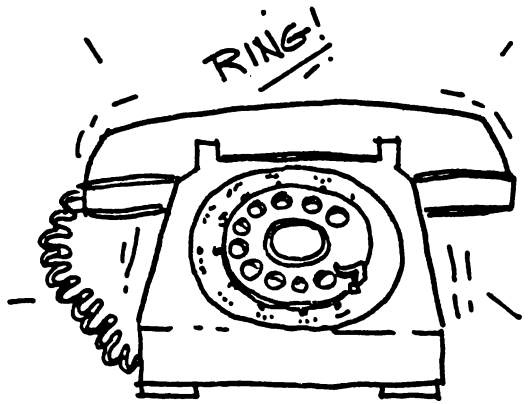
Apenas um caráter foi substituído no string antigo, usando o primeiro caráter de SP.

Você vai descobrir que MID$ é duplamente funcional quando usado com INSTR. Através dessas duas funções, é possível buscar e destruir textos; INSTR busca, MID$ muda ou destrói. O programa a seguir mostrará isso melhor.

```
5  CLS
10 INPUT "ENTRE COM A DATA (DD-MM), ";X$
```

```
20 P=INSTR (X$,"-")
30 IF P=0 THEN 10
40 MID$ (X$,P,1)="/"
50 PRINT X$ "E' MAIS FACIL DE SE LER, NAO E'?"
```

Neste programa INST busca um hífen "-". Quando encontra um, MID$ o substitui por uma barra "/".



FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 12.4

Faça de conta que você trabalha para a Cia. Telefônica nos velhos tempos, quando os centros telefônicos estavam sendo trocados de caracteres alpha para caracteres numéricos. Faça um programa que use MID$ para substituir todas as centrais alpha por números. Preste atenção para limpar espaço suficiente nos strings ou você obterá o erro ?OS ERROR.

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Cada caráter de representação visual pode ser representado por um número. (Falso ou Verdadeiro)

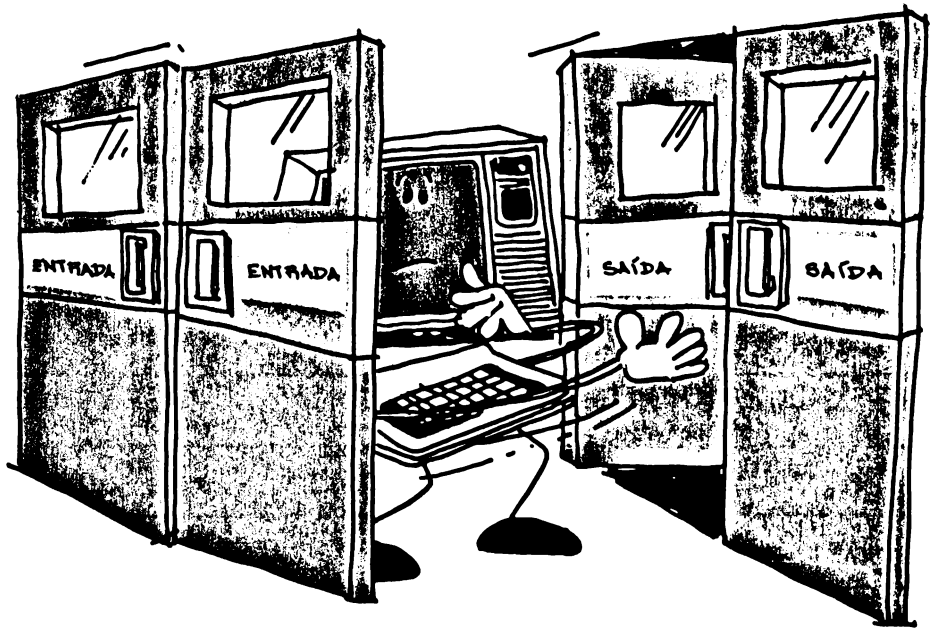
2) A instrução (STRING$) (INSTR) permite que você procure um string centro de outro. (Falso ou Verdadeiro)

3) A instrução MID$ e a função MID$ servem para o mesmo propósito. (Falso ou Verdadeiro)

4) Relacione corretamente a primeira coluna com a segunda:

a - Instrução MID$	1 - Pesquisa dentro de um string
b - INSTR	2 - Cria um string de caracteres
c - STRING$	3 - Substitui parte de um string

CAPÍTULO 13



ENTRADA E SAÍDA

13

ENTRADA E SAÍDA

As instruções de entrada e saída permitem transmitir dados do teclado para o computador, do computador para a televisão e do computador para a impressora.

Não se preocupe, não vamos voltar ao programa "LINHAS", pelo menos por enquanto:

LINE INPUT mensagem, variável string	
mensagem	é a mensagem de orientação. É opcional e, se for usada, deve ser colocada entre aspas.
variável string	é o nome designado para a linha que será fornecida pelo teclado.

LINE INPUT é similar a INPUT, exceto que:

. Quando a instrução é executada, e o computador está esperando pela entrada do teclado,

não aparece nenhum ponto de interrogação.

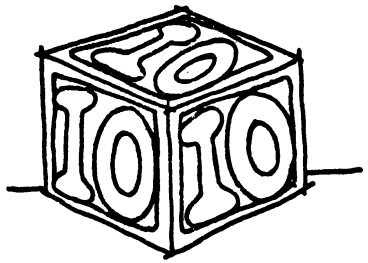
. Cada instrução LINE INPUT pode designar um valor para apenas uma variável.

. Vírgulas e aspas são aceitas como parte do string de entrada.

. Espaços precedentes não são ignorados, eles se tornam parte da variável

LINE INPUT é uma maneira conveniente de se fazer entrada de dados string, sem ter que se preocupar com a entrada acidental de delimitadores (vírgulas, interrogações, dois pontos, etc); tudo é aceito. Na verdade, algumas situações exigem que você dê entrada em vírgulas, aspas e espaços precedentes como parte da informação. A instrução LINE INPUT se aplica muito bem em tais casos. Por exemplo:

```
LINE INPUT X$
```



permite a entrada de X$ sem mostrar qualquer orientação.

```
LINE INPUT "ULTIMO NOME, PRIMEIRO NOME? ";N$
```

mostra uma mensagem de orientação e faz entrada de dados. As vírgulas não concluem o string de entrada. Repare bem que colocamos o ponto de interrogação e o espaço a seguir.

Experimente este programa e terá uma idéia da instrução LINE INPUT.

```
10 CLEAR 300: CLS
20 PRINT TAB (8);"STRING DE ENTRADA": PRINT
30 PRINT: PRINT "*** ENTRE O TEXTO ***"
40 '*** RECEBA O STRING E IMPRIMA-O ***
50 A$= "" 'INICIALIZA A$ COM NULOS
60 LINE INPUT "==>";A$
70 IF A$="" THEN END    'SE O STRING FOR NULO, PARE!
80 PRINT A$
90 GOTO 50
```

PRINT USING

Você provavelmente já deve ter chegado à conclusão de que quanto mais você trabalha com o computador mais condições ele tem de trabalhar para você. Por exemplo, suponha que você quer que o computador crie uma tabela que use números mas não quer ter que digitar os sinais de mais e menos tantas vezes seguidas.

O comando PRINT USING resolve este tipo de problema capacitando o computador a imprimir string e números num formato pré-estabelecido.

Isto pode ser tremendamente útil quando você estiver trabalhando com relatórios contábeis, cheques, tabelas, gráficos e qualquer situação que exija um formato de impressão específico.

PRINT USING formato, lista de itens

formato — é uma expressão string que diz ao computador que formato deve usar quando imprimir cada um dos itens da lista de itens. Consiste de "especificadores de campo" e outros caracteres. É único.

lista de itens — são os dados a serem formatados.

NOTA: PRINT USING não deixa automaticamente espaços em branco ao redor dos números, exceto quando indicado no formato.

Os especificadores de campo a seguir podem ser usados como parte do formato.

Os exemplos na lista de especificadores de campo estão no modo imediato mas podem ser incorporados numa linha de programa.

\# Este sinal especifica a posição de cada dígito localizado no número que você entra. O número de sinais estabelece o campo numérico. Se este campo for maior que o número de dígitos do seu número então as posições não usadas à esquerda do número serão mostradas como espaços e aquelas à direita do ponto decimal serão mostradas como zeros (veja abaixo). Se o campo numérico for muito pequeno para representar um determinado número, o mesmo será mostrado precedido por um sinal "%".

```
PRINT USING "#####"; 66.2
  66
PRINT USING "#"; 66.2
%66
PRINT USING "#.#"; 66.25
%66.3
```

Você pode colocar o ponto decimal em qualquer lugar do campo numérico que foi estabelecido pelo sinal #. O computador irá arredondar automaticamente qualquer dígito à direita do ponto decimal, que não couber no campo.

```
PRINT USING "##.#"; 58.76
58.8
PRINT USING "##.##  ";10.2,5.3,66.789,.234
10.20   5.30   66.79     0.23
```

> NOTA: Neste exemplo, foram colocados dois espaços no string de formato após o último sinal #, para separar os números quando estes fossem impressos.

A vírgula, quando colocada em qualquer posição entre o primeiro dígito e o ponto decimal do número, porá uma vírgula antes de cada conjunto de três dígitos. A fim de evitar um estouro de capacidade (indicado por um sinal de percentagem), é interessante colocar-se uma vírgula a cada terceira posição no campo numérico. O estouro de capacidade ocorre quando o campo não é grande o suficiente para conter o número a ser impresso.

```
PRINT USING "########,";12345678
12,345,678
PRINT USING "########,";123456789
%123,456,789
```

```
PRINT USING "###,###,###";123456789
123,456,789
```

** Quando você coloca dois asteriscos no início do campo numérico, todas as posições não usadas à esquerda do ponto decimal serão preenchidas com asteriscos. Os dois asteriscos vão estabelecer mais duas posições no campo numérico.

```
PRINT USING "**####"; 44.0
****44
```

$ Se o número representa um valor em dinheiro, coloque um sinal "$" na frente do campo numérico. Deste modo, um sinal $ será colocado na frente do número.

```
PRINT USING "$###.##"; 18.6735
$  18.67
```

$$ Dois sinais dólar colocados no começo do campo atuarão como um sinal monetário flutuante. O sinal virá sempre na frente do primeiro dígito.

```
PRINT USING "$$##.## ";18.6735
$18.67
```

**$ Se esses três sinais forem usados no início do campo, então as posições desocupadas à esquerda do número serão preenchidas pelo sinal * e o sinal "$" passará novamente para a primeira posição, precedendo o número.

```
PRINT USING "**$.##";8.333
*$8.33
```

+ Quando o sinal mais (+) for colocado no início ou no final do campo numérico, ele será impresso como um + para números positivos e como menos (-) para números negativos.

```
PRINT USING "+**#####";75200
**+75200
PRINT USING "+###";-216
-216
```

- Quando um sinal menos (-) for colocado no final do campo, fará com que apareça um sinal negativo após qualquer número negativo. Aparecerá um espaço após os números positivos.

```
PRINT USING "####.#-"; -8124.420
8124.4-
```

^^^^ Quatro setas apontando para cima, indicam que o número será impresso em forma exponencial.

```
PRINT USING "##.####^^^^";123456
1.2346E+05
```

! Um ponto de exclamação faz o computador imprimir apenas o primeiro caráter do string fornecido.

```
PRINT USING "!"; "ARITMÉTICA"
A
```

%espaços% Usa-se %espaços% para especificar um campo string de mais de um caráter. A extensão do campo string será dois mais o número de espaços entre os sinais de percentagem.

```
PRINT USING  "%  %";"ABCDEFG"
ABCD
PRINT USING "%   %";"CASA GRANDE"
CASA
```

Execute o programa a seguir para ver como tudo isso funciona:

```
5  CLS
10 A$="**$##,######.##  CRUZEIROS"
20 INPUT "QUAL E' O SEU PRIMEIRO NOME";P$
30 INPUT "QUAL E' O SEU NOME DO MEIO";M$
40 INPUT "QUAL E' O SEU ULTIMO NOME";U$
50 INPUT "ESCREVA A QUANTIA A SER PAGA";P
60 CLS
70 PRINT "PAGUE A ORDEM DE ";
```

Para aprender mais sobre PRINT USING, experimente esse programa:

```
5  CLS
10 INPUT "FORMATO";F$
20 INPUT "LISTA DE ITENS
   ";I
30 PRINT USING F$;I
40 GOTO 5
```

Isto servirá muito bem no caso de dados numéricos . Para dados string, mude I nas linhas 20 e 30 para I$.

```
80 PRINT USING "!! !! ";P$;"."M$;".";
90 PRINT U$
100 PRINT: PRINT USING A$;P
110 GOTO 110
```

A linha 10 define um formato usando **$ para preencher os espaços precedentes com asteriscos e colocar um sinal dólar $ imediatamente antes do primeiro número (Este formato é às vezes usado para evitar que cheques sejam alterados). A linha 10 também estabelece o campo numérico usando o sinal #. Se for fornecido um número pequeno que não preencha todo o campo, os espaços à esquerda serão preenchidos com asteriscos. Temos na linha 10 ainda mais dois especificadores de campo — o ponto decimal e a vírgula.

O ponto decimal é impresso no campo exatamente onde foi especificado. Já que você disse ao computador para tomar mais dois lugares à direita do ponto decimal (para centavos), qualquer número que tenha mais de dois dígitos, será arredondado para dois. Se você fornecer um número com um ou nenhum dígito à direita do ponto decimal, será inserido um zero.

Os pontos de exclamação (!) na linha 80 dizem ao computador para usar apenas os primeiros caracteres em P$ (seu primeiro nome) e M$ (seu nome do meio). Se você incluir U$ junto com P$ e M$, o cheque será nominal a J.L.M. (ou quaisquer que sejam as suas iniciais).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 13.1

Mude o programa de maneira que não haja nenhum asterisco precedente (*) no cheque, apenas um sinal dólar.

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 13.2

Escreva um programa que crie uma tabela mostrando sua receita e despesas mensais. Por enquanto você não precisará relacionar sua situação econômica, afinal de contas não somos da Receita Federal. Encontre apenas o total em dinheiro que entra e sai e faça o computador calcular o lucro líquido (mais ou menos).
Use STRING$ para organizar a tabela, mas certifique-se de que há flexibilidade suficiente para poder usar a tabela por vários meses sem ter que mudar o programa inteiro.

POS é uma função de entrada e saída que permite testar a posição do cursor na tela ou na impressora.

POS (número do dispositivo)

número do dispositivo — 0 (tela) ou -2 (impressora)

POS retorna um número indicando a posição atual do cursor na tela ou a posição do carro na impressora.

```
PRINT TAB(8)POS(0)
```

retorna o número 8 na coluna 8 na linha atual.

> NOTA: O espaço em branco que precede o "8" faz com que ele apareça na coluna 9.

Uma maneira de se usar POS seria desativando a característica de passar automaticamente para a próxima linha ao atingir a última coluna tanto na tela quanto na impressora. Isto muitas vezes traz vantagens porque evita que as palavras sejam quebradas ao meio. Por outro lado, essa função necessariamente diminui a extensão da linha. Execute o programa a seguir para ver POS em funcionamento.

Escolhemos a posição 22 para ser testada, porque ela é 10 posições a menos que a última coluna (a tela tem 32); este espaço é grande o suficiente para se escrever uma palavra comprida.

```
5  CLS
10 A$=INKEY$
20 IF A$="" THEN 10
30 IF POS(0) > 22 THEN IF A$=CHR$(32) THEN
   A$=CHR$(13)
40 PRINT A$;
50 GOTO 10
```

Este programa permite usar o teclado como uma máquina de escrever (exceto que os erros não podem ser corrigidos, a menos que a impressora seja desativada primeiro). POS vigia o final da linha para que nenhuma palavra seja dividida.

Na linha 30, o computador verifica se a posição atual do cursor é maior que a posição da

coluna 22 (a tela tem a largura de 32 colunas). Se o cursor passar da posição 22, o computador começa uma nova linha da próxima vez que você pressionar (ESPAÇO) (CHR$ (32)). Quando o computador decide começar uma nova linha, ele o faz imprimindo um retorno do carro (CHR$ (13)). Na verdade, o computador pressiona (ENTER).

FAÇA VOCÊ MESMO O SEU PROGRAMA - Nº 13.3

Escreva um programa que use POS para espaçar as palavras uniformemente numa única linha.

Você já pensou na sua tela como sendo um dispositivo de "saída" e no seu teclado como sendo um dispositivo de "entrada"? Isso os fazem parecer importantes, não?

Com PRINT, PRINT USING, LINE INPUT e POS, você pode usar números de dispositivos para entradas ou saídas diretas.

Por exemplo, se você digitar:

```
PRINT #-2,USING "###.###";123.45678 (ENTER)
```

a tela permanecerá "silenciosa" enquanto a impressora imprime:

```
123.456
```

Você pode usar qualquer um dos especificadores de campo disponíveis com PRINT#-2,USING.

POS(-2) retorna a posição de impressão atual da impressora (ou seja, a posição atual do carro). Execute o programa a seguir:

```
5  CLS
10 FOR I=1 TO 10
20 PRINT #-2,"*";
30 PRINT "POS. DA IMPRESSORA="; POS(-2)
40 NEXT I
50 PRINT #-2,""
```

A tela mostrará a posição do carro de impressão à medida que o mesmo vai mudando. Repare que a "posição" é calculada internamente, não mecanicamente. A maioria das impressoras não imprimem até que a linha 50 seja executada.

LINE INPUT # pode ser usado quase da mesma maneira, exceto que este comando permite ler uma "linha de dados" de um arquivo em cassete.

LINE INPUT # lê tudo, desde o primeiro caráter até:

. Um caráter de retorno de carro que não venha precedido por um caráter de avanço de linha.

. O final do arquivo.

. O 249º caráter de informação

Outros caracteres encontrados (aspas, vírgulas, espaços em branco precedentes, seqüências de avanço de linha e retorno de carro são incluídos no string. Por exemplo:

```
LINE INPUT #-1,A$
```

dá entrada numa linha de dados de um arquivo em cassete, em A$

O programa a seguir usa LINE INPUT # para contar o número de linhas de qualquer programa armazenado em cassete. Entretanto, o programa tem que ter sido gravado através de CSAVE em formato ASCII (opção A).

```
10  CLEAR 500
20  LINE INPUT "NOME DO ARQUIVO DE INFO
    RMACOES? ";F$
30  K=0    'K E' O CONTADOR
40  OPEN "I",-1,F$
50  IF EOF(-1) THEN 100
60  LINE INPUT #-1,A$
70  K=K+1
80  PRINT A$
90  GOTO 50
100 CLOSE #-1
110 PRINT "O ARQUIVO CONTEM";K;"LINHAS"
```

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) Qual dos comandos a seguir não está relacionado a entrada e saída?

   a - LINE INPUT
   b - POS
   c - HEX$
   d - PRINT USING

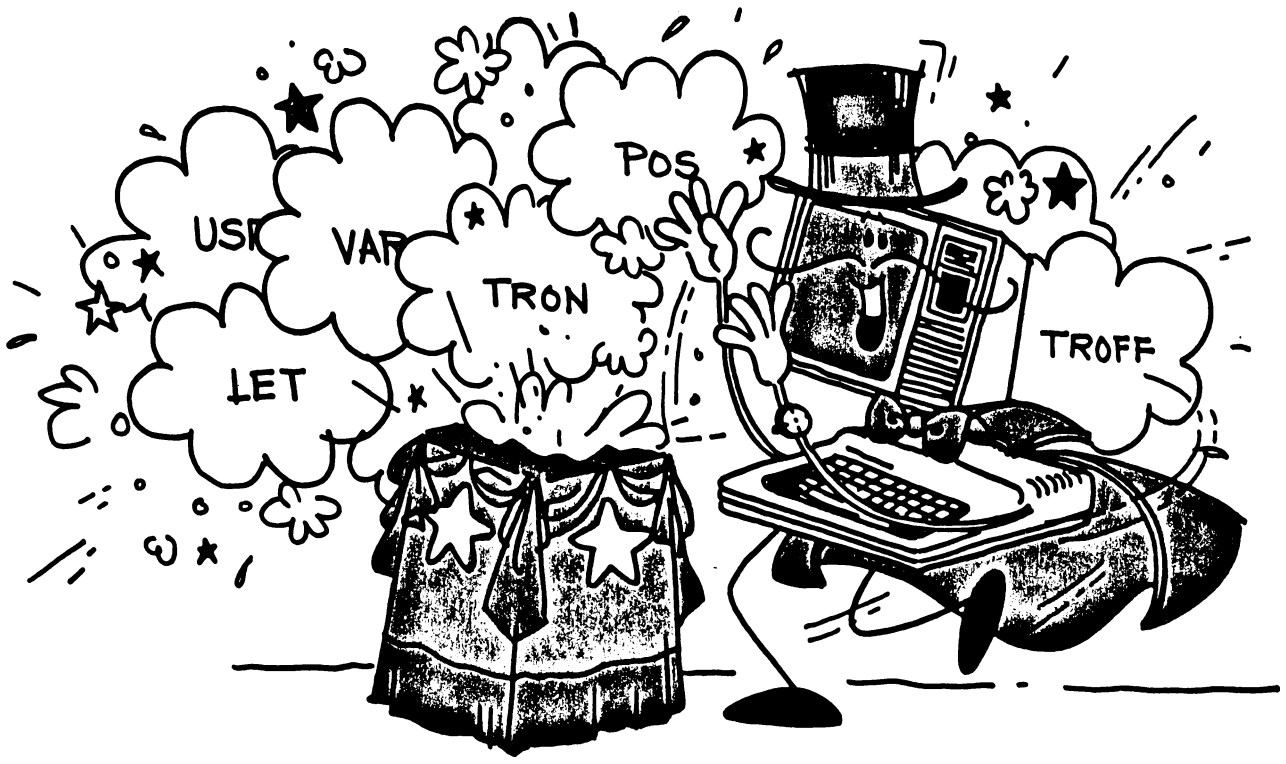
2) Qual dos caracteres a seguir especifica um campo numérico?

   a - %
   b - $
   c - &
   d - #

3) POS é usado para:

   a - Testar a posição atual do cursor.
   b - Posicionar o cursor em um ponto desejado.

CAPÍTULO 14



UM POUQUINHO DE TUDO

14

UM POUQUINHO DE TUDO

Ainda existem algumas características do COLOR BASIC EXTENDIDO com as quais você ainda não teve a oportunidade de trabalhar.

LET

Em muitas linguagens BASIC, a instrução LET tem que ser usada sempre que você atribuir um valor para uma variável (por exemplo, LET X=5). Como você sabe, o COLOR BASIC EXTENDIDO não exige LET. Entretanto, você pode usá-lo sem problemas. Uma das razões para o seu uso é garantir compatibilidade com outras versões do BASIC.

Por exemplo, você pode achar LET muito útil.

```
10 LET A$="A#"
```

que seria o mesmo que:

```
10 A$="A#"
```

NOTA: O COLOR BASIC não reconhece a instrução LET.

TRON / TROFF

TRON e TROFF (que são formas abreviadas para "trace on" e "trace off") são formas de depuração que o ajudam a acompanhar a execução das instruções do programa.

TRON liga um "rastreador" que imprime cada número de linha do programa à medida que o mesmo vai sendo executado. Os números aparecem entre colchetes. TROFF desliga esse "rastreador".

TRON e TROFF são usados da seguinte maneira: TRON (ENTER) e TROFF (ENTER).

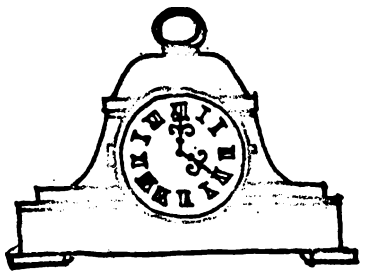
Você já esqueceu tudo que foi visto sobre o programa "LINHAS" dado na seção I? Vamos acompanhar sua execução. Digite TRON (ENTER) e execute este programa:

```
5  PCLS
10 PMODE 3,1
20 SCREEN 1,1
30 LINE (0,0)-(255,191),PSET
40 GOTO 40
```

O computador mostrará:

```
[5]  [10]  [20]  [30]  [40]
OK
```

Tudo isso signfica que o programa executou primeiro a linha 5 e depois a 10, 20 e finalmente a 30. Não se esqueça de digitar TROFF (ENTER) para desligar o "rastreador".



UM CRONÔMETRO (TIMER)

Este computador colorido também tem um "cronômetro" embutido que será muito útil em qualquer programa que necessite de um controle de "tempo". Chamamos essa função de TIMER porque ela mede "tempo" em 1/60 centésimos de segundo (aproximadamente).

No momento em que você liga o computador, o cronômetro começa a contar a partir de zero. Quando ele chega ao 65535, o cronômetro faz uma reciclagem para zero e começa de novo. Isso leva quase 18 minutos. A contagem é interrompida durante as operações com cassete e impressora. O cronômetro faz uma pausa durante essas interrupções.

A função TIMER retorna um valor de 0-65535.

Para saber o conteúdo do contador a qualquer momento, digite

```
PRINT TIMER (ENTER)
```

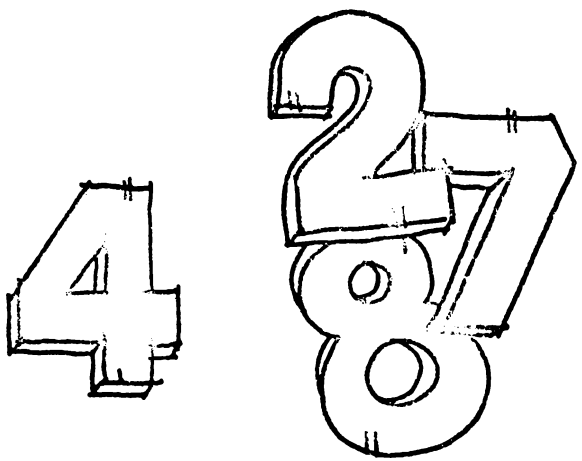
e aparecerá na tela um número de 0 a 65535.

Você também pode ajustar o cronômetro para qualquer "tempo" especificado digitando

TIMER = número (ENTER)

onde número é uma expressão numérica de 0 a 65535.

Para testar a função TIMER (e PRINT@ USING, uma outra função "nova"), execute o programa a seguir, que é chamado "Teste de Matemática". Ele lhe dará um problema matemático e quando você pressionar as letras A, B, C ou D, o computador dirá se a resposta está certa ou errada. Depois disto, ele dirá quanto tempo você levou para responder (Ele faz isso usando a função TIMER).



```
10 DIM CH(3),L$(3) 'CH(N)=ESCOLHAS, L$=FORMATOS DAS RESPOSTAS
20 LI=10:LS=20 'LIMITE INFERIOR E LIMITE SUPERIOR PARA X E Y
30 NV=LS-LI+1
40 P$="QUANTO E' ### + ###?"'FORMATO DA PERGUNTA
50 FOR I = 0 TO 3 'INICIALIZACAO DE CH( )
60 L$(I)=CHR$(I+65)+") ###"
70 NEXT I
80 CLS
90 X=INT(RND(NV)+LI-.5) 'GERA UM NUM. ALEATORIO X ENTRE LI E LS
100 Y=INT(RND(NV)+LI-.5) 'GERA UM NUM. ALEATORIO Y ENTRE LI E LS
110 R=INT(X+Y+.5) 'RESPOSTA CORRETA
130 FOR I = 0 TO 3 'GERA AS OPCOES DA MULTIPLA ESCOLHA
```

```
140 CH(I)=INT(RND(NV)+LI-.5)
150 NEXT I
160 RC=RND(4)-1  'MARCA A RESPOSTA CORRETA
170 CH(RC)=R
180 PRINT @ 32,USING P$;X,Y  'APRESENTACAO DO PROBLEMA
190 FOR LN = 3 TO 6
200 PRINT @ LN * 32+10,USING L$(LN-3); CH(LN-3)
210 NEXT LN
220 TIMER = 0
230 A$="" 'LIMPA O TECLADO
240 A$=INKEY$: IF A$="" THEN 240
250 SV=TIMER  'SE A TECLA FOI PRESSIONADA, GUARDE O CONTEUDO DO TIMER
260 IF A$<"A" IF A$ > "D" THEN 240   'TECLA INVALIDA-REPITA A ESCOLHA
270 K=ASC(A$)-65
280 IF CH(K)=R THEN PRINT "CERTO!": GOTO 300
290 PRINT "ESTA' ERRADO! A RESPOSTA E' ";R
300 PRINT "VOCE LEVOU ";SV/60;" SEGUNDOS"
310 INPUT "PRESSIONE <ENTER> PARA O PROXIMO PROBLEMA"; EN
320 GOTO 80
```

Através de tentativa e erro, mude os limites superiores e inferiores (linha 20) de x e y. Você pode fazê-lo executar uma outra operação matemática além da adição. Ou até mesmo querer que o computador mantenha uma marcação de como você está indo. A pessoa com a menor contagem de tempo, ganha. (Some 5 segundos para cada resposta incorreta).

CONSTANTES OCTAIS E HEXADECIMAIS

O COLOR BASIC EXTENDIDO permite o uso de constantes octa e hexadecimal.

Os números hexadecimais são quantidades representadas n base 16, composta pelos algaris mos de 0-9 e pelas letras A-F. As constantes hexadecima s devem estar entre os limites de 0-FFFF,correspondente aos limites decimais de 0-655:5.

Qualquer número precedido pelo símbolo &H é interpre ado como um número hexadecimal.Por exemplo:

&HA010 &HFE &HD1 &HC &H4000

Números octais são quantidades representadas na base 8, composta pelos algarismos 0-7. Constantes octais devem esta entre os limites 0-177777. Eles são armazenados como 2-bytes inteiros, correspondendo aos limites decimais 0-65535.

Qualquer número precedido pelo símbolo &O ou & interpretado como uma constante octal. Por exemplo, todas as constantes a seguir são octais:

&070 &044 &1777 &17170 &7170 &17 &01234

Constantes octais e hexadecimais são muito convenientes em programas que fazem referên - cia a locações de memória e conteúdos. Para maiores informações, consulte um livro sobre programação em linguagem de máquina. Veja também "Informações Técnicas", no Apêndice O.

É conveniente o uso de números hexadecimais sempre que você trabalhar com programas em linguagem de máquina.

HEX$

Este computador facilita, com o uso de HEX$, a conversão de números de base decimal para hexadecimal. Ele retorna um string que representa um valor hex.

HEX$ (número)
Número é uma variável ou número decimal de 0 a 65535.

Por exemplo, o programa a seguir retorna um valor hexadecimal de qualquer número que você pedir (desde que o número decimal seja menor que 65535).

```
5  CLS
10 INPUT "SE O VALOR DECIMAL DE UM NUMERO E': ";DEC
20 PRINT "SEU VALOR HEXADECIMAL E': "HEX$(DEC)
```

VERIFICAÇÃO DE APRENDIZADO

Escolha a resposta certa:

1) LET deve vir sempre em declarações de atribuição (" = ") (Falso ou Verdadeiro)

2) TIMER faz a contagem em:

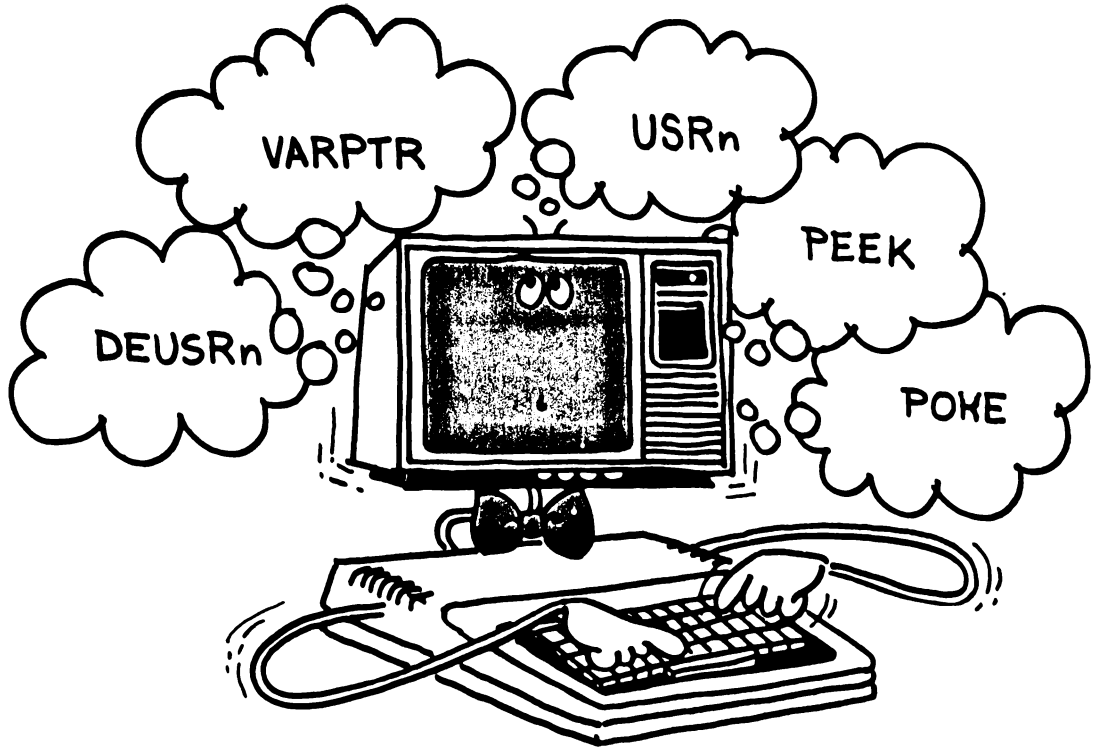
   a - segundos
   b - minutos
   c - horas
   d - todas as respostas anteriores
   e - nenhuma das respostas anteriores

3) Para desligar o "rastreador" do programa, digite .....(ENTER).

4) HEX$ converte os números para os seus ..... equivalentes.

NOTAS

CAPÍTULO 15



ROTINAS EM LINGUAGEM DE MÁQUINA

15

ROTINAS EM LINGUAGEM DE MÁQUINA

Na parte final do Manual COLOR BASIC, apresentamos um pouco de programação em linguagem de máquina. Você deve se lembrar que os programas em linguagem de máquina são escritos na "linguagem-nativa" do computador. É possível passar para programas em linguagem de máquina usando as características do COLOR BASIC EXTENDIDO, que serão discutidos neste capítulo.

Por que você poderia querer usar linguagem de máquina? Principalmente pelo fato dessa linguagem ser mais rápida (chega a ser até 1000 vezes mais rápida que a linguagem BASIC), exige menos espaço de memória e, com ela, você pode desenvolver operações que não seriam possíveis com o COLOR BASIC EXTENDIDO.

Todos os princípios de linguagem de máquina vistos no Manual COLOR BASIC podem ser usados neste manual. Portanto, se tiver qualquer dúvida a esse respeito, você encontrará informações mais completas naquele manual. Aqui nos limitaremos a ver as novas características que estão disponíveis com o COLOR BASIC EXTENDIDO.

USRn

USRn é uma das funções de linguagem de máquina mais usadas. Ela é exatamente igual ao USR que você viu no Manual COLOR BASIC, exceto que com o USRn você pode chamar até 10 subrotinas em linguagem de máquina,que tenham sido previamente carregadas e definidas por DEF USR. Você pode então continuar a execução do seu programa em BASIC.

Quando uma chamada USR é encontrada numa instrução, o controle vai para o endereço definido na instrução DEF USRn. Este endereço especifica o ponto de entrada para a sua rotina em linguagem de máquina. O controle pode ser retornado para a instrução após USRn. No COLOR BASIC você teve que colocar o endereço inicial em RAM, mas você não pode usar este método no COLOR BASIC EXTENDIDO.

USRn (argumento)	
n	determina uma das chamadas USR disponíveis e é um número de 0 a 9. É opcional e,se for omitido, será usado 0.
argumento	é uma expressão numérica ou string.

ALOCACAO DE MEMORIA PARA FUNCOES USR

A instrução CLEAR deve ser usada no início de um programa para reservar memória para funções USR. Por exemplo:

```
CLEAR 50, 12000
```

Reserva RAM a partir do endereço 12001.

FUNCOES DE CARREGAMENTO USR

As funções USR podem ser inseridas na memória com o auxílio do comando POKE ou carregadas de cassete usando-se CLOADM. A instrução DLOAD pode ser usada para transferir funções USR de um outro computador.

O USO DA PILHA DE MEMORIA

Uma função USR que necessite mais de 30 bytes de área de armazenamento em pilha, deve prover sua própria área de empilhamento. Isto é feito armazenando-se o ponteiro da pilha do COLOR BASIC antes da entrada na função USR, estabelecendo um novo ponteiro da pilha e retornando o valor original do ponteiro da pilha do COLOR BASIC antes de voltar para BASIC.

DEFININDO AS FUNCOES USR

DEF USR é usado para definir o endereço de entrada de uma função USRn:

DEF USRn = endereço	
n	é um dígito de 0 a 9; se omitido, será usado 0.
endereço	especifica o endereço de entrada para uma rotina em linguagem de máquina e deve estar entre 0 e 65535.

O valor dessa expressão é passado para a função USR como seu argumento (Veja Argumentos da Função USR para informações mais detalhadas a respeito da passagem de argumentos para funções USR).

RETORNANDO DA FUNCAO USR PARA BASIC

Para se retornar da função USR para a função BASIC, deve ser executada uma instrução RTS ou uma sequência de instruções equivalente. O ponteiro da pilha deve ser atualizado com o seu valor de entrada, antes de se retornar ao BASIC. Os valores dos registradores A, B, X e CC não precisam ser preservados pela função USR.

As funções USR sempre retornam um valor ao BASIC. A menos que a função USR designe explicitamente um valor de retorno, o valor retornado é o do argumento passado para a função USR. Para maiores detalhes, veja "Valores de Retorno para Basic".

ARGUMENTOS DA FUNCAO USR

O argumento passado para a função USR é o valor da expressão argumento especificada na chamada USR. Ao se entrar na função USR, o conteúdo do registrador A indica o tipo de argumento, como abaixo:

A = zero (argumento numérico)
A = não-zero (argumento string)

Se o argumento for numérico, o registrador X conterá um ponteiro para o Acumulador de Pon

to Flutuante (FAC) que contém o argumento. É possível tornar o argumento um inteiro, chamando a rotina INTCNV do BASIC a partir da função USR (INTCNV = X'B3ED'). Se o argumento for um string, INTCNV causa um erro 'TM' e o controle retorna para BASIC. Se o argumento for um número de ponto flutuante fora do intervalo -32768 a +32767, INTCNV causa um erro de estouro de capacidade e o controle retorna para BASIC. INTCNV retorna o complemento a dois de um inteiro de 16 bits no registrador D.

Para argumentos numéricos ((A)=0), FAC contém o expoente, FAC+1 contém o bit mais significativo (MSB)...FAC+4 contém o bit menos significativo (LSB) e FAC+5 contém o sinal da mantissa. O expoente é um inteiro de oito bits com sinal com 128(na base 10) adicionado a ele. Um expoente zero significa que o número é zero; neste caso, a mantissa é insignificante. A mantissa é armazenada em formato normalizado com o bit mais significante do byte mais significativo assumido como sendo igual a 1. Este bit pode então ser usado para indicar o sinal: 0 para mantissa positiva e 1 para mantissa negativa.

Para um argumento string, o registrador X aponta para um descritor de 5 bytes. O primeiro byte do descritor contém o comprimento (em caracteres) do string. O terceiro e quarto bytes contêm o endereço do primeiro byte do string. O segundo e o quinto são reservados para o computador e não estão disponíveis para seu uso.

Um ponteiro para uma variável BASIC pode ser passado por uma função USR usando-se a função VARPTR na expressão argumento da chamada da função USR. Sua sintaxe é:

VARPTR (nome de variável)

Por exemplo:

X = USR0(VARPTR)(A)).

passa um ponteiro para a variável A. INTCNV pode ser chamada para obter o ponteiro. É tarefa da função USR determinar o tipo da variável. Esta informação não é passada para a função USR pelo BASIC.

Para variáveis numéricas, o ponteiro aponta para um campo de 5 bytes em ponto flutuante , sendo que o primeiro byte é o expoente e os quatro bytes restantes formam a mantissa.

Valores em ponto flutuante são armazenados na tabela de variáveis num formato ligeiramente diferente dos que são armazenados em FAC. O bit mais significativo do byte mais significativo da mantissa é assumido como sendo um 1 (já que mantissas de pontos flutuantes são sempre normalizadas), e a posição deste bit é usada para armazenar o sinal da mantissa. O número é positivo se o bit de sinal for um 0 e negativo se o bit de sinal for um 1.

Para variáveis string, o ponteiro aponta para um descritor de string de 5 bytes. O primeiro byte contém o comprimento do string e o terceiro e o quarto bytes contêm o endereço do primeiro caráter no string. O segundo e o quinto bytes são reservados para o computador e não estão disponíveis para seu uso.

É possível passar um ponteiro de um vetor como o argumento de uma função USR. Deste modo a função USR pode ter acesso a qualquer dos elementos do vetor. Os valores dos elementos são armazenados na memória da seguinte maneira (ordenados da memória inferior à superior):

. Valor do primeiro elemento da última dimensão
. Valor do último elemento da última dimensão
. Valor do primeiro elemento da primeira dimensão
. Valor do último elemento da primeira dimensão

O comprimento de cada elemento é 5 bytes. Valor é o valor designado para as variáveis dentro do string.

Outro método de passar valores para uma função USR é colocar (usando POKE) os valores nas posições de memória reservadas para o uso da função USR.

RETORNO DE VALORES PARA BASIC

Uma função USR retorna sempre pelo menos um valor para BASIC, o valor da função. Se a rotina de USR não retorna explicitamente este valor, o valor retornado é o valor do argumento passado para a função USR. Geralmente, o tipo de valor retornado é o mesmo do argumento da função USR. Neste caso, a função USR retorna seu valor para FAC ou para o descritor de string apontado por X, no mesmo formato do argumento.

Independentemente do tipo do argumento, um valor inteiro pode ser retornado através do carregamento de D com o complemento a dois de um inteiro de 16 bits e da chamada da rotitina GIVABF do BASIC (GIVABF = X'B4F4') para retornar para BASIC.

Valores adicionais podem ser retornados para BASIC através da modificação de valores de variáveis do BASIC. Deve-se ter extrema cautela ao se retornar valores string para BASIC. A advertência a seguir se aplica quando se retorna um string como o valor da função,bem co

mo quando se modifica uma variável string do BASIC.

O comprimento de um string pode ser modificado com a mudança do byte de comprimento no descritor do string. Os strings podem ser encurtados desta forma, mas não se deve nunca tentar aumentar o comprimento de um string com a função USR. Se o comprimento do string a ser retornado para BASIC não for conhecido no momento da chamada, o string deve ser forçado a ter o comprimento máximo de 255 caracteres. Por exemplo:

```
A$ = USR0 (STRING$(" ",255))
```

passa um string de 255 espaços em branco para a função USR. Esta função pode então colocar um string de até 255 caracteres dentro da memória apontada pelo descritor de string e encurtar esse string caso seja necessário.

O endereço inicial de um string pode ser modificado mudando-se o ponteiro de 2 bytes no descritor de string. Entretanto, o novo endereço inicial deve ser, geralmente, o de alguma locação de memória incluída no string original. Isto é, o endereço inicial pode ser mudado para qualquer um dos endereços OSA até OSA+OSL -1, onde OSA é o endereço inicial original e OSL é o comprimento do string original. Também é aceitável se permutar os endereços iniciais de dois strings. Deve-se tomar cuidado para que dois strings não se interceptem.

É possível para um descritor de string indicar o dado em string que está dentro do programa BASIC. Isto só pode ocorrer quando um string é definido como um string literal. Por exemplo, A$ = "ABC" e USR0("DEF") causam ambos a criação de descritores que apontam para dentro do texto do programa em BASIC.

Se uma função USR fosse usada para modificar tal string, o programa BASIC seria na verdade mudado. Este problema pode ser evitado acrescentando-se o string nulo ("") a quaisquer literais string que venham a ser modificados por uma função USR. Por exemplo:

```
A$ = "ABC" + ""
```

irá fazer com que o string seja copiado para dentro do espaço de string, onde ele poderá ser modificado por uma função USR com toda a segurança.

Finalmente, é possível retornar valores adicionais para BASIC, através do armazenamento dos mesmos dentro das posições de memória alocadas para uso pela função USR. O programa BASIC pode então ter acesso a esses valores usando a função PEEK.

APÊNDICES

APÊNDICE A

RESPOSTAS DAS VERIFICAÇÕES DE APRENDIZADO

CAPÍTULO 1

1 - a
2 - b
3 - a
4 - d

CAPÍTULO 2

1 - d
2 - RESET
3 - b
4 - c
5 - X... Y...

CAPÍTULO 3

1 - c
2 - PCLS
3 - 8
4 - a,2
    b,3
    c,2
    d,3
    e,2
5 - vermelho, azul, amarelo, verde
6 - e requer a maior memória e a a menor.

CAPÍTULO 4

1 - a
2 - e
3 - d

CAPÍTULO 5

1 - c
2 - c
3 - c

APÊNDICE A

CAPÍTULO 6

1 - 8 e 2 são conjuntos de cores diferentes
2 - Falsa
3 - Falsa

CAPÍTULO 7

1 - Mx,y
2 - c
3 - B
4 - c
5 - d
6 - Verdadeira
7 - c

CAPÍTULO 8

1 - d
2 - b

CAPÍTULO 9

1 - b
2 - a
3 - L4;
4 - f

CAPÍTULO 10

1 - a
2 - c
3 - Falsa
4 - e

CAPÍTULO 11

1 - Verdadeira
2 - INSTR
3 - Falsa
4 - a,3 / b,1 / c,2

APÊNDICE A

CAPÍTULO 13

1 - c
2 - d
3 - a
4 - a,0
    b,-1
    c,-2

CAPÍTULO 14

1 - Falso
2 - e
3 - TROFF
4 - Hexadecimal

APÊNDICE B

RESPOSTAS DOS EXERCÍCIOS

Exercício de Programação nº 1.2

```
5 PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 X = RND(256)-1
40 Y = RND(192)-1
50 C = RND(9)-1
60 PSET(X,Y,C)
70 GOTO 30
```

Exercício de Programação nº 2.1

```
5 PMODE 1,1
10 PCLS
20 SCREEN 1,1
25 LINE (0,0)-(255,191),PSET
30 LINE (0,191)-(255,0),PSET
35 LINE (0,0)-(255,191),PSET,B
40 GOTO 40
```

Exercício de Programação nº 2.2

```
5 PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 LINE (72,168)-(200,72),PSET,B
     'CONTORNO
40 LINE (72,72)-(136,36),PSET
     'TELHADO
45 LINE (200,72)-(136,36),PSET
     'TELHADO
50 LINE (120,168)-(152,100),PSET
,B   'PORTA
55 LINE (152,60)-(168,36),PSET,B
F    'CHAMINE
60 LINE (165,128)-(191,100),PSET
,B   'JANELA
65 LINE (178,120)-(178,100),PSET
     'JANELA
70 LINE (165,114)-(191,114),PSET
     'JANELA
75 LINE (85,128)-(111,100),PSET,
B    'JANELA
80 LINE (85,114)-(111,114),PSET
     'JANELA
85 LINE (98,100)-(98,128),PSET
     'JANELA
90 GOTO 90
```

Exercício de Programação nº 2.3

```
5 PMODE 1,1
10 PCLS
20 SCREEN 1,1
30 Y=0
40 FOR X=0 TO 200 STEP 10
50 OY=Y
60 Y=30-OY
70 LINE(X,100-Y)-(X+10,100-OY),P
SET
80 NEXT
90 GOTO 90
```

Exercício de Programação nº 3.1

```
PMODE 2,(1-3)
PMODE 1,(1-3)
PMODE 0,(1-4)
```

Exercício de Programação nº 3.2

Faça as seguintes alterações:

```
22 PCOPY 4 TO 3
32 PCOPY 3 TO 2
42 PCOPY 2 TO 1
```

e elimine as linhas 11, 21 e 31.

APÊNDICE B

Exercício de Programação nº 3.3

```
10 PCLEAR 8
20 PMODE 4,1
25 PCLS
30 SCREEN 1,1
40 LINE (0,0)-(255,191),PSET
45 FOR Y = 1 TO 20: NEXT Y
50 PMODE 4,2
55 SCREEN 1,0
60 LINE (0,0)-(255,191),PSET
65 FOR Z = 1 TO 20: NEXT Z
70 PMODE 0,3
75 SCREEN 1,1
80 LINE (0,0)-(255,191),PSET
85 FOR A = 1 TO 20:NEXT A
90 PMODE 1,4
95 SCREEN 1,1
96 PCLS
100 LINE (0,0)-(255,191),PSET
105 FOR R = 1 TO 20: NEXT R
110 GOTO 20
```

Exercício de Programação nº 4.1

```
1 Y = -1
5 CLS
10 PRINT @ 194,"VOCE QUER VER UM
 QUADRADO?"
20 FOR X = 1 TO 1000: NEXT X
30 PMODE 1,1
35 PCLS
40 SCREEN 1,Y+1
60 LINE (75,150)-(150,75),PSET,B
70 FOR X = 1 TO 1000:NEXT X
75 Y = -Y
80 GOTO 5
```

Exercício de Programação nº 5.1

```
10 PMODE 4,1
20 PCLS
30 SCREEN 1,0
40 FOR RAIO = 1 TO 100 STEP 10
50 CIRCLE (128,96),RAIO
60 NEXT RAIO
70 GOTO 70
```

Exercício de Programação nº 5.3

```
5 PMODE 4,1
10 PCLS
20 SCREEN 1,0
30 CIRCLE (200,40),30,,1,.13,.63
40 CIRCLE (230,10),52,,1,.29,.48
50 GOTO 50
```

Exercício de Programação nº 5.4

```
5 PMODE 1,1
10 SCREEN 1,0
15 PCLS 3
20 COLOR 1,0
25 CIRCLE (200,40),30,,1,.13,.63
   'LUA
30 CIRCLE (230,10),52,,1,.29,.48
   'LUA
35 LINE (100,185)-(180,125),PSET
,B'CONTORNO
40 LINE -(140,85),PSET
   'TELHADO
45 LINE -(100,125),PSET
   'TELHADO
55 LINE (110,160)-(125,130),PSET
,B'JANELA
60 LINE (155,160)-(170,130),PSET
,B'JANELA
70 LINE (130,130)-(149,185),PSET
,B'PORTA
75 PSET (134,157,1)
   'MACANETA DA PORTA
80 LINE (160,105)-(160,90),PSET
   'CHAMINE
85 LINE -(175,90),PSET
   'CHAMINE
90 LINE  (175,115),PSET
   'CHAMINE
100 ' A FUMACA COMECA AQUI
105 X=167:Y=89
   'CENTRO DO CIRCULO
110 SP=0: EP=0
   'PONTA INICIAL E FINAL DO CIRC
```

APÊNDICE B

```
ULO
115 FOR R= 1 TO 50 STEP .05
  'MEIO DO CIRCULO
120 EP=EP+.02: IF EP > 1 THEN EP
 = 0
125 CIRCLE (X+R, Y-R),R,4,1,SP,E
P 'FUMACA
130 NEXT R
200 GOTO 200
```

Exercício de Programação nº 6.1

Acrescente a linha 65:

```
65 PAINT (150,100),8,B
```

ou elimine a linha 30.

Exercício de Programação nº 6.3

```
5 PMODE 1,1
10 PCLS
15 SCREEN 1,0
20 PCLS 3
25 COLOR 1,0
30 CIRCLE (200,30),15
35 PAINT (200,30),2,1
40 LINE (100,185)-(130,125),PSET
,B
45 LINE -(140,90),PSET
50 LINE -(100,125),PSET
55 PAINT (135,115),4,1
60 LINE (110,160)-(125,130),PSET
,B
65 LINE (155,160)-(170,130),PSET
,B
70 PSET (134,157,1)
75 PAINT (120,180),0,1
80 LINE (130,130)-(149,185),PSET
,B
85 LINE (101,135)-(41,185),PSET,
B
90 LINE (91,140)-(51,185),PSET,B
95 PAINT (55,138),0,1
100 PAINT (89,183),4,1
105 FOR X = 1 TO 500: NEXT X
110 PAINT (89,183),2,1
115 FOR X = 1 TO 500: NEXT X
120 PAINT (89,155),4,1
140 GOTO 110
```

Exercício de Programação nº 7.1

```
5 PMODE 4,1
10 PCLS
20 SCREEN 1,0
30 DRAW "BM68,116;E20;DE20;E20;F
20;BF20;F20;L40;BL40;L40;BU40;R4
0;BR40;R40;G20;BG20;G20;H20;BH20
;H20;BM128,96;NU40;ND40;NE20;NF2
0;NG20;NH20;NL40;R40"
40 GOTO 40
```

A estrela que você criou não é tão bonita quanto esta porque ainda não explicamos como usar os parâmetros B e N. Não se preocupe, estes parâmetros lhe serão apresentados ainda neste capítulo.

Exercício de Programação nº 7.2

```
5 PMODE 4,1
10 PCLS
20 SCREEN 1,1
25 DRAW "BM40,80;U40;R40;D40;L40
"
30 DRAW "BM+20,20;U40;R40;D40;L4
0"
40 LINE (60,100)-(40,80),PSET
50 LINE (60,60)-(40,40),PSET
60 LINE (100,60)-(80,40),PSET
70 LINE (100,100)-(80,80),PSET
80 GOTO 80
```

Exercício de Programação nº 7.3

```
5 PMODE 4,1
10 PCLS
20 SCREEN 1,1
25 DRAW "BM50,50L30D30R30D30L30"
30 DRAW "BM90,50D60R30U60"
40 DRAW "BM140,50D60R30BU60L30D3
0R30"
50 GOTO 50
```

APÊNDICE B

Exercício de Programação nº 7.4

```
5 PMODE 4,1
10 PCLS
20 SCREEN 1,0
30 DRAW "BM98,96;NU80;NE56;NR80;
NF56;ND80;NG56;NL80;NH56"
40 CIRCLE (98,96),80,1,1,.125,1
50 CIRCLE (135,110),80,1,1,1,.12
5
60 LINE (135,110)-(190,167),PSET
70 LINE (135,110)-(213,110),PSET
80 GOTO 80
```

Exercício de Programação nº 7.5

```
1 CLEAR 2500
5 DIM AZ$(25)
6 FOR LE = 0 TO 25
10 READ AZ$(LE)
15 NEXT LE
20 NC$="BR4BU7" 'PROXIMO CARATER
25 NL$="BD4" 'PROXIMA LINHA
30 BS$="BL9" 'VOLTA UM CARATER
35 HM$="BM0,10" 'POSICAO INICIAL
100 CW=6: CH=8 'TAMANHO DO PONTO
110 R1=7: R24=191 'POSICAO DA LI
NHA
120 C1=8: C42=247 'POSICAO DA CO
LUNA
125 CC=1: CL=1 'POSICAO ATUAL LI
NHA COLUNA
200 PCLS
220 SCREEN 1,0
225 DRAW "S4"
230 DRAW HM$
250 A$=INKEY$: IF A$="" THEN 250
260 IF "A">A$ OR "Z" < A$ THEN 2
50
262 CC=CC+ 1
265 IF CC>27 THEN DRAW  NL$: FOR
 I = 1 TO 27: DRAW BS$: NEXT I:C
C=1: GOTO 270
269 DRAW NC$
270 DRAW AZ$(ASC(A$)-65)
290 GOTO 250
1000 ' A
1010 DATA BD1D6U4NR5U2E1R3F1D6
1020 ' B
1030 DATA ND7R4F1D1G1NL4F1D2G1NL
4BR1
1040 ' C
1050 DATA BD1D5F1R3E1U1BU3U1H1L3
G1BD6BR5
1060 ' D
1070 DATA D7R4E1U5H1L4BD7BR5
1080 ' E
1090 DATA NR5D3NR4D4R5
1100 ' F
1110 DATA NR5D3NR4D4BR5
1120 ' G
1130 DATA BD1D5F1R3E1U2NL2BU2U1H
1L3G1BD6BR5
1140 ' H
1150 DATA D7U4R5NU3D4
1160 ' I
1170 DATA R4L2D7L2R4BR1
1180 ' J
1190 DATA BD5D1F1R3E1U6D7
1200 ' K
1210 DATA D7U4R3E2NU1G2F2D2
1220 ' L
1230 DATA D7R5
1240 ' M
1250 DATA ND7R2ND7R2D7BR1
1260 ' N
1270 DATA D1ND6E1R3F1D6
1280 ' O
1290 DATA BD1D5F1R3E1U5H1L3G1BD6
BR5
1300 ' P
1310 DATA ND7R4F1D2G1L4BD3BR5
1320 ' Q
1330 DATA BD1D5F1R3E1U5H1L3G1D4B
R3F2
1340 ' R
1350 DATA ND7R4F1D1G1NL4F1D3
1360 ' S
1370 DATA BD1D1F1R3F1D2G1L3H1BU5
E1R3F1BD6
1380 ' T
1390 DATA R4L2D7BR3
1400 ' U
1410 DATA D6F1R3E1U6BD7
1420 ' V
1430 DATA D5F2E2U5BD7BR1
1440 ' W
1450 DATA D7R2NU6R2U7BD7BR1
1460 ' X
1470 DATA D1F5D1BL5U1E5U1BD7
1480 ' Y
1490 DATA D2F2ND3E2U2BD7BR1
1500 ' Z
1510 DATA R5D1G5D1R5
```

APÊNDICE B

Exercício de Programação nº 7.6

```
5 PMODE 3,1
10 PCLS
15 SCREEN 1,0
20 DRAW "BM50,170;U80;NG30;E80;F
80;NF30;D80;L50;U70;L50;D70;L60"
25 LINE (50,170)-(170,170),PSET
30 LINE (110,170)-(160,170),PSET
35 FOR X = 1 TO 500: NEXT X
40 LINE (110,170)-(160,170),PRES
ET
45 LINE (120,180)-(120,110),PSET
50 LINE (160,100)-(135,110),PSET
55 LINE (160,170)-(125,180),PSET
60 LINE (120,180)-(120,110),PRES
ET
65 LINE (160,100)-(125,110),PRES
ET
70 LINE (160,170)-(125,180),PRES
ET
75 DRAW "BM110,170;BU70;BR50;G25
;D70;E25"
80 CIRCLE (130,125),10,,1,,135,.
7
85 DRAW "BM130,130;D15;D15;G10;E
10;U15;L10"
90 LINE (120,145)-(120,135),PSET
91 FOR X = 1 TO 60: NEXT X
95 LINE (120,145)-(120,135),PRES
ET
96 FOR X = 1 TO 120: NEXT X
100 LINE (120,145)-(110,145),PSE
T
101 FOR X = 1 TO 60: NEXT X
105 LINE (120,145)-(110,145),PRE
SET
106 FOR X = 1 TO 60: NEXT X
110 LINE (120,145)-(120,135),PSE
T
120 FOR X = 1 TO 120: NEXT X
121 CIRCLE (130,125),10,1
122 DRAW "BM130,130;C1;D30;G10;E
10;U15;L10"
125 DRAW "BM110,170;BU70;BR50;C1
;G25;D70;E25;"
130 COLOR 4,1
135 LINE (120,180)-(120,110),PSE
T
140 LINE (160,100)-(125,110),PSE
T
145 LINE (160,170)-(125,180),PSE
T
150 LINE (120,180)-(120,110),PRE
SET
155 LINE (160,100)-(125,110),PRE
SET
160 LINE (160,170)-(125,180),PRE
SET
165 LINE (110,170)-(140,170),PSE
T
170 FOR X = 1 TO 500: NEXT X
175 GOTO 20
```

Exercício de Programação nº 8.1

```
5 PCLEAR 4
10 PMODE 4,1
15 PCLS
20 SCREEN 1,1
25 DIM V(35,35)
30 X=10: Y=10
35 DRAW "BM10,10; S2; H10;R15;F1
0;R20;F10;G10;L20;G10;L15;E10;U2
0;D4;NL8;D4;NL12;D4NL16;D4;NL12;
D4;NL8"
40 GET (X-X,Y-Y)-(X*3.5,Y*3.5),V
,G
45 A$=INKEY$: IF A$="" THEN 45 '
 PRESSIONE QUALQUER TECLA PARA C
OMECAR
50 PCLS
55 FOR A = 10 TO 200 STEP 5
60 PUT (X-A,Y)-(X+A+35,Y+35),V,P
SET :NEXT A
70 PCLS: GOTO 55
```

Note que usamos a opção GET e PUT. Se você quer que o foguete voe mais rápido, elimine a opção e use PMODE 3.

Exercício de Programação nº 9.1

```
5 CLS
10 FOR N   12 TO 1 STEP -1
20 PLAY STR$(N)
30 NEXT N
```

APÊNDICE B

Exercício de Programação nº 9.2

```
100 A$="T5;C;E;F;L1;G;P4;L4;C;E;
F;L1;G"
105 B$="P4;L4;C;E;F;L2;G;E;C;E;L
1;D"
110 C$="P4;L4;D+;L8;E;G;E;P8;L4;
C;L8;D;D+"
115 D$="L4;E;C;L2;O3;C;L8;O3;D;L
8;O2;B-"
120 E$="G;E;L4;G;L1;F;P4;L8;G;F;
E;F"
125 F$="L2;G;E;L4;C;L8;D;D+;E;G;
L4;A;L1;O3;C"
130 X$="XA$;XB$;;XC$;XD$;XE$;XF$
;"
```

Exercício de Programação nº 11.1

```
5 CLS
10 FOR NUM = 1 TO 10
20 PRINT NUM ^ 2
30 NEXT NUM
```

Exercício de Programação nº 11.2

```
5 CLS
10 FOR NUM = 100 TO 1 STEP -10
20 PRINT SQR(NUM)
30 NEXT NUM
```

Exercício de Programação Nº 11.3

```
5 CLS
10 FOR A = -180 TO 179 STEP 10
15 RD=A/57.29577951
30 CP=COS(RD)*14+16.5 'POSICAO D
O COSSENO
40 SP=SIN(RD)*14+16.5 'POSICAO D
O SENO
50 IF SP<=CP THEN 70
60 PRINT TAB(CP); "C";TAB(SP);"S
": GOTO 80
70 PRINT TAB(SP);"S";TAB(CP);"C"
80 NEXT A
90 GOTO 10
```

Exercício de Programação nº 11.4

a)
```
?LOG (1003)
6.91075079
```

b)
```
?LOG (74.9865)
4.3173061
```

c)
```
?LOG (3.334285)
1.21023863
```

Exercício de Programação nº 11.5

```
5 CLS
10 INPUT "QUAL NUMERO "; NUM
15 X=LOG(NUM)/LOG(10)
20 PRINT "O LOG NA BASE 10 DE" N
UM "E'" X : GOTO 10
```

a) 7.00890077 E -11

Nota: O log de 1 em qualquer base é 0. A resposta que o computador imprime , deve-se a erros de precisão. Qualquer computador produz este tipo de erro.

b) 1
c) 2
d) 2.69897001
e) -1
f) 3.00043408

Exercício de Programação nº 11.6

```
DEFNR(X) = X * 57.29577951

5 CLS
10 DEF FNC(X) = X ^ 3
20 INPUT "QUAL O NUMERO QUE VOCE
 QUER ELEVAR AO CUBO";X
30 X=FNC(X)
40 PRINT X
50 FOR A= 1 TO 75 : NEXT A
60 GOTO 5
```

APÊNDICE B

Exercício de Programação nº 12.1

```
5 CLS
10 X$=STRING$(30,"-")
20 FOR X=64 TO 416 STEP 64
30 PRINT @ X,X$
40 PRINT @  97, "JOAO"
41 PRINT @ 161, "MARIA"
42 PRINT @ 225, "PEDRO"
43 PRINT @ 287, "MARTA"
51 PRINT @  39, "MAT."
52 PRINT @  45, "PORT."
53 PRINT @  53, "INGLES"
61 PRINT @ 103, "X"
62 PRINT @ 175, "X"
63 PRINT @ 231, "X"
64 PRINT @ 311, "X"
70 NEXT X
80 GOTO 80
```

Exercício de Programação nº 12.2

```
5 CLS
10 X$ = "ABCDEE"
20 Y$ = "E"
30 PRINT INSTR(X$,Y$); INSTR(4,X
$,Y$)
```

Exercício de Programação nº 12.3

```
15 X=1
20 X$="JOAO SILVA,PRES. VARGAS 1
7,RIORJ*22411:OSWALDO MONTENEGRO
,RUA PIO XII,SAOPAULOSP*01011:MA
RCEL FONTOURA,RUA MACEIO 46,RIOR
J*22411:MARIA DAS GRACAS,SQS 114
BL.A AP.107,BRASILIADF*45230"
50 P= INSTR(X,X$,A$): PRINT P
60 IF P<>0 THEN X=P+1 : GOTO 50
```

Exercício de Programação nº 12.4

```
10 DIM TEL$(26)
20 FOR I=0 TO 25
30 READ TEL$(I):NEXT I
40 PRINT "FORNECA O NUMERO DO TE
LEFONE NA FORMA ANTIGA"
50 INPUT N$
60 IF N$="" THEN 40
70 FOR I=1 TO LEN(N$)
80 C$=MID$(N$,I,1)
90 IF C$<"A" OR C$>"Z" THEN 120
100 C$=TEL$(ASC(C$)-65)
110 MID$(N$,I)=C$
120 NEXT I
130 PRINT "ESTA E' A NOVA FORMA
= "; N$
140 REM A B C D E F
150 DATA "2","2","2","3","3","3"
160 REM G H I J K L
170 DATA "4","4","4","5","5","5"
180 REM M N O P Q R
190 DATA "6","6","6","7","Q","7"
200 REM S T U V W X
210 DATA "7","8","8","8","9","9"
220 REM Y Z
230 DATA "9","Z"
```

Exercício de Programação nº 13.1

```
10 A$="$$###,######.## CRUZEIROS"
```

APÊNDICE B

Exercício de Programação nº 13.2

```
5 CLS
10 INPUT "RECEITA"; R
15 INPUT "DESPESA"; D
20 N = R-D  ' LUCRO OU PERDA
25 A$ = "$$#####.##"
30 B$ = "$$#####.##"
35 C$ = "+$$#####.##"
40 CLS: PRINT @ 33, "RELATORIO E
CONOMICO MENSAL"
45 PRINT @ 96, STRING$ (32,"-")
50 PRINT @ 160, "RECEITA"
55 PRINT @ 256, "DESPESAS"
60 PRINT @ 352, "TOTAL (+) OU (-
)"
65 PRINT @ 340, STRING$(10,"-")
70 PRINT @ 180, USING A$; R
75 PRINT @ 276, USING B$; D
80 PRINT @ 371, USING C$; N
90 GOTO 90
```

Você poderia alterar este programa para controlar a sua conta de luz e armazenar os dados por ano.

Exercício de Programação nº 13.3

```
5 CLS
10 PRINT "ISTO" TAB(POS(0)+4) "E
";
20 PRINT TAB(POS(0)+4) "ESPACAME
NTO" TAB(POS(0)+4) "UNIFORME"
```

APÊNDICE C

PROGRAMAS SIMPLES

Programa simples nº 1

```
1 CLS
2 PRINT @  32,"xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxx"
3 PRINT @ 160,"       GAROTA DE
 IPANEMA"
4 PRINT @ 192,"            (BOSSA-
NOVA)"
5 PRINT @ 256," TOM JOBIM E VINI
CIUS DE MORAIS"
6 PRINT @ 416,"xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxx"
10 FOR I=1 TO 1000
11 NEXT I:CLS
12 PRINT @ 64,"OLHA QUE COISA MA
IS LINDA"
13 PRINT @ 128,"MAIS CHEIA DE GR
ACA"
14 PRINT @ 192,"E ELA MENINA QUE
 VEM E QUE PASSA"
15 PRINT @ 256,"NUM DOCE BALANC'
A CAMINHO DO MAR"
100 A$="V31;T3;O3;L4.;G;L8;E;L4;
E;L8;D;L4.;G;L8;E;L4;E;L8;E;D;L4
.;D;L4;E;E;L8;C;L4;C"
110 B$="L8;G;E;L4;E;L3;E;D;L4;F;
D;D;L3;D;C;L4;E;C;C;L8;C;O2;L4;D
-"
120 C$="P4;O3;L1;G;P2"
121 PLAY A$:PLAY B$:PLAY C$
122 CLS
123 PRINT @ 64,"MOCA DO CORPO DO
URADO"
124 PRINT @ 128,"DO SOL DE IPANE
MA"
125 PRINT @ 192,"O SEU BALANCADO
 E MAIS QUE UM  POEMA"
126 PRINT @ 293,"E A COISA MAIS
LINDA QUE EU JA  VI PASSAR"
130 D$="L4.;G;L8;E;L4;E;L8;D;L4.
;G;L8;E;L4;E;L8;D;E;L4;DCEE;L8;D;L
4.;G;L8;E;L4;E;L8;ED;L4;F"
140 E$="L4;DD;L8;DC;L4;ECC;L8;C;
O2;L4;B-;P4;C3;L1;C;P2"
141 PLAY D$:PLAY E$
150 F$="O3;L1;F;L12;FG-FE-FE-;L4
.;D-E-;L2.;E ;FG;L8;C#;L1;G#;L12
;G#AC#F#G#F#;L4.;E;L2.;F#F#;P8"
160 G$="L8;A;L1;A;L12;AB-AGAG;L4
.;F8;L2;G;P12;L12;A2-;O4;C;O3;CD
EFG;L2.;G#"
170 H$="L4;A;L12;D-;O2;B-;O3;CDE
F;L2.;F#;P4"
171 CLS
172 PRINT @ 64,"AH! COMO TUDO E
TAO TRISTE!"
173 PRINT @ 128,"AH! POR QUE EST
OU TAO SOZINHO?"
174 PRINT @ 192,"AH! A BELEZA QU
E EXISTE!"
175 PRINT @ 256,"A BELEZA QUE NA
O E SO MINHA"
176 PRINT @ 320,"E TAMBEM PASSA
SOZINHA"
177 PLAY F$:PLAY G$:PLAY H$
190 I$="L4.;G;L8;E;L4;E;L8;D;L4.
;G;L8;E;L4;E;L8;ED;L4.;G;L4;EE;L
8;D;L4;G;L8;DE;L4;E;L8;ED;L2;A"
195 J$="L8;FFFD;O4;L2;C;O3;L8;E;
L12;EED;L1;C;P1"
200 L$="T2;V20;P4;P8;L8;E;L12;EE
D;L1;E;V10;T1;P4;P8;L8;E;L12;EED
;L1.;E;P2"
201 CLS
202 PRINT @ 64,"AH! SE ELA SOUBE
SSE"
203 PRINT @ 128,"QUE QUANDO ELA
PASSA"
204 PRINT @ 192,"O MUNDO INTEIRI
NHO SE ENCHE DE  GRACA"
205 PRINT @ 288,"E FICA MAIS LIN
DO POR CAUSA DO  AMOR..."
206 PRINT @ 384,"POR CAUSA DO AM
OR...POR CAUSA DO AMOR..."
207 PLAY I$:PLAY J$:PLAY L$
1100 CLS
```

APÊNDICE C

Programa simples nº 2

```
1 CLS
2 PRINT @  32,"**********************
**********"
3 PRINT @ 160,"      NOTURNO  OP.
 9  NUM. 3"
4 PRINT @ 256,"            J. F. C
HOPIN"
7 PRINT @ 416,"**********************
**********"
10 A$="V31;T2;O2;L8;B-;O3;L2;G;L
8;FG;L4.;F;L4;E-;O2;L8;B-;O3;L4;
G;L32;CD-C;O2;B-;O3;C;O4;L4;C;O3
;L8;G;L4.;B-;L4;A-;L8;G"
20 B$="L4.;F;L4;G;L8;D;L4.;E-C;O
2;L8;B-;O4;DC;O3;L16;B-A-GA-CD;L
4.;E-;P4;O2;L8;B-;O3;L4.;G;L16;F
G;L32;FGF;L16;EFG;L8;F;L4;E-;L16
;E-F;L32;E-FE-;L16;DE-F"
30 C$="G;O2;B;O3;CD-CFEA-G;O4;D-
C;O3;G;L4.;B-;L4;A-;L8;G;L16;E;L
64;FGFGFGFG;L16;EF;L8;GGD;L4.;E-
C"
40 D$="O2;L8;B-;O4;DC;O3;L16;B-A
-GA-;L32;A-;L16;CD;L4.;E-;L8;E-D
E-;L4.;F;L4;G;L8;F;L4.;FC;L8;E-E
-E-E-;L16;DE-;L16.;F;L32;E-;L4.;
E-;O2;B-"
50 E$="O3;L4.;B-;L4;A;L8;G;L4.;F
DE-;L8;DCD;O2;B-BB;O3;CCD"
60 F$="O2;L16;GB-;O3;E-;L4;G;O2;
L16;AD-BB-O3;C#D;L16.;G;L32;F;L4
;F;L8.;E-;L16;F;L32;E-FE-;L16;DE
-FG;O2;B;O3;CD-CFEA-G;O4;D-C;O3;
G;L4.;B-;L4;A-;L8;G"
70 G$="L16;E;L64;FGFGFGFG;L16;EF
;L8;GGD;L4.;E-C;O2;L8;B-;O4;DC#;
L32;C;O3;BB-AA-FD;O2;BB-;O3;DGFE
-;L4.;E-;L8;E-DE-"
80 H$="L4.;F;L4;G;L8;F;L4.;FC;L8
;E-E-E-E-E-;L16;DE-;L16.;F;L32;E
-;L4.;E-;O2;B-"
90 I$="O3;B-;L4;A;L8;G;L4.;FDE-;
L8;DCD;O2;B-BB;O3;CCD"
100 J$="O2;L16;GB-;O3;E-;L4;G;O2
;L16;AB-BB-;O3;C#D;L16.;G;L32;F;
L4;F;L8.;E-;L16;FE-DE-FG;O2;B;O3
;CD-CFEA-G;O4;D-C;O3;GG;L4.;B-;L
4;A-;L8;G"
110 K$="L16;E;L64;FGFGFGFG;L16;E
F;L8;GGD;L4.;E-C;O2;L16;B-;O4;L8
;D;L16;C#C;O3;L32;B;L16;B-A;L32;
A-;O2;AB-B;O3;CC#DGFE-;L2.;E-"
120 L$="L4.;E-;L8;FE-F;L2.;G;L4.
;E-;L16;E-EE-FE-FG;L32;CD-C;O2;B
-;O3;C;O4;L8;E-DC"
130 M$="O3;L4;B-;L8;AA-CDE-;L16;
F;L32;E-FE-;L16;E-E-;O4;L8;G;L16
;FE-DC;O3;L8;BB-A;L16;AA-A-G;L16
.;G;L32;F;L2.;E-;L4.;E-;L16;E-FE
-FE-FE-F"
140 N$="L4.;G;P4;P8;L8;E-;O2;L16
;A-B-A-GA-B;O3;E-A-;O4;E-;P32;L3
2;F;O4;L8;GE-;O5;L4;E-;L8;DCB;O4
;B-AA-GDE-;O5;L4.;E-;O4;L8;F;O5;
C"
150 O$="O4;L2.;EB-"
160 P$="O4;L16;BB-;O5;C;O4;A"
170 Q$="O4;L16;BB-;O5;DC;O4;B-AA
-GFDE-C;O3;L8;B-A-CDE-;O2;B-;O3;
GE-;O2;B-;O3;GE-;O2;B-;O3;G;L4.;
E-;O4;E-;O2;L2.;E-"
1000 PLAY A$:PLAY B$:PLAY C$:PLA
Y D$:PLAY E$:PLAY F$:PLAY G$:PLA
Y H$:PLAY I$:PLAY J$:PLAY K$:PLA
Y L$:PLAY M$:PLAY N$:PLAY O$:PLA
Y P$:PLAY P$:PLAY P$:PLAY P$:PLA
Y P$:PLAY P$:PLAY P$:PLAY P$:PLA
Y P$:PLAY P$:PLAY P$:PLAY P$:PLA
Y Q$
1010 CLS
```

APÊNDICE C

Programa simples nº 3

```
1 CLS
2 PRINT @  32,"*****************
***************"
3 PRINT @ 160,"     APANHEI-TE, C
AVAQUINHO!"
4 PRINT @ 192,"                (CHOR
O)"
5 PRINT @ 256,"  ERNESTO NAZARET
H - BALDONAM"
8 PRINT @ 416,"*****************
***************"
10 A$="T3;L16;04;GF#EEDC#DF#EDCC
;03;BA#B;04;EDC;03;BBAG#A;04;DC;
03;BAAGF#G;04;C;03;BAGF#ED#EAGF#
E"
20 B$="EDC#DEF#GAB;04;CC#DF#EDC;
03;A#B;04;ED;P16;GF#EEDC#DF#EDCC
;03;BA#B;04;EDC;03;B"
30 C$="BAG#A;04;DC;03;BAAGF#G;04
;C;03;BAGF#ED#EAGF#EEDC#DEF#GAB;
04;CC#DD#EE#F#;L4;G;P16"
40 D$="L16;03;B;L32;04;D;L16;C;0
3;BA#;B;04;EF#G;03;B;04;G;03;B;0
4;G;03;A#;04;C#F#F#;03;A#;04;F#;
03;A#;04;G;03;AB;04;F#F#;03;A;04
;F#;03;A"
50 E$="04;F#;03;GB;04;E;P16;03;B
;L32;04;D;L16;C;03;BA#B;04;EF#G;
03;B;04;G;03;B;04;G;03;B;04;F#;0
3;B;04;E;03;B;04;D;03;B;04;D;03;
BF#;04;C#E;03;E;04;C#;03;E;04;C;
03;D#F#B;P16;B;L32;04;D;L16;C;03
;B"
60 F$="A#B;04;EF#G;03;B;04;G;03;
B;04;G;03;A#;04;C#F#F#;03;A#;04;
F#;03;A#;04;G;03;AB;04;F#F#;03;A
;04;F#;03;A;04;F;03;G#B;04;E;P16
;L8;D#;L16;EF;03;A;04;E;03;A;04;
D;03;A;04;C;03;A"
70 G$="04;C;03;EGB;P16;L8;A#;L16
;B;04;C;03;BA#B;04;ED#GF#;L4;E;P
16"
80 H$="04;L16;GF#DAGEC;03;AGEC;L
8.;A;L16;A;P8;L8;A;P16;G;L16;AB;
04;L8;C;L16;D;L8.;E;L16;E;P8;L8;
E"
90 I$="P16;L16;ED#EFED#ECE;03;B;
04;E;03;A;04;E;03;G;04;E;P16;DC#
DEDC#DED;03;B;04;G;P16;GF#G"
100 J$="AGEC;03;AGEC;L8.;A;L16;P
8;L8;A;P16;G;L16;AB;L8;04;C;L16;
D;L8.;E;L16;E;P8;L8;E;P16;D#;L16
;EFEDE;L8.;A;L16;A;P8;L8;A;P16;L
16;GFD;03;BGFD;L8.;C;04;L16;C;P1
6"
900 X$="XA$;XB$;XC$;"
901 Y$="XH$;XI$;"
910 PLAY "V31":PLAY X$:PLAY X$:P
LAY D$:PLAY E$:PLAY F$:PLAY G$:P
LAY D$:PLAY E$:PLAY F$:PLAY G$:P
LAY X$:PLAY D$:PLAY E$:PLAY F$:P
LAY G$:PLAY Y$:PLAY J$:PLAY Y$:P
LAY J$:PLAY X$:PLAY D$:PLAY E$:P
LAY F$:PLAY G$
990 CLS
```

APÊNDICE C

Programa Simples nº 4

```
1 CLS
2 PRINT @  32,"**********************
**************"
3 PRINT @ 160,"        TICO-TICO
NO FUBA"
4 PRINT @ 192,"              (CHOR
O)"
5 PRINT @ 256,"        ZEQUINHA D
E ABREU"
8 PRINT @ 416,"**********************
**************"
110 DATA O3;T4;L8;5;4;5;6;5;L4.;
10;L8;5;4;5;6;5;L4.;9;L8;5;4;5;6
;5;04;L8;3;03;L8;12;9
120 DATA 5;3;2;L3;1;P8;L8;10;9;8
;6;10;L4.;04;3;L8;1;03;L8;12;10;
5;10;L4.;04;1
130 DATA L8;1;03;L8;12;11;12;02;
L8;12;03;L8;4;7;10;04;L8;1;03;L8
;12;10;9;L4;04;5
140 DATA 03;L8;12;04;L8;5;03;L8;
5;4;5;6;5;L4.;10;L8;5;4;5;6;5;L4
.;9;L8;5;4;5;6;5
150 DATA 04;L8;3;03;L8;12;9;5;3;
2;L3;1;P8;L8;10;7;8;6;10;04;L4.;
3;L8;1;03;12;10
160 DATA 5;10;04;L4.;1;L8;1;02;1
2;11;12;9;7;12;8;L8;5;3;1;03;12
;L3;10
170 DATA 04;P8;L8;10;05;2;5;10;0
4;10;05;2;L4;7;04;L8;10;05;2;7;7
;04;10;05;2;L4;5
180 DATA 04;L8;10;05;2;7;7;04;10
;05;2;L4;5;04;L8;10;05;2;7;7;04;
12;05;3;L4;5
190 DATA L8;3;5;7;10;3;5;L4;9;L8
;3;5;7;7;04;L8;12;05;3;L4;5;L8;3
;5;7;10;3
200 DATA 5;L4;9;L8;3;5;7;7;04;10
;05;2;L4;5;04;L8;10;05;2;5;10;04
;10;05;2
210 DATA L4;9;04;L8;10;05;2;7;7;
04;10;05;2;L4;5;04;L8;10;05;2;5;
7;5;2;04;11
220 DATA 05;7;5;2;04;11;12;11;12
;05;2;L4;3;P4;03;L8;3;2;3;5;7;9;
10;12;04;2;3;4;5
230 DATA 7;5;3;2;03;12;10;9;7;5;
3;2;02;12;L3;10;P8;03;L8;12;10;9
;8;04;1;5
240 DATA 8;1;5;8;9;L4;10;6;P8;03
;L8;12;10;9;8;12;04;3;6;03;12
250 DATA 04;3;6;8;L4;10;5;P8;05;
L8;1;1;1;L4;1;L8;04;12;12;L4;12
260 DATA L8;10;10;L4;10;6;P8;L8;
12;12;12;L4;12;L3;10;10;L4;10;L8
270 DATA 8;8;L4;8;3;P8;L8;5;1;03
;10;8;04;1;5;3;1;5;8;9;L4;10
280 DATA 6;P8;03;L8;12;10;9;8;12
;04;3;6;03;12;04;3;6;8;L4;10;5
290 DATA P8;L8;1;03;12;11;10;9;1
0;12;04;3;1;03;12;04;1;5;03;8;04
;1;5
300 DATA 8;7;6;5;3;1;03;12;10;8;
6;5;3;L3;1;P8
310 FORI=110TO160STEP10:READA$:P
LAYA$:NEXT
320 RESTORE:FORI=110TO160STEP10:
READA$:PLAYA$:NEXT
330 RESTORE
340 CLS
```

APÊNDICE C

Programa simples nº 5

```
1 ' *** DESENHANDO TRIANGULOS ***
10 CLS:CLEAR
75 PRINT @ 96,STRING$(32,"*")
80 PRINT @ 288,STRING$(32,"*")
100 PRINT @ 164,"ESSE PROGRAMA D
ESENHA UM "
102 PRINT @ 226,"TRIANGULO E CAL
CULA SUA AREA"
110 FOR X=1 TO 2200:NEXT:CLS
120 CLS:PRINT "PARA 3 LADOS DIGI
TE, SSS(0-100)"
125 PRINT "PARA 2 LADOS (0-100)
E 1 ANGULO (0-90) DIGITE, SAS"
130 PRINT "PARA 1 LADO (0-60) E
2 ANGULOS (0-90) DIGITE, ASA"
140 INPUT A$:IF A$="SAS" GOTO 30
0
150 IF A$="ASA" GOTO 400
200 ' SSS
210 PRINT "FORNECA 3 LADOS:PRIME
IRO O MAIOR"
220 INPUT L1,L2,L3
225 IF L2>L1 OR L3>L1 THEN PRINT
 " POR FAVOR, PRIMEIRO O MAIOR"
:PRINT:GOTO 210
230 S=(L1+L2+L3)/2
235 IF S<=L1 THEN PRINT "*** NAO
 E TRIANGULO***":PRINT :GOTO 210
240 Y3=2*SQR(S*(S-L2)*(S-L1)*(S-
L3))/L1
250 A=Y3/L2:A=ATN(A/SQR(-A*A+1))
260 X3=COS(A)*L2
270 AR=(L1*Y3)/2
280 GOTO 490
300 ' SAS
310 PRINT "FORNECA 2 LADOS E 1 A
NGULO:":PRINT " AB,AC,T (PRIMEIR
O O MAIOR LADO)"
320 INPUT L1,L2,T
325 T=(T*3.14159)/180
330 Y3=L2*SIN (T)
340 X3=COS(T)*L2
350 AR=(L1*Y3)/2:GOTO 490
400 ' ASA
410 PRINT "FORNECA 2 ANGULOS E 1
 LADO:":PRINT " T1,T2,AB"
420 INPUT T1,T2,L2
425 T1=(T1*3.14159)/180:T2=(T2*3
.14159)/180
430 Y3=L2*SIN(T1)
440 B1=COS(T1)*L2
450 B2=Y3/TAN(T2)
460 L1=B1+B2:X3=B1:IF L2>L1 THEN
 X=L1:L1=L2:L2=X
470 AR=(L2*Y3)/2
490 CLS:PMODE 4,1:PCLS:SCREEN 1,
1
500 F=1
510 VC=(3.14159*(L1*F-X3*F)*(Y3*
F)^2)/3
520 VS=(3.14159*(X3*F)*(Y3*F)^2)
/3:VT=VC+VS
530 S1=Y3/X3:S2=Y3/(X3-L1)
532 IF INT(X3)=0 THEN 1100
533 IF INT(X3)=INT(L1) THEN 1000
535 IF X3>L1 THEN 1199
537 IF X3=L2 THEN 1000
540 FOR Y=20 TO L1*2+20 STEP 2:P
SET (Y,Y3+5,5):NEXT
550 FOR X=0 TO X3
551 PSET (X*2+20,S1*(X3-X)+5,5):
NEXT
560 FOR X=X3 TO L1:PSET (X*2+20,
Y3+(S2*(L1-X)+5),5):NEXT
580 FOR X=1 TO 4000:NEXT X
610 PRINT @ 130,"AREA=";AR;" UNI
DADES QUADRADAS";
630 PRINT @ 352,"PARA EXECUTAR N
OVAMENTE,":PRINT "PRESSIONE <1>
<ENTER>":INPUT BG:IF BG=1 THEN 1
20
640 STOP:GOTO 10
1000 FOR Y=5 TO Y3+5:PSET (X3*2+
20,Y,4):NEXT:GOTO 540
1100 FOR Y=5 TO Y3+5:PSET (20,Y,
5):NEXT :GOTO 540
1200 FOR X=L1 TO X3:PSET (X*2+20
,Y3+(S2*(L1-X)+5),5):NEXT:GOTO 5
40
1300 FOR X=X3 TO 0:PSET (X*2+20,
Y3+(S1*(0-X)+5),4):NEXT:GOTO 540
```

APÊNDICE C

Programa simples nº 6

```
1 ' *** DENTRO-FORA ***
2 '
5 PMODE 3,1
10 PCLS3
15 SCREEN 1,0
20 FOR I=3 TO 7
25 FOR J=2 TO 6
30 FOR S=0 TO 3
35 FOR R=0 TO 3
40 COLOR R,S
45 A=0:B=255:C=0:D=191
50 LINE (A,C)-(B,D),PSET,B
55 A=A+J:B=B-J:C=C+I:D=D-I
60 IF A<255 AND C<191 THEN 50
65 NEXT R
70 NEXT S
75 NEXT J,I
80 GOTO 30
```

Programa simples nº 7

```
1 '*** ESTUDOS DE PROJECAO ***
2 '
5 PMODE 4,1
10 PCLS
15 SCREEN 1,0
20 DRAW "BM50,50R60D10NL20D20L20
NU20L20NU20L20U20NR20U10" ' VIST
A DE TOPO
25 DRAW "BM50,100R20ND20R20ND20R
20D20NL20D10L60U10NR20U20" 'VIST
A FRONTAL
30 DRAW "BM150,100R30D30L30U10NE
20U20" 'VISTA LATERAL
35 ' VISTA OBLIQUA--LINHAS 40-60
40 DRAW "BM150,50U5E15R10BF20BD3
0NR5L20H25U10
45 DRAW "BM150,50U5F8U15R15H8F8L
15F8NR15D15F8ND10E15NR10H8
50 LINE (175,30)-(200,55),PSET
55 LINE -(200,80),PSET
60 LINE (167,60)-(183,46),PSET
65 GOTO 65
```

Programa simples nº 8

```
1 '*** ABRINDO CAIXAS ***
2 '
5 PCLEAR 8
10 PMODE 3,1
15 PCLS
20 COLOR 6,5
25 DRAW "BM100,100U30NR30E15R30N
G15D30G15NU30L30"
30 PAINT (105,95),8,6
35 PAINT (135,80),8,6
40 PAINT (110,65),8,6
45 SCREEN1,1
50 FOR X=1 TO 600:NEXT X
110 PMODE 3,5
112 PCLS
115 COLOR 6,5
120 DRAW "BM100,100U30NR30E20R30
G20D30NL30F20L30H20
125 LINE (100,100)-(70,95),PSET
130 LINE -(70,65),PSET
135 LINE -(100,70),PSET
140 LINE (70,95)-(40,65),PSET,B
145 LINE (130,100)-(160,95),PSET
150 LINE -(160,65),PSET
155 LINE -(130,70),PSET
160 PAINT (95,95),8,6
165 PAINT (105,95),8,6
170 PAINT (135,85),8,6
175 PAINT (45,85),8,6
180 PAINT (115,65),8,6
185 PAINT (125,114),8,6
190 SCREEN 1,1
195 FOR X=1 TO 600:NEXT X
200 GOTO 10
```

APÊNDICE C

Programa simples nº 9

```
1 ' *** CURVA DO SENO ***
2 '
5 PMODE 4,1
10 PCLS
15 SCREEN 1,1
20 LINE (0,86)-(255,86),PSET
25 PI=3.14159
30 A1=-4*PI
35 A2=4*PI
40 N=180
45 R=50
50 X=(A2-A1)/N
55 F=255/(A2-A1)
60 FOR I=A1 TO A2 STEP X
65 X=I*F
70 Y=R*SIN(I)
75 PSET ((X+140),(80+Y),1)
80 NEXT I
90 GOTO 90
```

Programa simples nº 10

```
1 ' *** SENO/COSENO ***
2 '
10 PMODE 4,1
20 PCLS
30 SCREEN 1,0
40 LINE (127,5)-(127,185),PSET
50 LINE (7,95)-(247,95),PSET
60 FOR XSCALE=7 TO 247 STEP 20
70 PRESET(XSCALE,95)
80 NEXT XSCALE
90 FOR YSCALE=5 TO 185 STEP 10
100 PRESET (127,YSCALE)
110 NEXT YSCALE
130 FOR X=-180 TO 180 STEP 1.5
140 AX=X/57.29578
145 XP=X/1.5+127
150 F1=-(SIN(AX)*90)+95
160 F2=-(COS(AX)*90)+95
170 PSET (XP,F1,1):PSET (XP,F2,1
)
180 NEXT X
190 GOTO 190
```

Programa simples nº 11

```
1 ' *** GRAFICOS RANDOMICOS ***
2 '
10 PMODE 3,1
15 PCLS
20 SCREEN 1,1
25 F=RND(4):B=RND(8):IF B=F OR (
B-4=F) THEN 25
30 COLOR F,B:PCLS B:FOR L=0 TO 5
35 LINE -(RND(255),RND(191)),PSE
T
40 CIRCLE (RND(255),RND(191)),RN
D(100)
50 NEXT :FOR P=0 TO 10
55 PAINT (RND(255),RND(191)),RND
(4),F
60 NEXT:FOR H=1 TO 7
65 FOR T=0 TO 600:NEXT T:GOTO 10
```

APÊNDICE C

Programa simples nº 12

```
1 ' *** COBERTOR NAVAHO ***
2 '
5 PMODE 3,1
10 PCLS 4
15 SCREEN 1,0
20 COLOR 1,0
25 FOR X=0 TO 255 STEP 10
30 OY=Y
35 Y=30-OY
40 LINE (X,100-Y)-(X+10,100-OY),
PSET
45 LINE (X,120+Y)-(X+10,120+OY),
PSET
50 NEXT
60 FOR C=2 TO 8
65 PAINT (0,110),C,1
70 NEXT
80 GOTO 5
```

Programa simples Nº 13

```
1 ' *** LACO PINTADO ***
2 '
5 PMODE 3,1
10 PCLS
20 SCREEN 1,1
30 DRAW "BM50,180U60BU20U60R60BR
20R60D60BD20D60L60BL20L60
40 DRAW "BM50,180U60R40BR20R80D2
0BL20L60BL20L20D20R20BR60R20U20
50 DRAW "BM50,180R60U80BU20U40L4
0BD20D20BD60D20R20U60BU20U20L20
60 DRAW "BM50,180U60BU40BR20R60B
R20R20U20L20D60BD20D20R20
70 DRAW "BM50,180BR80U40BU20U80
80 DRAW "BM50,180BU80R80BR20R40
90 PAINT (85,128),6,8
95 PAINT (95,70),6,8
97 PAINT (155,95),6,8
98 PAINT (135,145),6,8
99 PAINT (128,185),7,8
100 PAINT (75,150),7,8
101 PAINT (160,150),7,8
102 PAINT (75,75),7,8
103 PAINT (160,75),7,8
104 PAINT (120,110),7,8
110 FOR X=1 TO 600:NEXT X
200 GOTO 5
```

Programa simples nº 14

```
1 ' *** DESENHANDO ***
2 '
3 CLS
5 PRINT @ 128,STRING$(32,"*"):PR
INT @ 288,STRING$(32,"*")
10 PRINT @ 234,"DESENHANDO"
15 FOR X=1 TO 600:NEXT X
20 CLS
25 PRINT @ 64,"PRESSIONE (^) PAR
A SUBIR,(SETA P/ BAIXO) PARA DES
CER,(<-) PARA ESQUERDA,(->) PARA
 DIREITA,(A) PARA SUDOESTE,(S) P
ARA SUDESTE,(W) PARA NORDESTE,(Q
) PARA NOROESTE"
30 PRINT @ 256,"PRESSIONE <1> PA
RA RETAS INVISIVEIS, <2>,<3> OU
<4> PARA RETAS VISIVEIS DE DIFER
ENTES CORES E </> PARA MUDAR A C
OR DO FUNDO"
35 PRINT @ 416,"PRESSIONE <BARRA
 DE ESPACO> PARA PAUSA"
40 FOR X=1 TO 4800:NEXT X
45 CC=4:TG=0
50 PMODE 3,1
55 PCLS
60 SCREEN 1,TG
70 X=128:Y=96:XI=0:YI=0
80 U$="^":D$=CHR$(10):W$=CHR$(8)
:E$=CHR$(9)
90 NW$="Q":NE$="W":SW$="A":SE$="
S"
```

APÊNDICE C

```
100 C1$="1":C2$="2":C3$="3":C4$=
"4"
110 A$=INKEY$
120 IF A$=U$ THEN YI=-1:XI=0:GOT
O 240
130 IF A$=D$ THEN YI=1:XI=0:GOTO
 240
140 IF A$=W$ THEN XI=-1:YI=0:GOT
O 240
150 IF A$=E$ THEN XI=1:YI=0:GOTO
 240
160 IF A$=NE$ THEN XI=1:YI=-1:GO
TO 240
170 IF A$=NW$ THEN XI=-1:YI=-1:G
OTO 240
180 IF A$=SE$ THEN XI=1:YI=1:GOT
O 240
190 IF A$=SW$ THEN XI=-1:YI=1:GO
TO 240
200 IF C1$<=A$ AND A$<=C4$ THEN
CC=ASC(A$)-48:GOTO 240
210 IF A$="/" THEN TG=(NOT TG AN
D 1) OR (TG AND NOT 1):GOTO 240
220 SCREEN 1,TG
230 IF A$=" " THEN XI=0:YI=0
240 X=X+XI:Y=Y+YI:IF X<0 THEN X=
0
250 IF X>255 THEN X=255
260 IF Y<0 THEN Y=0
270 IF Y>191 THEN Y=191
275 IF CC=1 THEN PSET (X,Y,3)
280 PSET (X,Y,CC)
290 GOTO 110
```

Programa simples nº 15

```
1 ' *** RETAS ***
2 '
5 CLS
20 C=C+1
25 IF C>8 THEN C=5
30 COLOR C,1
50 PRINT "DIGITE X0,Y0";
60 INPUT X0,Y0
70 PRINT "DIGITE X1,Y1";
80 INPUT X1,Y1
90 PMODE 3,1
95 PCLS
100 SCREEN1,1
110 LINE (X0,Y0)-(X1,Y1),PSET
115 FOR X=1 TO 2000:NEXT X
120 GOTO 20
```

Programa simples nº 16

```
1 ' *** RETAS RANDOMICAS ***
2 '
20 PMODE 4,1
25 PCLS
30 SCREEN 1,1
35 X=RND(255):Y=RND(191)
40 LINE -(X,Y),PSET
45 FOR X=1 TO 200:NEXT X
50 GOTO 35
```

APÊNDICE C

Programa simples nº 17

```
1 ' *** ROSACEA ***
2 '
5 PCLEAR 8
10 PMODE 4,1
15 PCLS
20 SCREEN 1,0
25 PI=3.14159
30 A1=0:A2=2*PI
35 N=360:A=50
40 X=(A2-A1)/N
45 FOR I=A1 TO A2 STEP X
50 R=A*COS (4*I)
55 X=R*SIN(I)
60 Y=R*COS(I)
65 PSET (128+X,96+Y,5)
70 NEXT I
75 GOTO 25
```

Programa simples nº 18

```
1 ' *** BOMBA RELOGIO ***
2 '
10 PMODE 4,1
15 PCLS
20 SCREEN 1,1
25 CIRCLE (128,96),80
30 CIRCLE (128,96),90
35 PAINT (0,0),5
40 FOR T=30 TO -30 STEP -1
45 A=(2*3.1415)*T/60
50 LINE (128,96)-(75*SIN(A)+128,
75*COS(A)+96),PSET
55 SOUND 9*2+1,20/(9+1)+1
60 LINE (128,96)-(75*SIN(A)+128,
75*COS(A)+96),PRESET
65 9=60-2*T:FOR Y=0 TO 0 STEP -1
:NEXT
70 NEXT
75 CLS
80 PCLS
85 PRINT @ 237,"BOOM!"
90 SOUND 1,30
95 PMODE 4,1
100 SCREEN 1,1
105 FOR I=2 TO 200 STEP 2
110 CIRCLE (128,96),I
115 NEXT I
120 SCREEN 1,1
125 FOR X=2 TO 200 STEP 2
130 CIRCLE (128,96),X,,3
135 NEXT X
140 FOR I=2 TO 200 STEP 2
145 CIRCLE (128,96),I,3,5
150 NEXT I
155 GOTO 155
```

APÊNDICE C

Programa simples nº 19

```
1 ' *** CATAVENTO ***
2 '
5 PCLEAR 8
50 GOTO 600
60 LINE ((255-X),(191-Y))-(X,Y),
PSET
61 J=J+1:IF J>A THEN J=0:A=RND(5
0)
63 RETURN
600 REM CATAVENTO GIRANDO
601 FOR I=1 TO 5 STEP 4
602 PMODE 3,I
603 PCLS
604 SCREEN 1,0
605 A=25:X=0:Y=0:J=0
610 FOR X=0 TO 254
612 COLOR X/32+1,5
615 GOSUB 60:NEXT X
620 FOR Y=0 TO 190
623 COLOR Y/24+1,5
625 GOSUB 60:NEXT Y
630 FOR X=255 TO 1 STEP -1
633 COLOR X/32+1,5
635 GOSUB 60:NEXT X
640 FOR Y=191 TO 1 STEP -1
643 COLOR Y/24+1,5
645 GOSUB 60:NEXT Y
650 NEXT I
660 FOR I=1 TO 5 STEP 4
670 PMODE 3,I
680 SCREEN 1,0
690 FOR T=1 TO 30:NEXT T
700 NEXT I
710 GOTO 660
```

Programa simples nº 20

```
1 ' *** TRIANGULOS ***
2 '
10 FOR A=90 TO 0 STEP -4
15 S1=A*9:S2=191
20 A3=A/57.2958
30 X1=0:Y1=191
40 X2=S1+X1:Y2=Y1
50 X3=X1+S2*COS(A3):Y3=Y1-S2*SIN
(A3)
55 GOSUB 1000
90 NEXT A
99 GOTO 99
1000 PMODE 4,1
1005 PCLS
1010 SCREEN 1,0
1020 LINE (X1,Y1) (X2,Y2),PSET
1030 LINE -(X3,Y3),PSET
1040 LINE -(X1,Y1),PSET
1060 RETURN
```

Programa simples nº 21

```
1 ' *** CONTANDO ***
2 '
10 CLS
20 CLEAR 1000
30 PRINT "ONDE QUER COMECAR A CO
NTAR?"
35 INPUT A$
40 P=LEN(A$)
50 PRINT :PRINT A$
60 C=VAL(MID$(A$,P,1))+1
70 MS$=A$:MR$=RIGHT$(STR$(C),1):
PS=P:GOSUB 200:A$=MS$
80 IF C<10 THEN 40
90 P=P-1
100 IF P=0 THEN IF LEN(A$)=255 T
HEN PRINT "OVERFLOW":END:ELSE A$
="1"+A$:GOTO 40
110 GOTO 60
200 LS=LEN(MS$)
210 IF LS<>LEN(MR$)+LS-1 OR PS<1
 THEN STOP
220 MS$=LEFT$(MS$,PS-1)+MR$+RIGH
T$(MS$,LS-PS)
230 RETURN
```

APÊNDICE D

MAPA DE MEMÓRIA COM COLOR BASIC EXTENDIDO

ENDEREÇO DECIMAL	CONTEÚDO	ENDEREÇO HEXADECIMAL
0 - 1023	Usado pelo sistema	0 - 3FF
255	Página direta de RAM	0FF
1023	Página extendida de RAM	3FF
1024 - 1535	Memória de vídeo-texto	400 - 5FF
	Memória de vídeo-gráfica	
1536 - 3071	Página 1	600 - BFF
3072 - 4607	Página 2	C00 - 11FF
4608 - 6143	Página 3	1200 - 17FF
6144 - 7679	Página 4	1800 - 10FF
7680 - 9215	Página 5	1E00 - 23FF
9216 - 2559	Página 6	2400 - 29FF
2560 - 12287	Página 7	2A00 - 2FFF
12288 - 13823	Página 8	3000 - 35FF
13824 - 32767	Armazenamento de programas e variáveis	3600 - 7FFF
32768 - 40959	COLOR BASIC EXTENDIDO	8000 - 9FFF
40960 - 49151	COLOR BASIC	A000 - BFFF
49152 - 65279	Reservada para expansão	C000 - FEFF
65280 - 65535	Entrada e saída	FF00 - FFFF

APÊNDICE E

RELAÇÃO DE CÓDIGOS ASCII

A seguir, apresentamos a relação de códigos ASCII dos caracteres do teclado. A primeira coluna indica o caráter ; a segunda, representa o código em notação decimal (base 10); e a terceira é o código em notação hexadecimal (base 16).

CARATÉR	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓD:GO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
ESPAÇO	32	20	4	52	34	H	72	48
!	33	21	5	53	35	I	73	49
"	34	22	6	54	36	J	74	4A
#	35	23	7	55	37	K	75	4B
$	36	24	8	56	38	L	76	4C
%	37	25	9	57	39	M	77	4D
&	38	26	:	58	3A	N	78	4E
'	39	27	;	59	3B	O	79	4F
(	40	28	<	60	3C	P	80	50
)	41	29	=	61	3D	Q	81	51
*	42	2A	>	62	3E	R	82	52
+	43	2B	?	63	3F	S	83	53
,	44	2C	@	64	40	T	84	54
-	45	2D	A	65	41	U	85	55
.	46	2E	B	66	42	V	86	56
/	47	2F	C	67	43	W	87	57
0	48	30	D	68	44	X	88	58
1	49	31	E	69	45	Y	89	59
2	50	32	F	70	46	Z	90	5A
3	51	33	G	71	47			

APÊNDICE E

RELAÇÃO DE CÓDIGOS DAS TECLAS ESPECIAIS

CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
↑	94	5E	SHIFT ↑	95	5F
↓	10	0A	SHIFT ↓	91	5B
←	8	08	SHIFT ←	21	15
→	9	09	SHIFT →	93	5D
BREAK	03	03	SHIFT CLEAR	92	5C
CLEAR	12	0C			
ENTER	13	0D			

APÊNDICE E

RELAÇÃO DE CÓDIGOS DAS LETRAS MINÚSCULAS

A seguir apresentamos a relação de códigos ASCII das letras minúsculas. Para usar as letras minúsculas é preciso primeiro pressionar as teclas (SHIFT) (0) simultaneamente. As letras minúsculas aparecerão no vídeo em cores reversas (letras verdes e fundo preto).

CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
a	97	61	n	110	6E
b	98	62	o	111	6F
c	99	63	p	112	70
d	100	64	q	113	71
e	101	65	r	114	72
f	102	66	s	115	73
g	103	67	t	116	74
h	104	68	u	117	75
i	105	69	v	118	76
j	106	6A	w	119	77
k	107	6B	x	120	78
l	108	6C	y	121	79
m	109	6D	z	122	7A

APÊNDICE F

MENSAGENS DE ERRO

/0 - Divisão por zero. Foi pedido ao computador para dividir um número por zero. Esta operação é impossível. Este erro também ocorre se você escrever o nome de um arquivo sem aspas.

AO - Tentativa de abrir um arquivo de dados que já se encontra aberto (Already Open).

BS - O índice de um vetor está fora dos limites admissíveis. Use a instrução DIM para dimensionar os vetores. Por exemplo, se você escrever no seu programa A(12) sem especificar na instrução DIM que A é um vetor de 12 posições ou mais, você obterá este erro (Bad Subscript).

CN - Não é possível continuar o programa. Se você utilizar o comando CONT estando no fim (END) do programa, este erro ocorrerá (Can Not).

DD - Tentativa de redimensionar um vetor. Um vetor só pode ser dimensionado uma vez. Por exemplo, você não pode ter DIM A(12) e DIM A(50) no mesmo programa (Dimension).

DN - Erro no número do periférico. Você só pode usar três tipos de periféricos com os comandos OPEN, CLOSE, PRINT ou INPUT; 0, -1 ou -2. Se você tentar usar outro número ocorrerá este erro (Device Number).

DS - Instrução direta. Existe uma instrução direta no arquivo. Este erro ocorre se você carregar um programa cujas instruções não tenham um número de linha (Direct Statement).

APÊNDICE F

FC - Chamada incorreta de função. Este erro ocorre quando você usa uma palavra BASIC com uma parâmetro (número) fora dos limites admissíveis. Por exemplo, SOUND(260,260) e CLS(10) provocarão este erro. Existem outras situações em que ocorre este erro, como por exemplo, quando usamos vetores com índices negativos(A(-1) ou quando chamamos funções USR sem termos fornecido (POKE) o seu endereço antes (Function Call).

FD - Formato de campo incorreto. Este erro ocorre quando você lê (INPUT) ou grava (PRINT) um dado usando um formato inadequado. Se tentar, por exemplo, ler um string com a instrução INPUT#-1,A este erro ocorrerá (File Data).

FM - Tipo de arquivo incorreto. Este erro ocorre quando você tenta ler (INPUT) dados de um arquivo aberto para saída (OUTPUT), ou tenta gravar (PRINT) dados num arquivo aberto para entrada (INPUT). (File Mode)

ID - Instrução direta incorreta. Você só pode usar a instrução INPUT dentro de um programa. Não é possível usá-la como um comando(Illegal Direct statement).

IE - Tentativa de ler um dado além do fim do arquivo. Use as instruções EOF e LOF para descobrir se você atingiu o fim de um arquivo. Quando atingir o fim do arquivo, feche-o (CLOSE) (Input past End).

IO - Erro de entrada ou saída. O computador está encontrando dificuldades para ler ou gravar informações num arquivo em fita.

LS - String muito longo. Um string pode ter no máximo 255 caracteres (String too Long).

NF - Instrução NEXT sem FOR correspondente. Você usou uma instrução NEXT sem a instrução FOR correspondente. Este erro também ocorre se você inverter a ordem das instruções NEXT num ninho FOR/NEXT (NEXT without FOR).

APÊNDICE F

NO - O arquivo não foi aberto. Você não pode ler ou gravar dados num arquivo sem antes abrí-lo (OPEN) (Not Open).

OD - Não há dados nas instruções DATA para atender a todas as instruções READ. Verifique as instruções DATA e READ do seu programa (Out of Data).

OM - Estouro de memória. Você já usou toda a memória disponível do computador. Tente alterar a estrutura do programa dividindo-o em vários módulos independentes e transformando cada módulo num programa autônomo (Out of Memory).

OS - Estouro de string. Não há espaço suficiente na memória do computador para executar as operações envolvendo strings. Use a instrução CLEAR no começo do programa para reservar mais espaço para os strings (Out of String).

OV - Estouro de variável numérica. Uma das operações aritméticas, no seu programa, gerou um número muito grande ou muito pequeno e o computador não tem condições de processar este número (OVerflow).

RG - Instrução RETURN sem o GOSUB correspondente. Você usou uma instrução RETURN sem a instrução GOSUB correspondente. (Return without Gosub).

SN - Erro de sintaxe. Este erro ocorre quando você escreve incorretamente uma palavra BASIC ou usa incorretamente a pontuação, os parênteses ou as aspas. Digite novamente a linha incorreta (SiNtaxe).

ST - Tentativa de usar strings em fórmulas muito complexas. Uma operação com strings está muito complexa para ser processada pelo computador. Desdobre esta operação em outras de menor porte (STring).

APÊNDICE F

TM - Incompatibilidade de tipo de variável. Este erro ocorre quando você tenta atribuir um número a uma variável string (A$=3) ou um string a uma variável numérica (A="MICRO"). Este erro também ocorre se você não colocar o nome de um arquivo entre duas aspas (Type Mismatch).

UL - Linha não definida. Existe no programa uma instrução GOTO ou GOSUB ou qualquer outra instrução de desvio pedindo ao computador que vá para uma linha cujo número não existe (Undefined Line).

APÊNDICE G

SUMÁRIO DO COLOR BASIC

EXTENDIDO

COMANDO	DESCRIÇÃO	EXEMPLO
ABS	Retorna o valor absoluto da expressão fornecida.	PRINT ABS(-5*2)
ASC	Retorna o código ASCII do primeiro caráter do string fornecido.	PRINT ASC("B") S = ASC (T$)
ATN	Retorna o arcotangente em radianos.	Y = ATN(X/3)
AUDIO	Liga e desliga a saída do gravador cassete para o alto-falante da televisão.	AUDIO ON AUDIO OFF
CHR$	Retorna o caráter ASCII, de controle ou gráfico relativo ao código fornecido.	PRINT CHR$(67) Y$ = CHR$(92)
CIRCLE	Desenha um círculo com centro no ponto (x,y), raio y, cor c e relação altura/comprimento (hw) entre 0 e 4. O círculo pode começar e terminar em um ponto dado (0-1).	CIRCLE (128,96), 50, 4, 1, .5, .75
CLEAR	Reserva espaço de memória para armazenar string (sem o CLEAR, o computador reserva apenas 200 bytes). Se estiver carregando um programa em linguagem de máquina, você pode usar um segundo número para indi-	CLEAR CLEAR 1000 CLEAR 200, 14000

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	car o endereço mais alto do BASIC. Este comando também inicializa as variáveis numéricas com zero e as variáveis string com nulos (LEN=0).	
CLOAD	Carrega um programa em BASIC do gravador cassete. Se não for indicado o nome do arquivo, é carregado o primeiro que for encontrado na fita. O nome do arquivo pode ter no máximo 8 caracteres.	CLOAD CLOAD "PROG"
CLOADM	Carrega um programa em linguagem de máquina do gravador cassete. Você pode fornecer um número que será somado ao endereço de carregamento do programa.	CLOADM "PROG" CLOADM CLOADM "PROG", 1000
CLOSE	Fecha os arquivos abertos. Cuidado, não se esqueça que este comando esvazia o buffer de comunicação.	CLOSE#-1 CLOSE#-2 CLOSE
CLS	Limpa a tela e coloca a cor do código fornecido. Caso não seja indicado nenhum código, é usado a cor verde.	CLS 3 CLS
COLOR	Determina a cor do primeiro plano e a cor do fundo.	COLOR 1,3
CONT	Continua a execução de um programa interrompido por uma instrução STOP ou (BREAK).	CONT
COS	Retorna o cosseno do ângulo dado em radianos.	Y = COS(7)
CSAVE	Salva um programa na fita cassete (o nome do programa pode ter no máximo oito caracteres)... Se for especificado um A após o nome, ele será salvo no formato ASCII.	CSAVE CSAVE "PROG" CSAVE "PROG", A
CSAVEM	Grava um arquivo em linguagem de máquina.	CSAVEM X, 4E, 6F, 5F

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
DATA	Armazena dados dentro do programa. Use a instrução READ para ler estes dados.	DATA 5, 3, PAPEL DATA PERA, UVA
DEF FN	Define uma função numérica.	DEF FN(X) = X*3
DEFUSRn	Define o endereço da entrada da função USRn.	DEFUSR5 = 45643
DEL	Permite eliminar linhas de programas. DEL- Elimina todo o programa DEL n Elimina a linha n DEL n- Elimina todas as linhas depois da n DEL -n Elimina todas as linhas até n DEL n-m Elimina todas as linhas entre n e m	 DEL- DEL 100 DEL 200- DEL -300 DEL 20-60
DIM	Reserva espaço na memória para vetores e matrizes numéricas ou strings.	DIM R(12), T(14,27) DIM X$(5), Y$(7,9)
DLOAD	Carrega um programa Basic numa determinada taxa de transmissão (baud). 0 = 300 baud 1 = 1200 baud	DLOAD X,1
DRAW	Desenha uma linha começando num determinado ponto com um determinado comprimento e uma determinada cor. Esta instrução trata também de escalas, traça linhas em branco e executa substrings. Se não for indicado o ponto de início, o computador assume a posição inicial da instrução DRAW anterior ou o ponto (128,96).	DRAW "BM100,100;S10; V25;BR25;ND25;XA$;"
EDIT	Permite corrigir uma linha de programa. nC substitui n caracteres nD elimina n caracteres I permite inserir um novo caráter H elimina o resto da linha e permite inserir novos caracteres	EDIT 25

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	L mostra na tela a linha atual e permite correções nSc move o cursor à enésima ocorrência do caráter c, contando a partir da posição atual do cursor. X posiciona o cursor no fim da linha e permite inserir novos caracteres (SHIFT) (↑) sai do controle do comando EDIT n (ESPAÇO) move o cursor n posições para a direita n (←) move o cursor n posições para a esquerda	
END	Termina o programa. Não é possível usar o comando CONT após uma instrução END.	IF A = 1 THEN END END
EOF	Verifica se foi atingido o fim de um arquivo. Se EOF=0 ainda existem dados a serem lidos. Se EOF=1 não há mais dados. -1 - gravador cassete 0 - teclado	IF EOF(-1)<>0 THEN CLOSE IF EOF(-1) = 0 THEN INPUT#-1, A$
EXEC	Transfere o controle para um programa em linguagem de máquina começando no endereço especificado. Se o endereço for omitido, o controle é transferido para o endereço indicado no último comando CLOADM.	EXEC EXEC 32453
EXP	Retorna a exponencial natural (e^n).	Y = EXP(7)
FIX	Retorna a parte inteira.	Y = FIX(7.6)
FOR...TO...STEP/NEXT	Cria um loop no programa que o computador repete do primeiro ao último número especificado. Use STEP para indicar de quanto é o incremento a cada passagem pelo NEXT. Se não for indicado o STEP, é usado o número 1.	FOR X=2 TO 5/NEXT X FOR A=1 TO 10 STEP 5 NEXT A FOR M=30 TO 15 STEP 5/NEXT M

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
GET	Lê o conteúdo (gráfico) de um retângulo para dentro de um vetor para ser usado futuramente pela instrução PUT.	GET (5,20)-(3,8), G
GOSUB	Transfere o controle para uma rotina BASIC começando na linha indicada.	GOSUB 500 GOSUB 1000
GOTO	Desvia a execução do programa para a linha indicada.	GOTO 10 GOTO 1110
HEX$	Retorna o valor hexadecimal.	PRINT HEX$(30) Y = HEX$(X/16)
IF teste THEN ação 1 ELSE ação 2	Faz o teste. Se o resultado for verdadeiro, o computador executa a ação 1. Caso contrário (falso), executa a ação 2.	IF A=5 THEN 30 IF B$="BRASIL" THEN PRINT "CORRETO" ELSE PRINT "ERRADO"
INKEY$	Testa o teclado e fornece o código ASCII da tecla pressionada (caso tenha sido pressionada alguma).	A$ = INKEY$
INPUT	Interrompe a execução do programa e faz o computador esperar a entrada de dados pelo teclado.	INPUT X$ INPUT "NOME"; X$ INPUT A, B$
INPUT#-1	Lê dados do gravador cassete.	INPUT#-1, A
INSTR	Pesquisa a primeira ocorrência do string "alvo" no string "pesquisa" começando na posição "pos". Retorna a posição em que foi achado o "alvo".	PRINT INSTR(5,X$,Y$)
INT	Retorna a parte inteira da expressão fornecida.	X = INT(A*5.2)

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
JOYSTK	Retorna a coordenada horizontal ou vertical do joystick esquerdo ou direito. 0 - Coord. horizontal do joystick direito 1 - Coord. vertical do joystick direito 2 - Coord. horizontal do joystick esquerdo 3 - Coord. vertical do joystick esquerdo	X = JOYSTK(0) Y = JOYSTK(1)
LEFT$	Retorna os n primeiros caracteres (contando a partir da esquerda) do string fornecido.	P$ = LEFT$(M$,7)
LEN	Retorna o comprimento em bytes, do string fornecido.	X = LEN(X$)
LET	Atribui um valor a uma variável. Esta instrução é opcional.	LET P$="PGM"
LIST	Mostra todo ou parte do programa na tela.	LIST LIST 10-50 LIST -1000 LIST 1000- LIST 20
LLIST	Imprime todo ou parte do programa na impressora.	LLIST LLIST 10-100
LINE	Traça uma linha da posição (X1,Y1) até a posição (X2,Y2). Se (X1,Y1) for omitido, é usado o último ponto fornecido ou (128,96). PSET seleciona a cor do primeiro plano e PRESET seleciona a cor do fundo. B traça um retângulo com as coordenadas (X1,Y1) e (X2,Y2) nos cantos opostos. BF preenche este retângulo com a cor do primeiro plano.	LINE (5,3)-(6,6),PSET
LINE INPUT	Lê uma linha inteira do teclado.	LINE INPUT "RESPOSTA" ;R$

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
LOG	Retorna o logarítmo natural (base e).	Y = LOG(353)
MEM	Indica qual o tamanho do espaço que ainda permanece livre na memória do computador.	PRINT MEM
MID$	Retorna um substring do string fornecido começando na posição indicada com um comprimento determinado.	PRINT MID$ (A$,3,7)
MID$	Substitui uma parte de um string por outro string.	MID$(A$,4,2)="KS"
MOTOR	Liga ou desliga o gravador cassete.	MOTOR ON MOTOR OFF
NEW	Apaga o conteúdo da memória.	NEW
ON... GOSUB...	Transfere o controle para uma das subrotinas especificadas dependendo do valor da expressão.	ON Y*A GOSUB 1, 5, 7
ON... GOTO...	Desvia a execução do programa para uma das linhas indicadas dependendo do valor da expressão.	ON X+2 GOTO 5, 7, 9
OPEN	Abre um arquivo da seguinte forma: 0 - Tela ou teclado -1 - Gravador cassete -2 - Impressora A abertura pode ser: I - Entrada O - Saída OBS.: Não é preciso abrir nenhum arquivo para ler dados do teclado ou mostrar informações na tela.	OPEN "I",#-1, "ARQ" OPEN "O",#-1, "DADOS"

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
PAINT	Desenha um gráfico na tela começando no ponto (X,Y) com cor c e terminando no bordo da cor especificada.	PAINT (10,30),4,2
PCLEARn	Reserva n partições de memória de 1,5K para gráficos.	PCLEAR 8
PCLSc	Limpa e coloca a cor c na tela. Se o código da cor for omitido, é mantida a cor de fundo atualmente em uso.	PCLS 3
PCOPY	Copia gráficos da partição origem para a partição destino.	PCOPY 5, TO 6
PEEK	Retorna o conteúdo do endereço de memória especificado.	A = PEEK(32706)
PLAY	Soa o tom da nota (A-G ou 1-12), oitava(O), volume(V), comprimento de nota(L), tempo(T), pausa(P) e permite a execução de substrings.	PLAY "L1;A#;P8;V10; T3;L2;B-;9;XA$;"
PMODE	Indica a resolução gráfica da partição de memória reservada para gráficos.	PMODE 4,3
POINT	Testa se o ponto indicado está ativado ou desativado. O valor retornado pode ser: -1 - o ponto está em modo texto 0 - o ponto está desativado Cod - código da cor que o ponto possui	IF POINT(15,12)=3 THEN PRINT "AZUL"
POKE	Coloca um código (0-255) no endereço de memória especificado.	POKE 15872, 123
POS	Retorna a posição atual de impressão.	PRINT TAB(8) POS(0)
PPOINT	Testa se o ponto gráfico está ativado ou desativado e retorna o código da sua cor.	PPOINT (13,35)

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
PRESET	Devolve a cor de fundo para um determinado ponto.	PRESET(5,6)
PRINT	Imprime a lista de dados fornecida no periférico indicado. Se você não indicar o periférico, os dados serão mostrados na tela. Consulte o comando OPEN para conhecer os códigos dos periféricos.	PRINT "COLOR" PRINT B$ PRINT#-1,A$,B$ PRINT#-2,A$;"COLOR"
PRINT#-1	Transfere dados para o gravador cassete.	PRINT#-1,A
PRINT#-2	Transfere dados para a impressora.	PRINT#-2,CAP$
PRINT TAB	Move o cursor para uma determinada coluna antes de imprimir os dados.	PRINT TAB(5)"MARCEL"
PRINT USING	Imprime os números num formato pré-determinado. # campo numérico (um dígito por #) . ponto decimal , imprime uma vírgula a cada grupo de três algarismos ** preenche os espaços a esquerda com asteriscos $ imprime um símbolo $ antes do número $$ símbolo $ flutuante **$ símbolos $ e * flutuantes + na primeira posição, imprime o sinal do número no início. Na última posição imprime o sinal logo após o número.	PRINT USING "###"; 12 PRINT USING "##.#"; 1.5 PRINT USING "####,"; 44.0 PRINT USING "**##.##"; 33.3 PRINT USING "$##.##"; 33.3 PRINT USING "$$##.#"; 11.54 PRINT USING "**$##.###"; 8.345 PRINT USING "+#.#"; -2.3

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	↑↑↑↑ formato exponencial	PRINT USING "### . # ↑↑↑↑ "; 546
	- sinal negativo depois de um número negativo	PRINT USING "##.#-"; -5.3
	! imprime o primeiro caráter de um string	PRINT USING "!";"AZUL"
	% espaços % imprime um string com um comprimento igual a dois, mais o número de espaços entre os símbolos de percentagem.	PRINT USING "% %"; "AMARELO"
PRINT @	Imprime os dados na tela, à partir da posição especificada.	PRINT @ 194,"01"; A$
PSET	Ativa a posição (x,y) da tela com a cor c. Se for omitido o código da cor, é usada a cor do primeiro plano.	PSET (5,6,3)
PUT	Transfere o gráfico da partição origem para um retângulo na tela. O tamanho do retângulo tem que ser igual ao retângulo gerado com a instrução GET.	PUT (3,2)-(5,6), V, PSET
READ	Lê o próximo item de uma instrução DATA e atribui o dado a uma variável especificada.	READ A$ READ A, B$, C
REM	Permite a colocação de comentários num programa. Tudo que for digitado após o REM (antes de pressionar (ENTER) é ignorado pelo computador).	REM ISTO E' UM COMENTARIO
RENUM	Permite renumerar as linhas de um programa.	RENUM 1000,5,100
RESET	Desativa o ponto ativado pelo comando SET.	RESET (14,15)
RESTORE	Retorna o indicador da instrução DATA para a sua posição inicial.	RESTORE

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
RETURN	Transfere o controle do programa para a linha imediatamente abaixo do último GOSUB executado.	RETURN
RIGHT$	Retorna os n primeiros caracteres (contando a partir da direita) do string fornecido.	Z$ = RIGHT$(A$,5)
RND	Retorna um número inteiro aleatório entre 1 e o número indicado.	A = RND(15)
RUN	Executa um programa.	RUN
SCREEN	Determina o modo da tela (0=texto e 1=gráfico) e o conjunto de cores que será usado (0 ou 1).	SCREEN 1,1
SET	Ativa a posição especificada da tela na cor indicada.	SET(14,13,3)
SGN	Retorna o sinal da expressão numérica fornecida. -1 - se a expressão for negativa 0 - se a expressão for zero 1 - se a expressão for positiva	A = SGN(-4*B) B = SGN(-5) B = SGN(438.37)
SIN	Retorna o seno do ângulo dado em radianos.	Y = SIN(5*B/A)
SKIPF	Pula para o próximo programa do cassete ou para o fim do programa indicado.	SKIPF SKIPF "PGM"
SOUND	Soa um determinado tom com uma determinada duração.	SOUND 128,3
STOP	Interrompe a execução de um programa.	STOP
STRING$	Retorna um string de comprimento c contendo um código ASCII ou o primeiro caráter do string fornecido.	A$=STRING$(5,"*")

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
STR$	Converte uma expressão numérica em string.	S$=STR$(A*B)
SQR	Retorna a raiz quadrada.	S = SQR(24)
TAB	Inicia a impressão dos dados a partir da coluna indicada.	PRINT TAB(2) "COLOR" PRINT#-2, TAB(5); A$
TAN	Retorna a tangente do ângulo dado em radianos.	Y = TAN(5.3)
TIMER	Retorna o conteúdo do cronômetro ou inicializa-o (0-65535)	PRINT TIMER TIMER = 0
TROFF	Desativa o rastejador de programa (program tracer).	TROFF
TRON	Ativa o rastejador de programa (program tracer).	TRON
USR	Chama uma rotina em linguagem de máquina cujo endereço está armazenado nas posições 275 e 276.	X = USR(Y)
VAL	Converte um string em um número.	A = VAL(B$)
VARPTR	Retorna o endereço de uma variável.	Y = USR(VARPTR(X))

APÊNDICE H

CARACTERES DO TECLADO

(←) - Apaga o último caráter digitado; move o cursor uma posição para trás. O cursor é o ponto que fica pis - cando na tela.

(SHIFT) (←) - Volta o cursor para o início da linha atual, apagando tudo que foi digitado.

(BREAK) - Interrompe a execução do programa.

(CLEAR) - Limpa a tela.

(ENTER) - Indica o fim da linha atual.

(ESPACO) - Gera um caráter em branco e move o cursor uma posição para frente.

(SHIFT) (@) - Provoca uma pausa na execução do programa. Pressione qualquer tecla para continuar.

(SHIFT) (0) - Permite a passagem do modo letras maiúsculas para o modo letras minúsculas e vice-versa. As letras minúsculas são representadas por letras maiúsculas em cores reversas.

SÍMBOLOS BASIC

" " - Indica que a informação entre aspas é uma constante (string).

: - Separa os comandos escritos numa mesma linha.

( ) - Indica ao computador que as operações entre parênteses devem ser executadas primeiro.

; - Faz com que as constantes e variáveis sejam impressas coladas umas nas outras.

APÊNDICE H

OPERADORES BASIC

+	combina strings (concatenação)
+	adição
-	subtração
*	multiplicação
/	divisão
=	igualdade
>	maior que
> = ou = >	maior que ou igual a
< = ou = <	menor que ou igual a
<	menor que
< > ou > <	diferente de
AND	conjunção E
OR	conjunção OU
NOT	advérbio NÃO

APÊNDICE I

CORES DO COLOR BASIC EXTENDIDO

Existe um código para cada uma das 9 cores que você pode usar com o COLOR BASIC EXTENDIDO.

CÓDIGO	COR
0	preto
1	verde
2	amarelo
3	azul
4	vermelho

CÓDIGO	COR
5	bege
6	azul esverdeado
7	carmim
8	laranja

A tonalidade da cor pode variar um pouco dependendo do ajuste da televisão. A cor 0 (preto) indica ausência de cor.

TABELA DE CONJUNTO DE CORES

PMODE Nº	CONJUNTO DE COR	COMBINAÇÃO COM DUAS CORES	COMBINAÇÃO COM QUATRO CORES
4	0	preto/verde	-
	1	preto/bege	-
3	0	-	verde/amarelo/azul/vermelho
	1	-	bege/azul esv./carmim/laranja
2	0	preto/verde	-
	1	preto/bege	-
1	0	-	verde/amarelo/azul/vermelho
	1	-	bege/azul esv./carmim/laranja
0	0	preto/verde	-
	1	preto/bege	-

APÊNDICE J

NOTAS MUSICAIS

NÚMERO	NOTA
1	C
2	C#/D-
3	D/E-
4	E-/D#
5	E/F-
6	F/E#

NÚMERO	NOTA
7	F#/G-
8	G
9	G#/A-
10	A
11	A#/B-
12	B

NOTA: O comando PLAY não reconhece a notação "B#" ou "C-", você tem que usar os números 1 e 12 respectivamente ou substituir C por B# e B por C- . Caso tente usar uma destas notações ocorrerá um erro FC.

APÊNDICE K

CONVERSÃO DE BASE

DEC.	BINÁRIO	HEX.	OCTAL
0	00000000	00	000
1	00000001	01	001
2	00000010	02	002
3	00000011	03	003
4	00000100	04	004
5	00000101	05	005
6	00000110	06	006
7	00000111	07	007
8	00001000	08	010
9	00001001	09	011
10	00001010	0A	012
11	00001011	0B	013
12	00001100	0C	014
13	00001101	0D	015
14	00001110	0E	016
15	00001111	0F	017
16	00010000	10	020
17	00010001	11	021
18	00010010	12	022
19	00010011	13	023
20	00010100	14	024
21	00010101	15	025
22	00010110	16	026
23	00010111	17	027
24	00011000	18	030
25	00011001	19	031
26	00011010	1A	032
27	00011011	1B	033
28	00011100	1C	034
29	00011101	1D	035

DEC.	BINÁRIO	HEX.	OCTAL
30	00011110	1E	036
31	00011111	1F	037
32	00100000	20	040
33	00100001	21	041
34	00100010	22	042
35	00100011	23	043
36	00100100	24	044
37	00100101	25	045
38	00100110	26	046
39	00100111	27	047
40	00101000	28	050
41	00101001	29	051
42	00101010	2A	052
43	00101011	2B	053
44	00101100	2C	054
45	00101101	2D	055
46	00101110	2E	056
47	00101111	2F	057
48	00110000	30	060
49	00110001	31	061
50	00110010	32	062
51	00110011	33	063
52	00110100	34	064
53	00110101	35	065
54	00110110	36	066
55	00110111	37	067
56	00111000	38	070
57	00111001	39	071
58	00111010	3A	072
59	00111011	3B	073

DEC.	BINÁRIO	HEX.	OCTAL
60	00111100	3C	074
61	00111101	3D	075
62	00111110	3E	076
63	00111111	3F	077
64	01000000	40	100
65	01000001	41	101
66	01000010	42	102
67	01000011	43	103
68	01000100	44	104
69	01000101	45	105
70	01000110	46	1060
71	01000111	47	107
72	01001000	48	110
73	01001001	49	111
74	01001010	4A	112
75	01001011	4B	113
76	01001100	4C	114
77	01001001	4D	115
78	01001110	4E	116
79	01001111	4F	117
80	01010000	50	120
81	01010001	51	121
82	01010010	52	122
83	01010011	53	123
84	01010100	54	124
85	01010101	55	125
86	01010110	56	126
87	01010111	57	127
88	01011000	58	130
89	01011001	59	131

APÊNDICE K

DEC.	BINÁRIO	HEX.	OCTAL
90	01011010	5A	132
91	01011011	5B	133
92	01011100	5C	134
93	01011101	5D	135
94	01011110	5E	136
95	01011111	5F	137
96	01100000	60	140
97	01100001	61	141
98	01100010	62	142
99	01100011	63	143
100	01100100	64	144
101	01100101	65	145
102	01100110	66	146
103	01100111	67	147
104	01101000	68	150
105	01101001	69	151
106	01101010	6A	152
107	01101011	6B	153
108	01101100	6C	154
109	01101101	6D	1550
110	01101110	6E	156
111	01101111	6F	157
112	01110000	70	160
113	01110001	71	161
114	01110010	72	162
115	01110011	73	163
116	01110100	74	164
117	01110101	75	165
118	01110110	76	166
119	01110111	77	167
120	01111000	78	170
121	01111001	79	171
122	01111010	7A	172
123	01111011	7B	173

DEC.	BINÁRIO	HEX.	OCTAL
124	01111100	7C	174
125	01111101	7D	175
126	01111110	7E	176
127	01111111	7F	177
128	10000000	80	200
129	10000001	81	201
130	10000010	82	202
131	10000011	83	203
132	10000100	84	204
133	10000101	85	205
134	10000110	86	206
135	10000111	87	207
136	10001000	88	210
137	10001001	89	211
138	10001010	8A	212
139	10001011	8B	213
140	10001100	8C	214
141	10001001	8D	215
142	10001110	8E	216
143	10001111	8F	217
144	10010000	90	220
145	10010001	91	221
146	10010010	92	222
147	10010011	93	223
148	10010100	94	224
149	10010101	95	225
150	10010110	96	226
151	10010111	97	227
152	10011000	98	230
153	10011001	99	231
154	10011010	9A	232
155	10011011	9B	233
156	10011100	9C	234
157	10011101	9D	235

DEC.	BINÁRIO	HEX.	OCTAL
158	10011110	9E	236
159	10011111	9F	237
160	10100000	A0	240
161	10100001	A1	241
162	10100010	A2	242
163	10100011	A3	243
164	10100100	A4	244
165	10100101	A5	245
166	10100110	A6	246
167	10100111	A7	247
168	10101000	A8	250
169	10101001	A9	251
170	10101010	AA	252
171	10101011	AB	253
172	10101100	AC	254
173	10101101	AD	255
174	10101110	AE	256
175	10101111	AF	257
176	10110000	B0	260
177	10110001	B1	261
178	10110010	B2	262
179	10110011	B3	263
180	10110100	B4	264
181	10110101	B5	265
182	10110110	B6	266
183	10110111	B7	267
184	10111000	B8	270
185	10111001	B9	271
186	10111010	BA	272
187	10111011	BB	273
188	10111100	BC	274
189	10111101	BD	275
190	10111110	BE	276
191	10111111	BF	277

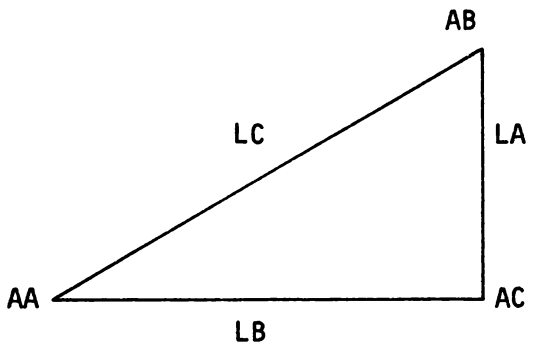
APÊNDICE K

DEC.	BINÁRIO	HEX.	OCTAL
192	11000000	C0	300
193	11000001	C1	301
194	11000010	C2	302
195	11000011	C3	303
196	11000100	C4	304
197	11000101	C5	305
198	11000110	C6	306
199	11000111	C7	307
200	11001000	C8	310
201	11001001	C9	311
202	11001010	CA	312
203	11001011	CB	313
204	11001100	CC	314
205	11001101	CD	315
206	11001110	CE	316
207	11001111	CF	317
208	11010000	D0	320
209	11010001	D1	321
210	11010010	D2	322
211	11010011	D3	323
212	11010100	D4	324
213	11010101	D5	325
214	11010110	D6	326
215	11010111	D7	327
216	11011000	D8	330
217	11011001	D9	331
218	11011010	DA	332
219	11011011	DB	333
220	11011100	DC	334
221	11011101	DD	335
222	11011110	DE	336
223	11011111	DF	337
224	11100000	E0	340

DEC.	BINÁRIO	HEX.	OCTAL
225	11100001	E1	341
226	11100010	E2	342
226	11100011	E3	343
228	11100100	E4	344
229	11100101	E5	345
230	11100110	E6	346
231	11100111	E7	347
232	11101000	E8	350
233	11101001	E9	351
234	11101010	EA	352
235	11101011	EB	353
236	11101100	EC	354
237	11101101	ED	355
238	11101110	EE	356
239	11101111	EF	357
240	11110000	F0	360
241	11110001	F1	361
242	11110010	F2	362
243	11110011	F3	363
244	11110100	F4	364
245	11110101	F5	365
246	11110110	F6	366
247	11110111	F7	367
248	11111000	F8	370
249	11111001	F9	371
250	11111010	FA	372
251	11111011	FB	373
252	11111100	FC	374
253	11111101	FD	375
254	11111110	FE	376
255	11111111	FF	377

APÊNDICE L

FÓRMULAS TRIGONOMÉTRICAS



Lado a = LA (lado oposto ao ângulo A)
Lado b = LB (lado adjacente ao ângulo A)
Lado c = LC (hipotenusa)
Ângulo A = AA
Ângulo B = AB
Ângulo C = AC

DESCRIÇÃO	FÓRMULA	COMANDO BASIC
Número total de graus de um triângulo	$180^{o} = A + B + C$	TTL = AA + AB + AC
Cálculo da área dados lado A e ângulos B e C	$A = 180 - (B + C)$	AA = 180 - (AB+AC) converter AA, AB e AC para radianos
	$\text{Área} = \frac{a^2 \text{ sen } B \,.\, \text{sen } C}{2 \text{ sen } A}$	AREA = LA↑2*SIN(AB)*SIN(AC)/(2*SIN(AA))
Dados os lados a, b e c	$s = 1/2\ (a + b + c)$	S = (LA + LB + LC) / 2
	$\text{Área} = \sqrt{s(s-a)(s-b)(s-c)}$	AREA = SQR(S*(S - LA) * (S - LB) * (S - LC))
Lei dos senos	$\frac{a}{b} = \frac{\text{sen } A}{\text{sen } B}$ ou $a = \frac{\text{sen } A}{\text{sen } B} \,.\, b$	LA = (SIN(AA)/SIN(AB))*LB

APÊNDICE L

DESCRIÇÃO	FÓRMULA	COMANDO BASIC
Lei dos cossenos	$a^2 = b^2 + c^2 - 2bc.\cos A$ ou $a = b^2 + c^2 - 2bc.\cos A$	LA = SQR(LB↑2 - LC↑2 - 2 * LB * LC * COS (AA))
Lei das tangentes	$\frac{a - c}{a + c} = \frac{\tan 1/2\ (A - C)}{\tan 1/2\ (A + C)}$ ou $\tan 1/2(A-C) = \frac{a-c}{a+c} .\tan 1/2\ (A+C)$	REM Y = TAN((AA - AC)/2) Y = (LA - LC)/(LA + LC) * TAN((AA+AC)/2)
Dados três lados determinar o ângulo	$s = 1/2\ (a + b + c)$ $r = \sqrt{\frac{(s-a)(s-b)(s-c)}{s}}$ $A = 2 \arctan\ (\frac{r}{s - a})$	S = (LA + LB + LC)/2 R = SQR((S - LA)*(S-LB)*(S-LC)/S) AA = 2 * ATN(R/(S-LA))
Equação quadrática	$ax^2 + bx + c = 0$ $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$	REM A*X↑2 + B*X + C = 0 Z = B↑2 - 4*A*C X1 = (-B + SQR(Z))/(2*A) 'IF Z =0 X2 = (-B - SQR(Z))/(2*A) 'IF Z =0
Equações algébricas	$(a^x)^y = a^{xy}$ $a^{-x} = \frac{1}{a^x}$	Z = (A↑X)↑Y ou Z = A↑(X * Y) Z = A↑(-X) ou Z = 1/(A↑X)

APÊNDICE L

DESCRIÇÃO	FÓRMULA	COMANDO BASIC
	$\log x^y = y \cdot \log x$	Z = LOG(X↑Y) ou Z = Y*LOG(X)
	$\log xy = \log x + \log y$	Z = LOG (X*Y) ou Z = LOG(X) + LOG(Y)
	$\log \frac{x}{y} = \log x - \log y$	Z = LOG(X/Y) ou Z = LOG(X) - LOG(Y)

APÊNDICE L

FUNÇÃO	FUNÇÕES EXPRESSAS EM COMANDOS DO COLOR BASIC EXTENDIDO (x em radianos)
SECANTE	SEC(X) = 1/COS(X)
COSSECANTE	CSC(X) = 1/SIN(X)
COTANGENTE	COT(X) = 1/TAN(X)
ARCO SENO	ARCSIN(X) = ATN(X/SQR(-X*X+1))
ARCO COSSENO	ARCCOS(X) = -ATN(X/SQR(-X*X+1)) + 1.5708
ARCO SECANTE	ARCSEC(X) = ATN(SQR(X*X -1)) + (SGN(X) -1) * 1.5708
ARCO COSSECANTE	ARCCSC(X) = ATN(1/SQR(X*X-1)) + (SGN(X) -1) * 1.5708
ARCO COTANGENTE	ARCCOT(X) = -ATN(X) + 1.5708
SENO HIPERBÓLICO	SINH(x) = (EXP(X) - EXP(-X))/2
COSSENO HIPERBÓLICO	COSH(X) = (EXP(X) + EXP(-X))/2
TANGENTE HIPERBÓLICA	TANH(X) = -EXP(-X)/(EXP(X) + EXP(-X)) * 2 + 1
SECANTE HIPERBÓLICA	SECH(X) = 2/(EXP(X) + EXP(-X))
COSSECANTE HIPERBÓLICA	COSH(X) = 2/(EXP(X) - EXP(-X))
COTANGENTE HIPERBÓLICA	COTH(X) = EXP(-X)/EXP(X) - EXP(-X)) * 2 + 1
ARCO SENO HIPERBÓLICO	ARGSINH(X) = LOG(X+SQR(X*X+1))
ARCO COSSENO HIPERBÓLICO	ARGCOSH(X) = LOG(X+SQR(X*X-1))
ARCO TANGENTE HIPERBÓLICO	ARGTANH(X) = LOG((1-X)/(1-X))/2
ARCO SECANTE HIPERBÓLICO	ARGSECH(X) = LOG((SQR(-X*X+1)+1)/X)
ARCO COSSECANTE HIPERBÓLICO	ARGCSCH(X) = LOG((SGN(X) * SQR(X*X+1)+1)/X)
ARCO COTANGENTE HIPERBÓLICO	ARGCOTH(X) = LOG((X+1)/(X-1))/2

APÊNDICE L

ARCO SENO	-1< X <1
ARCO COSSENO	-1< X <1
ARCO SECANTE	X<-1 ou X>1
ARCO COSSECANTE	X< -1 ou X>1
ARCO COSSENO HIPERBÓLICO	X>1
ARCO TANGENTE HIPERBÓLICO	X*X <1
ARCO SECANTE HIPERBÓLICO	0<X<1
ARCO COSSECANTE HIPERBÓLICO	X<>0
ARCO COTANGENTE HIPERBÓLICO	X*X>1

Certos valores especiais são matematicamente indefinidos mas as funções do COLOR BASIC EXTENDIDO podem fornecer respostas inválidas:

TAN e SEC de 90 e 270 graus
COT e CSC de 0 e 180 graus

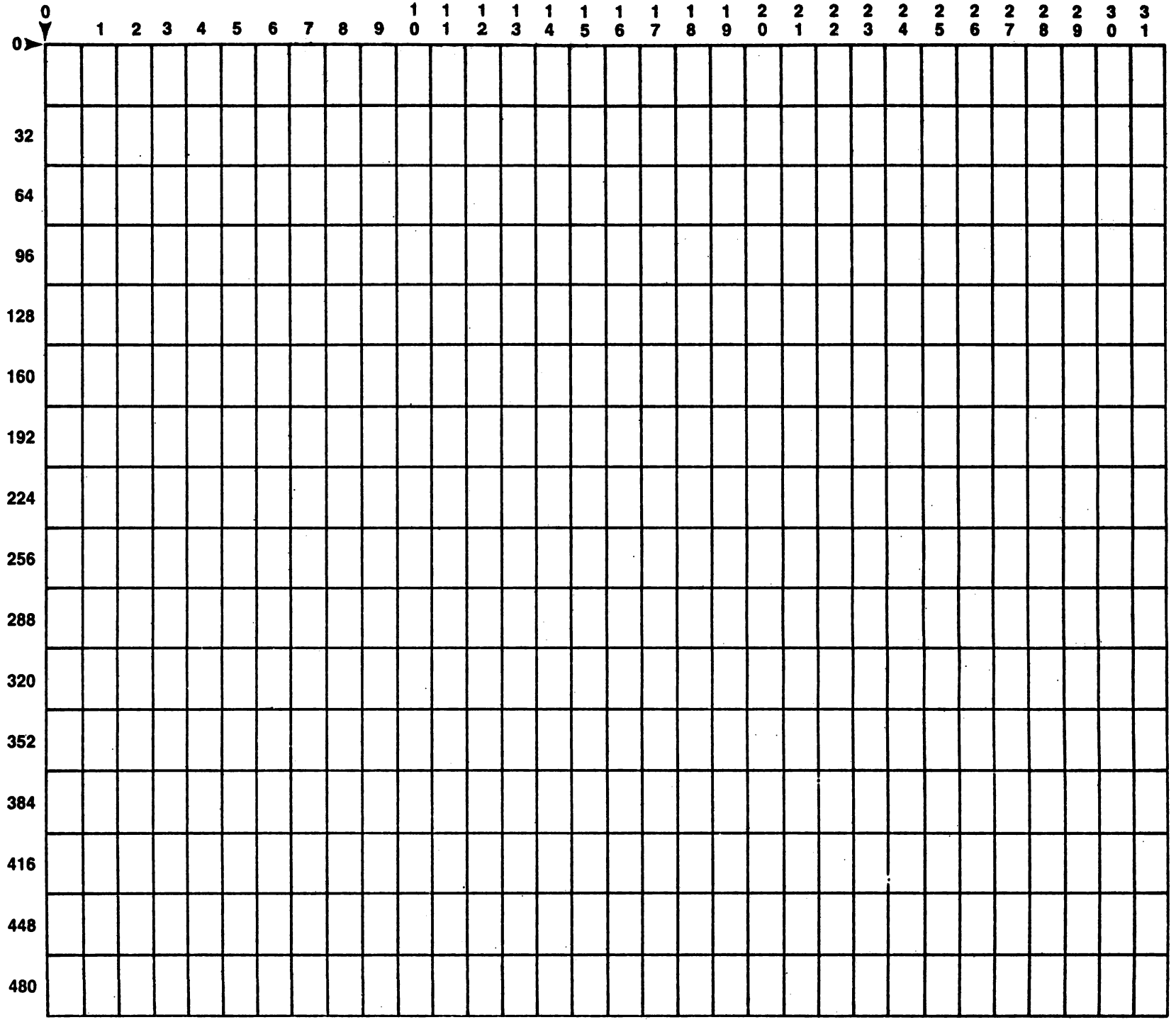
Por exemplo, TAN(1.5708) retorna um valor mas TAN (90*.01745329) retorna o erro divisão por zero apesar de 90*.01745329 = = 1.5708.

Os outros valores que não estão disponíveis para estas funções são:

ARCSIN (-1) = - PI/2
ARCSIN (1) = PI/2
ARCCOS (-1) = PI
ARCCOS (1) = 0
ARCSEC (-1) = - PI
ARCSEC (1) = 0
ARCCSC (-1) = - PI/2
ARCCSC (1) = PI/2

APENDICE M

GABARITO DE TELA 32 x 16



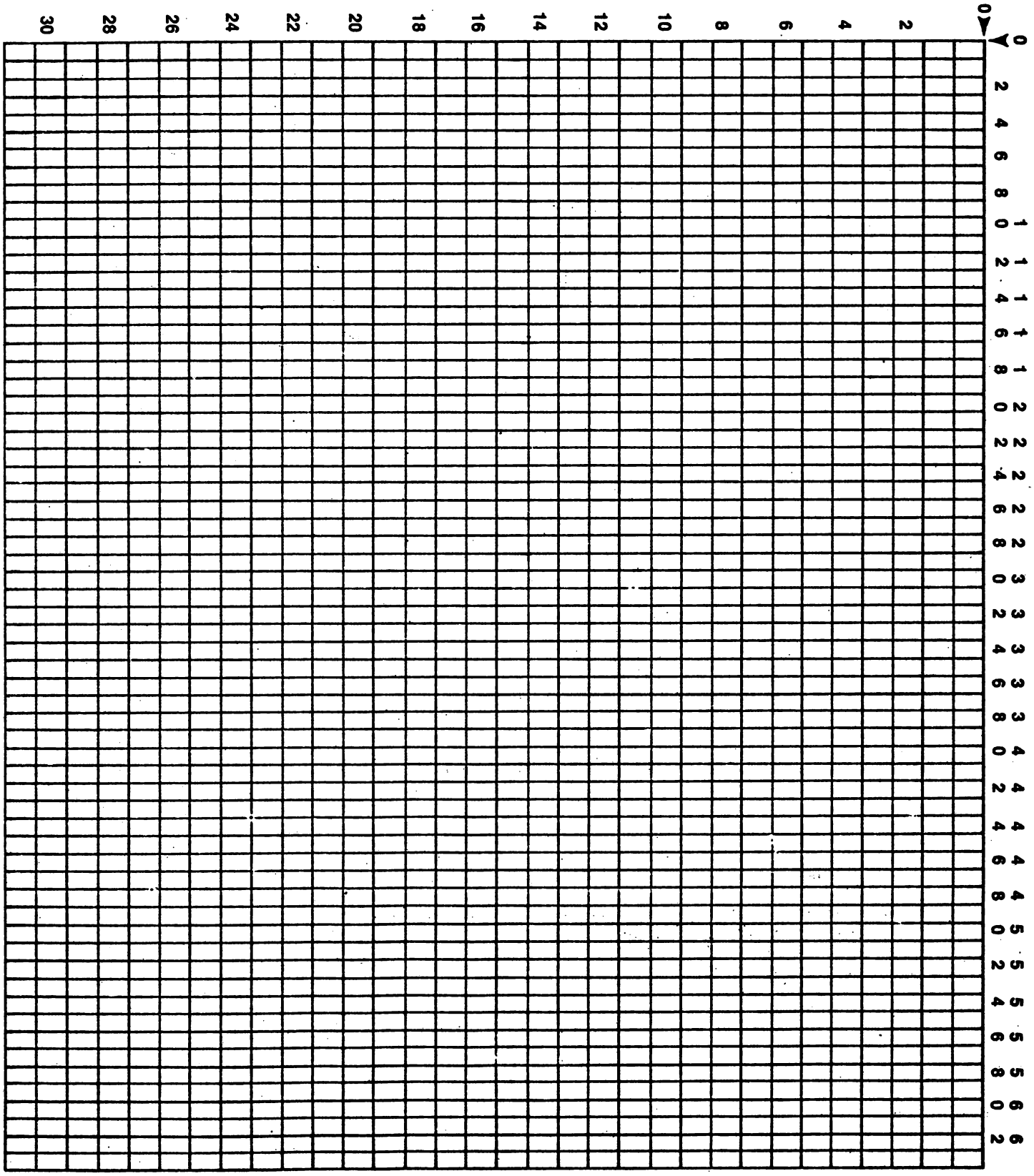
APÊNDICE M

GABARITO DE TELA 32 x 16

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319
320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351
352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383
384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415
416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447
448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479
480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	608	509	510	511

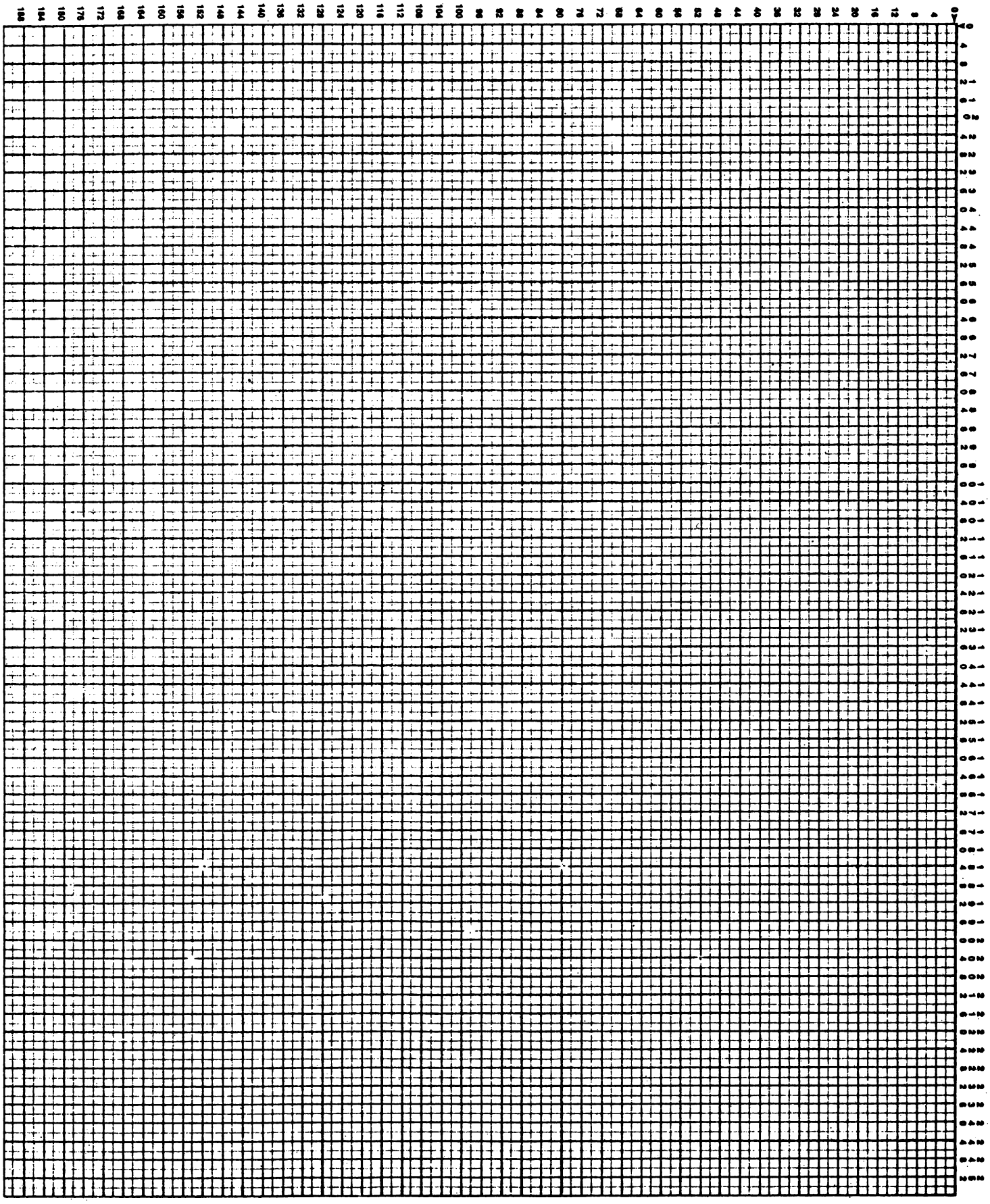
APÊNDICE M

GABARITO DE TELA 64 x 32



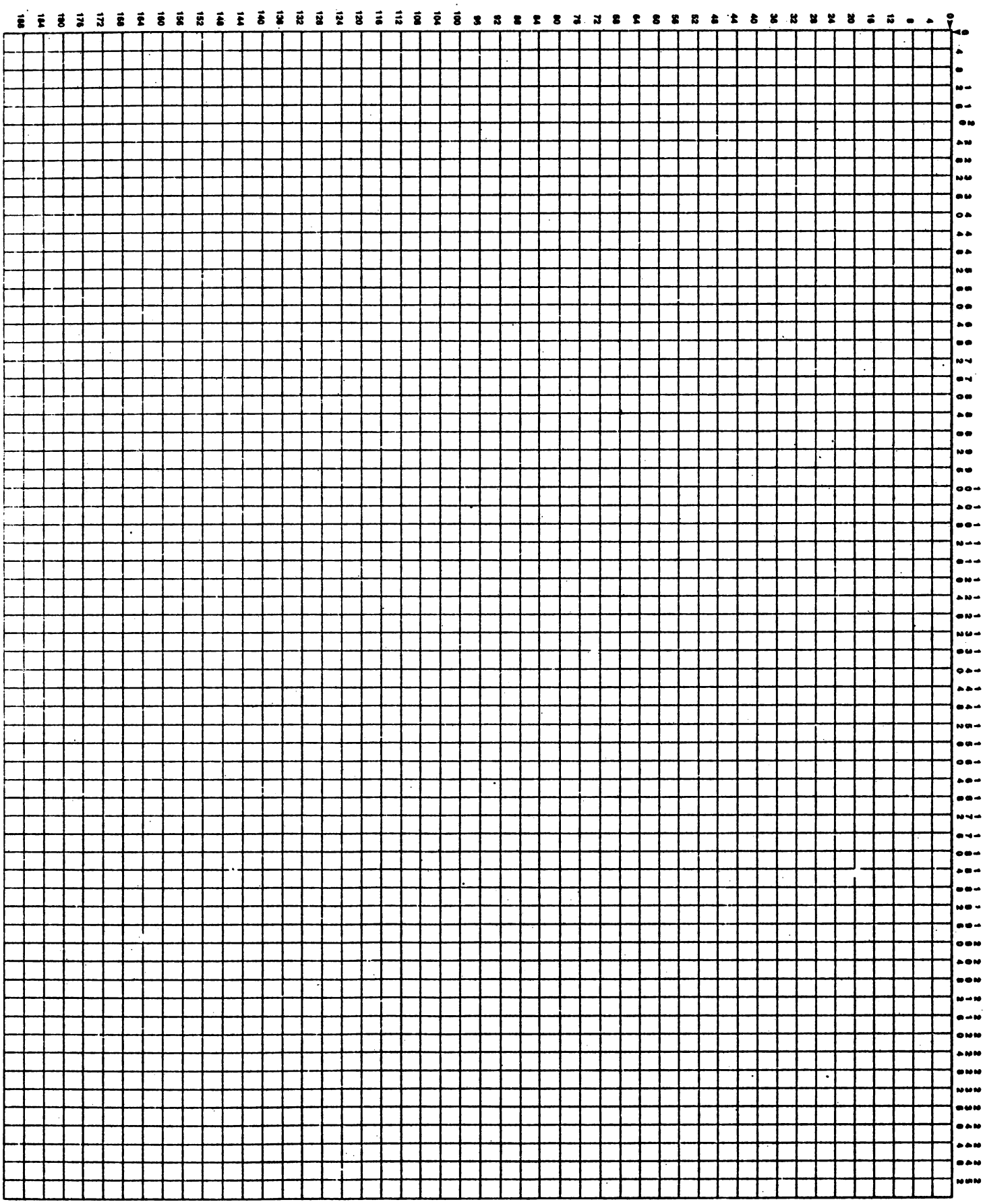
APÊNDICE M

GABARITO DE TELA 128 x 96



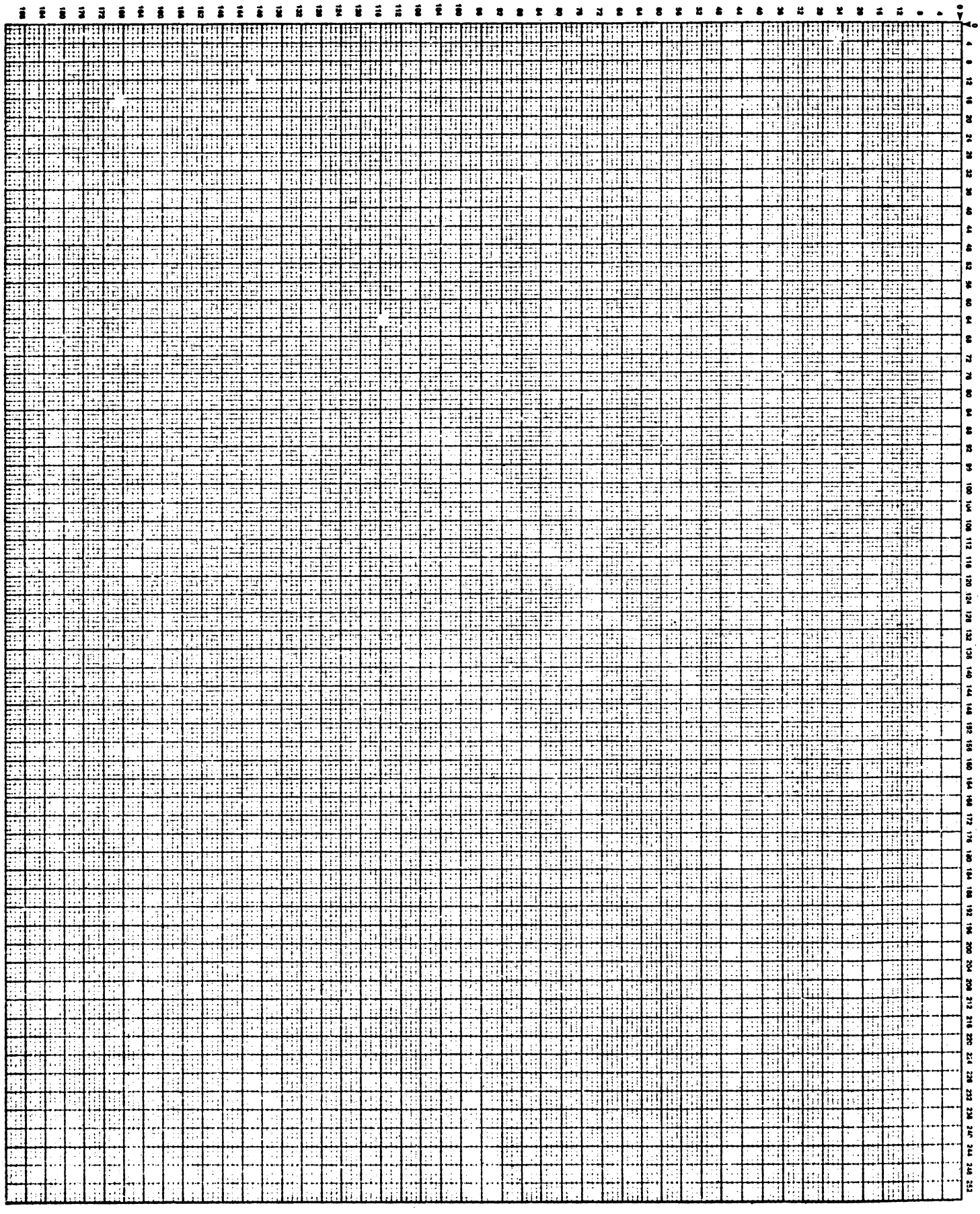
APÊNDICE M

GABARITO DE TELA 128 x 192



APÊNDICE M

GABARITO DE TELA 256 x 192



APÊNDICE N

PALAVRAS RESERVADAS DO COLOR BASIC EXTENDIDO

'	CLS	FOR	MID$	PPOINT	SQR
*	COLOR	GET	MOTOR	PRESET	STEP
+	CONT	GOSUB	NEW	PRINT	STOP
-	COS	GOTO	NEXT	PSET	STR$
REM	CSAVE	HEX$	NOT	PUT	STRING$
<	DATA	IF	OFF	READ	SUB
=	DEF	INKEY$	ON	RENUM	TAB
>	DEL	INPUT	OPEN	RESET	TAN
ABS	DIM	INSTR	OR	RESTORE	THEN
AND	DLOAD	INT	PAINT	RETURN	TIMER
ASC	DRAW	JOYSTK	PCLEAR	RIGHT$	TO
ATN	EDIT	LEFT$	PCLS	RND	TROFF
AUDIO	ELSE	LEN	PCOPY	RUN	TRON
CHR$	END	LET	PEEK	SCREEN	USING
CIRCLE	EOF	LINE	PLAY	SET	USR
CLEAR	EXEC	LIST	PMODE	SGN	VAL
CLOAD	EXP	LLIST	POINT	SIN	VARPTR
CLOADM	FIX	LOG	POKE	SKIPF	
CLOSE	FN	MEM	POS	SOUND	

APÊNDICE O

INFORMAÇÕES TÉCNICAS DO COLOR BASIC EXTENDIDO

Este apêndice é para pessoas que programam em linguagem de máquina e BASIC, que conhecem a aritmética de números binários e hexadecimais e que entendem os conceitos de bit e byte. O objetivo deste apêndice é permitir que você possa aproveitar ao máximo o potencial deste computador.

Caso queira entender e usar o sistema neste nível, mas não possui os conhecimentos necessários, recomendamos ler:

BASIC MICROPROCESSORS AND THE 6800 by RON BISHOP
Hayden Book Company, Inc.
Rochelle Park, New Jersey
1979

A ROM do COLOR BASIC EXTENDIDO possiu várias subrotinas que podem ser chamadas por programas em linguagem de máquina; algumas delas também podem ser chamadas em BASIC usando a função USR. Vamos descrever agora cada uma destas subrotinas respeitando a seguinte nomemclatura:

APÊNDICE O

NOME - *Endereço de entrada*
Operação executada
Condição de entrada
Condição de saída

> NOTA: O computador não reconhece o NOME da subrotina, este nome só serve como referência. O endereço de entrada é dado em hexadecimal e da seguinte maneira: é preciso dar um salto indireto para este endereço, e as condições de entrada e saída são fornecidas para programas em linguagem de máquina.

BLKIN = (A006)
Lê um bloco do gravador cassete
Condição de entrada:
O cassete tem que estar ligado e em "bir sync" (veja CSRDON). CBMFAD contém o endereço do buffer.
Condição de saída:
BLKTYP, que está na posição 7C (hexadecimal) contém o tipo de bloco que pode ser:

0 = início de arquivo (Header)
1 = dado
FF = fim de arquivo

BLKLEN, que está na posição 7D (hexadecimal), contém o número de bytes que existem no bloco (0-255).

Z = 1, A = CSRERR = 0 (se não houve erro).

Z = 0, A = CSRERR = 1 (se ocorrer um erro de paridade (checksum))

Z = 0, A = CSRERR = 2 (se ocorrer um erro de memória)

(NOTA: CSRERR = 81)

A menos que ocorra um erro de memória, X = CBUFAD + BLKLEN. Se ocorrer um erro de memória, X aponta para o endereço onde ocorreu o erro.

As interrupções são mascaradas e os registradores U e Y são preservados enquanto que os demais são modificados.

BLKOUT = (A008)

Grava um bloco no gravador cassete.

Condição de entrada:

A fita cassete tem que estar girando na sua velocidade normal e o código hexadecimal 55 tem que estar gravado se este for o primeiro bloco a ser gravado.

CBUFAD, que está na posição 7E (hexadecimal) contém o endereço do buffer

BLKTYP, que está na posição 7C (hexadecimal) contém o tipo de bloco

BLKLEN, que está na posição 7D (hexadecimal) contém o número de bytes que existem no bloco (0-255)

Condição de saída:

As interrupções são mascaradas.

APÊNDICE O

X = CBUFAD + BLKLEN e todos os registradores são modificados.

WRTLDR = (A00C)

Liga o gravador cassete e grava uma marca (LEADER).

Condição de entrada:

Não há.

Condição de saída:

Não há

CHROUT = (A002)

Grava um caráter num periférico

CHROUT grava um caráter no periférico indicado pelo conteúdo do endereço hexadecimal 6F (DEVNUM).

DEVNUM = -2 (impressora)

DEVNUM = 0 (tela)

Condição de entrada:

Na entrada, o caráter a ser transferido está no registro A.

Condição de saída:

Todos os registradores a menos do CC (hexadecimal) são mantidos.

APÊNDICE O

CSRDON = (A004)

Liga o cassete.

CSRDON liga o cassete e vai ao "bit sync" para leitura.

Condição de entrada:

Não há

Condição de saída:

FIRQ e IRO são mascarados e os registradores U e Y são preservados enquanto que os demais são modificados.

JOYIN = (A00A)

Testa a porta do joystick

JOYIN testa as portas dos quatro joysticks e armazena os seus valores de POTVAL até POTVAL+3.

JOYSTICK ESQUERDO	
Cima/Baixo	15A
Direita/Esqueda	15B

JOYSTICK DIREITO	
Cima/Baixo	15C
Direita/Esquerda	15D

Para CIMA/BAIXO, o menor valor é CIMA.

Para DIREITA/ESQUERDA, o menor valor é ESQUERDA.

Condição de entrada:

Não há.

Condição de saída:

O registrador Y é mantido enquanto que os demais são alterados.

APÊNDICE O

POLCAT = (A000)

Pesquisa o teclado.

Condição de entrada:

Não há

Condição de saída:

Z = 1, A = 0 (nenhuma tecla foi pressionada)

Z = 0, A = Código da tecla (se foi pressionada uma tecla)

O conteúdo dos registradores B e X são preservados enquanto que os demais são modificados.

SEQÜÊNCIA DE RESET

Quando o botão RESET é pressionado, o COLOR BASIC inicializa os chips PIAS e SAM e então a chave de re-início (RSTFLG) e o vetor de re-início são testados. Se RESTFLG=55 (hexadecimal) e RSTVEC apontar para NOP (12 hexadecimal), então o controle é transferido para o endereço que está em RSTVEC. Se estas condições não forem satisfeitas, a seqüência de inicialização do COLOR BASIC é interrompida.

APÊNDICE P

VARIÁVEIS DE CONTROLE DE IMPRESSORA

VARIÁVEL	ENDEREÇO HEXADECIMAL	ENDEREÇO DECIMAL	VALOR INICIAL HEXADECIMAL	VALOR DECIMAL
LPTBTD Baud				
MSB	0095	149	00	0
LSB	0096	150	57	87
LPTLND Retardador de linha				
MSB	0097	151	00	0
LSB	0098	152	01	1
LPTCFW Tamanho do campo com vírgula				
	0099	153	10	16
LPTLCF Último campo com vírgula				
	009A	154	70	112
LPTWID Tamanho da linha de impressão				
	009B	155	84	132
LPTPOS				
	009C	156	00	00

APÊNDICE P

O software deste computador usa as seguintes condições iniciais:

1. A taxa de transmissão é de 600 bauds
2. O tamanho da linha de impressão é de 132 caracteres
3. A impressora gera um sinal quando não está pronta
4. A impressora executa um "retorno de carro" automaticamente quando atingir a coluna 132.

A interface RS-232 usa um conector de 4 pinos tipo DIN. Você encontrará no Manual de Operação um diagrama detalhado do conector.

O pino número 4 é a saída do computador para a impressora;
O pino número 3 é o terra;
O pino número 1 não é usado pela impressora;
O pino número 2 deve ser conectado ao alarme da impressora ou ao indicador de situação.(status). Se sua impressora não possuir um indicador de status, então este pino deve ser ligado a uma corrente elétrica com uma voltagem positiva superior a 3 volts. Isto informa ao computador que a impressora está sempre pronta. Além disso, a variável que indica o retardador de linha tem que conter um valor correto.

Estamos fornecendo, a seguir, uma lista de valores para facilitar a ligação de qualquer impressora no computador.

APÊNDICE P

TAXA EM BAUD	VALOR DECIMAL (msb,lsb)	VALOR HEXADECIMAL
120 em Baud	458 (1 e 202)	01CA
300 em Baud	180	00BE
600 em Baud	87	0057
1200 em Baud	41	0029
2400 em Baud	18	0012

RETARDADOR DE LINHA	VALOR DECIMAL (",")	VALOR HEXADECIMAL
.288 segundos	64 e 0	4000
.576 segundos	128 e 0	8000
1.15 segundos	255 e 255	FFFF

COMPRIMENTO DA LINHA	VALOR DECIMAL (",")	VALOR HEXADECIMAL
16 caracteres por linha	16	10
32 caracteres por linha	32	20
64 caracteres por linha	64	40
255 caracteres por linha	255	FF

APÊNDICE P

A última variável de campo de vírgula deve indicar a largura do papel. (A largura do campo de vírgula é normalmente colocado em 16)

Na linguagem COLOR BASIC versão 1.0 o formato de saída para impressora é um bit de início e sete bits de dados (LSB primeiro) e dois bits de parada sem paridade.

NOTAS