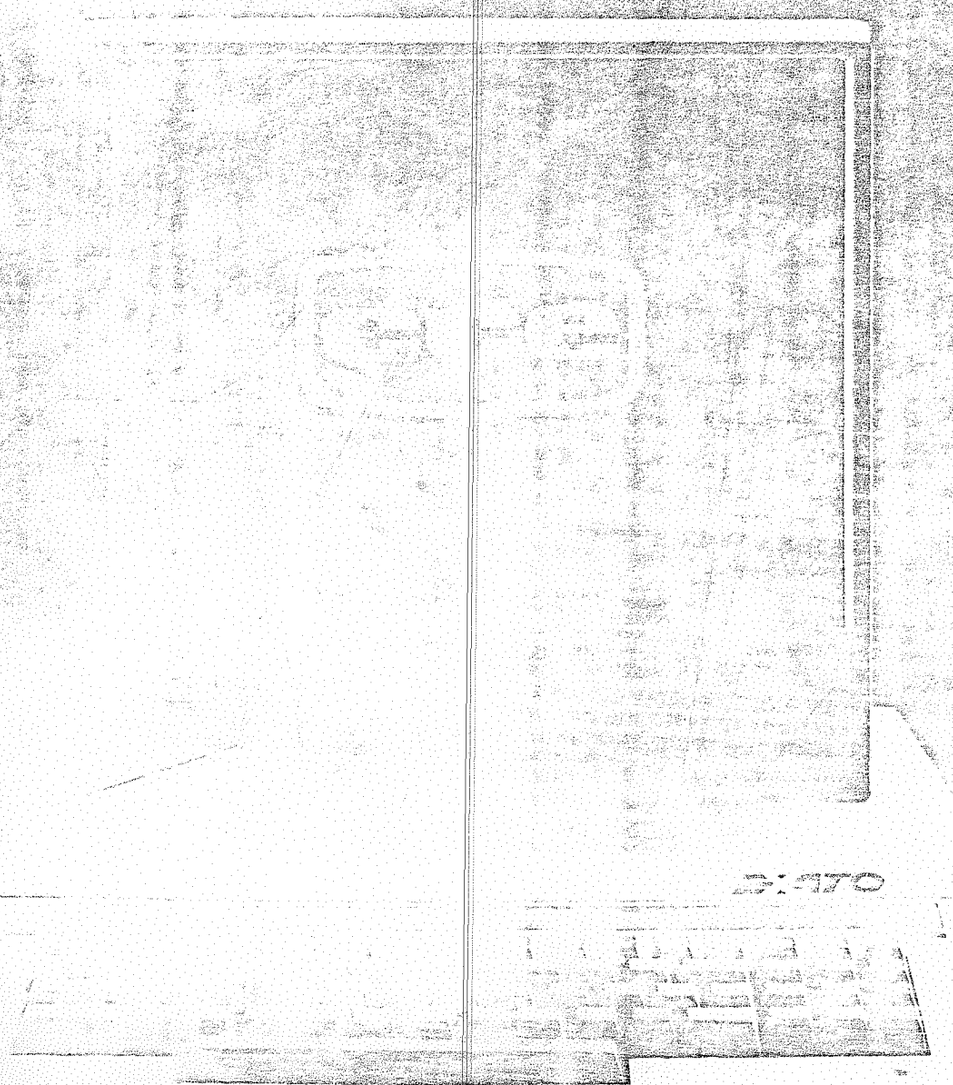
MICROCOMPUTADOR

EXATO MC-4000

MANUAL DE INSTRUÇÃO



EXATO

MANUAL DE INSTRUÇÃO

MICROCOMPUTADOR
MC-4000

2ª EDIÇÃO

CCE - Indústria e Comércio de Componentes Eletrônicos S/A - DIVISÃO DE INFORMÁTICA
Av. Otaviano Alves de Lima, 2724
CEP 02501 - São Paulo-SP - Brasil

BEM VINDO

Este manual é destinado a pessoas que querem aprender a programar na linguagem CCE BASIC. Com este manual, um computador EXATO, uma parcela de seu tempo e atenção, você descobrirá que não é difícil aprender a programar. No começo, como qualquer novidade, programar será estranho, mas este manual foi criado para sanar as dúvidas que possam aparecer. Em primeiro lugar, não há segredos que precisam ser conhecidos antes de se ler este manual. Revelamos tudo, e as explicações são feitas sequencialmente. Se você começar pelo princípio, experimente tudo, a medida que as novidades vão aparecendo, vá com calma e garantimos que você aprenderá a programar.

O verdadeiro segredo é dispor de tempo e experimentar tudo o que ensinamos. Você não poderá aprender a programar simplesmente através da leitura deste manual ou de outro livro. Assim como para aprender a tocar violino ou andar de bicicleta, não adianta ler manuais, é necessário praticar. Você fará erros e precisará corrigi-los e não deverá ficar frustrado por errar.

Se você já sabe programar, uma rápida leitura deste manual, servirá para familiarizá-lo com as características do CCE BASIC. Achamos que você ficará surpreso com a facilidade de fazer gráficos em Alta Resolução e da utilização de outras características que tornam o EXATO um ótimo computador para uma grande gama de aplicações.

Para se aprofundar no estudo das linguagens CCE BASIC e CCE INTEGER, leia o manual do BASIC do EXATO.

SUMARIO

CAPÍTULO 2

CONHECENDO O CCE BASIC

CAPÍTULO 3

PROGRAMAÇÃO ELEMENTAR

CAPITULO 5

CADEIAS E MATRIZES

APÊNDICES

CAPÍTULO 1

INICIANDO

INTRODUÇÃO

Este manual mostrará como ligar seu computador e servirá de guia enquanto você aprende a programá-lo. Se você já tiver experiência em programação, encontrará algumas novas características e vantagens na linguagem CCE BASIC, que a tornarão mais agradável. Se for um novato em programação, você sentirá que esta é muito agradável, pelas facilidades que ela apresenta. Se este é o seu caso, fique avisado que embora programação não seja difícil, só pode ser aprendida na prática. Falaremos a esse respeito mais adiante, mas lembre-se: este é um manual para ser usado e não simplesmente folheado.

Se você comprou seu EXATO de um revendedor autorizado, será bom que você ligue seu computador na loja, na presença do vendedor, para ter certeza que saberá ligá-lo em casa. Se você o recebeu como presente, não é difícil ligá-lo, é tão fácil quanto ligar um sistema estereofônico e não é necessário qualquer conhecimento técnico.

O QUE SERÁ NECESSÁRIO

Junto com este manual, na caixa de seu computador, você também deverá encontrar um cabo para ligar o EXATO a um gravador. Este cabo tem dois plugs em cada extremidade.

Além disso, e do EXATO propriamente dito, você necessitará de mais dois ítens a serem escolhidos dentre as opções descritas abaixo (nenhum destes ítens é fornecido junto com o computador).

1- a. um gravador cassete para ligá-lo ao EXATO e algumas fitas cassete
   ou
   b. o drive com interface para drives.

2- Você também necessitará de um dos seguintes ítens:

   a. um aparelho monitor (vídeo) a cores ou não e um cabo que tenha em uma extremidade um plug (também chamado conector macho tipo RCA) e na outra extremidade algum outro plug que encaixe no monitor. O revendedor poderá fornecer-lhe o cabo
   ou
   b. uma televisão comum, a cores ou não com os cabos de conexão.

O modulador de RF altera o sinal de saída do EXATO de modo que ele combine com o sinal que seu aparelho de TV espera receber. Há um modulador que acompanha o EXATO. A próxima seção indica como instalá-lo.

Um monitor ou uma TV em preto e branco, funcionará bem, mas não o deixará usufruir da vantagem do computador EXATO de produzir imagens a cores. As cores descritas neste manual aparecerão em diferentes tons de cinza, num monitor de TV preto e branco. Para que você possa utilizar a definição de cores, será necessário instalar um cartão CCE PAL-M CARD em seu EXATO.

3- Para utilização em jogos, você poderá adquirir:

a - um par de controladores simples (PADDLES), ou
b - um controlador do tipo orbital (JOYSTICK).

INSTALAÇÃO DO MODULADOR DE RF

A conexão do MODULADOR de RF ao EXATO é bastante simples. Siga as instruções abaixo:

1- confira que o EXATO esteja desligado da rede;

2- abra a tampa do micro puxando sua parte traseira (a mais afastada do teclado) para cima e em seguida deslizando-a para trás;

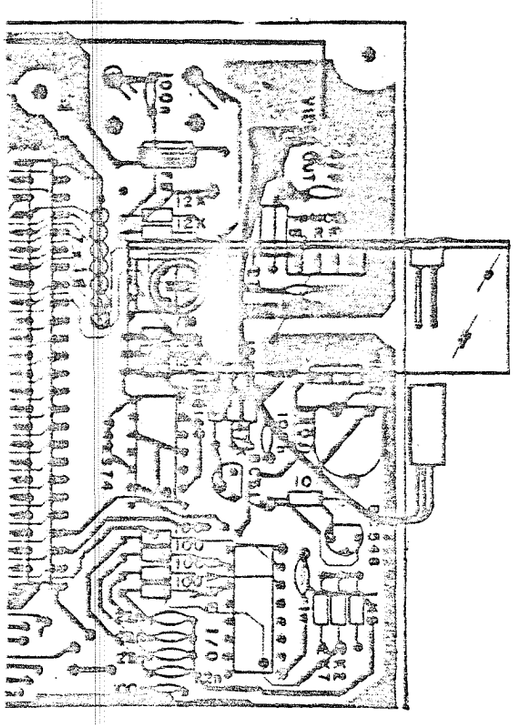
3- repare que, no extremo direito do lado interno do painel traseiro do EXATO, há um plug, do qual sai um fio em cujo extremo existe um conector fêmea duplo. Se este conector estiver encaixado à placa da UCP (placa principal do EXATO), desconecte-o puxando-o cuidadosamente. Em seguida, encaixe-o no conector macho de dois pinos (saída) do MODULADOR de RF, observando a polaridade correta através do ponto marcado no conector e na placa (eles devem ter suas posições coincidentes);

4- verifique então que, no canto direito da placa principal do EXATO, bem como num dos lados do MODULADOR de RF existe um conector de 4 pinos (o fêmea no MODULADOR e o macho na placa principal).

ATENÇÃO

Observe bem a polaridade desses conectores. Da mesma forma que no ítem anterior, as pintas devem estar localizadas no mesmo lado. Neste caso porém, a troca de polaridade poderá danificar seu MODULADOR bem como o próprio computador.

Encaixe então o MODULADOR de RF à placa principal. A figura abaixo ilustra a montagem.



5- Feche a tampa do computador.

Seu EXATO está agora pronto para operar, utilizando como saída um televisor. O televisor deverá estar sintonizado no canal 3. Caso os caracteres emitidos pelo computador não estejam sendo apresentados na tela, procure ajustar o controle de sintonia fina da TV.

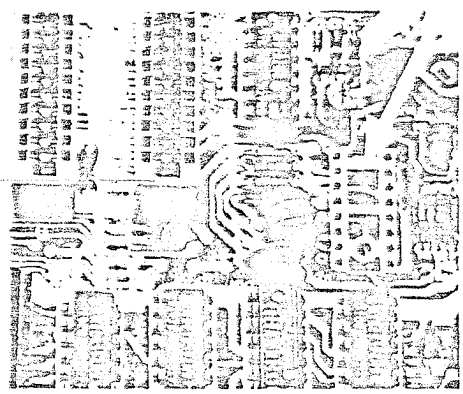
Observação: se o contraste da tela não estiver bem regulado (muito intenso, ou muito fraco), procure ajustar o trimpot que se encontra fora da blindagem (caixa metálica) do MODULADOR de RF (normalmente ele já está ajustado).

CONECTANDO A TELEVISÃO

Para conectar seu TV ao EXATO, ligue o cabo da antena na tomada marcada VIDEO OUT (na parte traseira do EXATO).

Caso você tenha um monitor, deverá retirar o modulador de RF (se ele estiver instalado). Para isso, desligue o micro e abra a tampa, puxando-a num movimento vertical para cima a parte traseira da tampa abrindo-o. Desconecte o cabo que faz a conexão da saída de vídeo com o modulador (no canto superior direito) e retire o modulador. Insira o conector (de 2 pinos) do cabo ao da placa principal localizado no canto direito traseiro, observando a polaridade através do ponto marcado no conector e na placa.

LIGANDO OS CONTROLADORES DE JOGO



Com a tampa aberta, encaixe o delicado plug do controlador na tomada GAME I/O, localizado no canto traseiro direito (vista frontal) da placa do EXATO. Seja cuidadoso e verifique se todos os pinos se encaixaram no conector. O onto branco do plug (pino 1) deverá ficar virado para a frente (lado do teclado do computador), coincidindo com o ponto branco da placa, conforme a figura ao lado.

O DRIVE

Se você tem uma unidade de disco, desembale-a cuidadosamente. Leia, então, o início e até o fim do primeiro capítulo do Manual de Sistema de Operação de Discos (geralmente chamado CDOS) que acompanha o pacote do drive. Esta leitura lhe fornecerá instruções completas de como instalar seu drive.

O GRAVADOR

Se você não estiver usando uma unidade de disco ou se você estiver usando os dois, o gravador e o disco, use o cabo fornecido (aquele com um plug em cada extremidade) para ligar o EXATO no seu gravador. Ligue um plug à tomada MIC ou MICROPHONE do gravador e o outro plug (da outra extremidade do cabo) à tomada situada na parte posterior do computador e marcada CASSETE OUT, para gravar seus programas.

Para carregá-los na memória do computador, ligue o plug do lado do gravador na tomada EAR ,EARPHONE ou MONITOR do gravador (marcas diferentes usam palavras diferentes). e outro plug no computador na tomada marcada CASSETE IN.

"OUT" significa "para fora do computador" e
"IN" significa "para dentro do computador".

Tudo o que resta é ligar o fio elétrico do gravador a uma tomada, e ele estará pronto para ser usado.

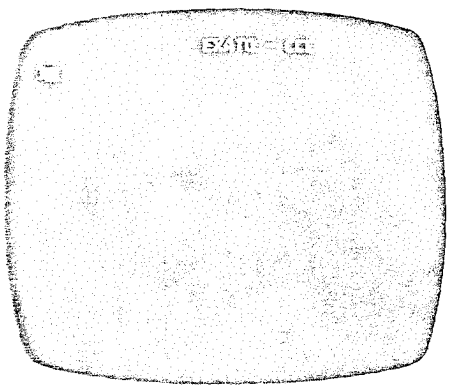
O SELETOR DE VOLTAGEM

Ao lado da chave liga-desliga na parte traseira do micro, está o seletor de voltagem, verifique se ele está na posição correspondente à voltagem da sua rede elétrica (110 ou 220V).

LIGANDO O COMPUTADOR

A primeira coisa a fazer já que todos os acessórios estão devidamente conectados e já foi feita a verificação do seletor de voltagem, é ligar o computador. O interruptor está na parte traseira do computador próximo ao cabo de força. Note que o "plug" de força do computador é diferente dos utilizados em outros aparelhos elétricos. Ele possui dois pinos chatos e um redondo. O pino redondo corresponde a ligação a "Terra". É importante que você ligue seu computador somente a tomadas com "Terra".

Agora o EXATO está totalmente montado e o que você tem a fazer é continuar a leitura deste manual para começar a explorar o mundo maravilhoso dos computadores pessoais.



Ligue o interruptor, a luz na parte inferior direita do teclado acenderá.

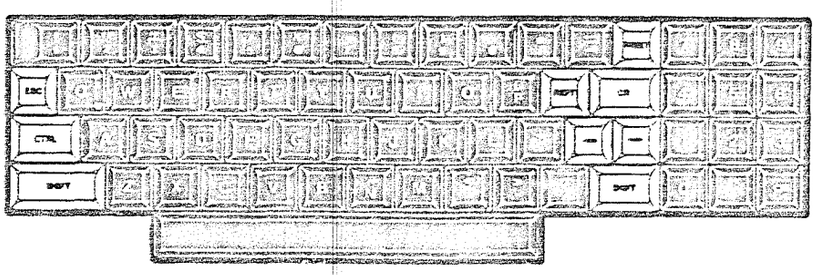
Também deverá aparecer no alto da tela a palavra "EXATO CCE", juntamente com um colchete (sinal de solicitação de entrada) e um quadrado luminoso piscando chamado "cursor". (Conforme a figura ao lado).

Se o seu computador EXATO não estiver respondendo corretamente às suas instruções (a medida que for se familiarizando com este manual, você descobrirá quais são as respostas certas) geralmente o ato de apertar a tecla RESET, resolverá seu problema. Se não funcionar, então provavelmente você conseguirá o resultado desejado, desligando o computador e ligando-o novamente.

Se você tiver um drive, ligado ao seu EXATO você terá os seguintes resultados: Você ouvirá alguns ruídos característicos e acenderá a luz vermelha "EM USO". O disco girará tanto tempo que parecerá que ele nunca mais vai parar. Na realidade, ele só vai parar quando você apertar a tecla RESET. Experimente fazê-lo. Desaparece o título "EXATO CCE", o colchete e o cursor aparecem no canto inferior esquerdo da tela.

O TECLADO

Se você estiver familiarizado com máquinas de escrever convencionais, encontrará poucas diferenças entre o teclado do EXATO e o da máquina de escrever. Em primeiro lugar, não há letras minúsculas. No EXATO pode-se obter apenas letras maiúsculas que são as necessárias para a programação em CCE BASIC. As letras minúsculas serão usadas para processamentos de textos e acessadas por Software.



Usando o desenho, localize as duas teclas SHIFT no teclado.

A razão da existência das teclas SHIFT é a de permitir quase o dobro do número de caracteres, com o mesmo número de teclas. Um teclado com uma tecla separada para cada caráter, seria muito grande, dificultando a procura das teclas desejadas.

Se você pressionar uma tecla que contém dois símbolos, o inferior aparecerá na tela. Se você apertar a mesma tecla, enquanto aperta qualquer uma das duas teclas SHIFT, aparecerá na tela o superior. Você descobrirá que a vírgula e o ponto junto com o SHIFT são respectivamente os sinais de maior (>) e menor (<). Você também descobrirá no teclado de seu EXATO outros símbolos que não aparecem numa máquina de escrever convencional, para se acostumar vá experimentando usar essas teclas.

Se na tecla houver só um símbolo, então apertar SHIFT junto com esta tecla será o mesmo de se digitar apenas a tecla. Há exceção: o M . Ao se pressionar SHIFT e a tecla M , aparece o colchete direito (sinal de solicitação de entrada).

Uma diferença importante na utilização do teclado EXATO e o da maioria das máquinas de escrever é que você não pode usar um L minúsculo como o número "1". Algumas datilógrafas terão que perder o hábito de usar a tecla L quando querem digitar o número "1".

Quando os matemáticos hindus inventaram o círculo aberto para o numeral zero, eles não usavam o alfabeto romano. Assim, eles escolheram um símbolo que não interferia com o seu alfabeto, mas que se parece com nossa letra "O". O computador precisa manter uma diferença entre os "zeros" e as letras "O". O método usual para esta diferença, utilizado pelo EXATO e muitos outros computadores, é cortar o zero com uma barra diagonal. Agora você é capaz de diferenciá-los. O teclado e a tela da TV tornam evidente a diferença. Tente teclar um "O" e um "zero" e veja a diferença.

Depois de digitar um pouco, a tela tende a ficar congestionada. Para limpá-la, você precisará usar a tecla marcada [ESC]. ESC é a abreviatura da palavra "ESCAPE", e ela não é nenhum caráter que aparece na tela do EXATO. Note que a tecla [ESC], ao contrário da tecla [SHIFT] não precisa estar pressionada enquanto se digita outra tecla. Você tem que operar três teclas para limpar a tela. Primeiro pressione [ESC] e solte. Depois, enquanto aperta [SHIFT] , pressione [F] . Prontamente desaparece o que estava escrito na tela.

NOTAÇÃO DO TECLADO

Aqui introduziremos uma notação simples. Como você já viu, como se digitar uma tecla, por exemplo a da letra [H]. Para indicar a digitação sucessiva de várias teclas, simplesmente listaremos as teclas na ordem a serem pressionadas:

[E] [X] [A] [T] [O]

As vezes, você terá que segurar uma tecla, enquanto pressiona outra. Por exemplo, para obter o DOLAR (<$>), você precisa segurar a tecla [SHIFT], enquanto pressiona a tecla [4]. Sempre que for necessária esta ação dupla, mostraremos os símbolos para ambas as teclas, uma sobre a outra:

[SHIFT]

[$ / 4]

A tecla superior deverá ser digitada ao mesmo tempo que a tecla inferior.

Usando a nova notação, veja como limpar a tela:

[ESC]

[SHIFT] [@ P]

Experimente.

CARACTERES DE CONTROLE

Quando você pressiona a tecla [5], o número 5 aparece no vídeo, provavelmente você acredita que isto seja verdade, mas tente assim mesmo. Se você segurar a tecla [SHIFT] enquanto digita a tecla [5], deverá aparecer na tela o sinal de porcentagem (%). Será que aparece ? A tecla [SHIFT] permite que algumas teclas tenham duas funções diferentes. Várias delas também apresentam uma terceira função, que é obtida pressionando-se a tecla [CTRL] enquanto se digita as outras teclas. [CTRL] significa "controle". Em vez de colocar novos caracteres na tela, ao usar a tecla [CTRL], o computador responde realizando certas ações. Os caracteres de controle nunca aparecem na tela.

Aperte a tecla [CTRL] e simultaneamente pressione [G]

[CTRL]

[G]

e tocará um BIP! Sempre que o computador quiser chamar sua atenção para alguma coisa, ele emitirá um "BIP". [CTRL] [G] é chamado "Bell", por razões históricas: o teclado atual é baseado no do teletipo; nessa máquina digitando-se [CTRL] [G], toca um sino de verdade.

Outra tecla que não é usualmente encontrada em máquinas de escrever é a [REPT] que significa "repita", pressionando a tecla [REPT] enquanto você digita qualquer outra tecla faz com que o caráter que você digitou apareça repetidamente na tela. Você precisa primeiro apertar e segurar a tecla do caráter que você quer ver repetido, e então apertar a tecla [REPT]. Experimente!

Há também no teclado uma tecla marcada [CR]. Nas máquinas antigas significava "retorno do carro". No EXATO, faz com que o cursor "volte" ao canto esquerdo da tela, mas também é uma mensagem especial para o computador. Adiante falaremos mais a respeito desta mensagem. Quando você apertar [CR], poderá tocar um "BIP" e a mensagem:

? SINTAX ERRO

aparecerá na tela, por enquanto ignore esta mensagem.

As únicas teclas não mencionadas são as teclas [<-] e [->]. Elas movem o cursor para a esquerda e para a direita. Mais tarde elas serão detalhadamente explicadas. Experimente estas teclas e quaisquer outras que possa encontrar.

LIGANDO O GRAVADOR

Se você não estiver usando um gravador com fitas cassete, pule até a seção "USANDO UM DRIVE".

O colchete direito e o cursor que aparecem no canto esquerdo da tela, fazem com que você saiba que está em "CCE BASIC". Agora você está pronto par ajustar o controle de volume no gravador.

Ao usar um gravador, a intenção é a reproduçã de sons que você possa escutar. Se forem muito baixos, você perderá algumas palavras da música, se muito altos, incomodarão.

O ato de tocar uma fita cassete para o EXATO é feito com o intuito de colocar as informações contidas na fita, na memória do computador. Se o volume ajustado estiver muito baixo, o computador perderá algumas das informações e reclamará, enviando uma mensagem de erro. Se o volume ajustado, estiver muito alto, o computador também reclamará.

Para se encontrar o ajuste ideal de volume, use o método de tentativa e erro. Toque a fita baixinho para o computador e verifique se ele recebeu bem as informações, se não funcionar, aumente um pouco o volume. Quando o volume estiver no nível certo, o computador o avisará emitindo um "bip".

Para limpar a tela, tecle:

[ESC] [SHIFT]

[F]

Coloque no gravador uma fita de programa. Para cada posição de controle de volume, faça o seguinte:

1 - rebobine a fita para o princípio;
2 - comece a tocar a fita;
3 - digite: LOAD e tecle [CR].

Quando fizer isso, o cursor desaparecerá. Poderá levar até 15 segundos para que alguma coisa aconteça.

Existem as seguintes possibilidades:

a. aparece a mensagem: ? SINTAX ERRO;
b. não acontece nada;
c. aparece a mensagem: ERRO (com ou sem um sinal sonoro);
d. o computador emite um "BIP" e nada aparece.

Na alternativa A , não reajuste o controle de volume, mas volte a etapa 1, onde você rebobina a fita.

Nas alternativas B e C , assegure-se de que esperou 15 segundos antes de desistir. Se não aparecer prontamente um caráter ou cursor e o EXATO não responder a seu teclado, pressione [RESET], aumente um pouco o volume e volte à etapa 1.

Raramente pode acontecer do comando LOAD não funcionar adequadamente, e então o cursor aparecerá imediatamente na tela não esperando que a fita seja carregada na memória. Se isso ocorrer, desligue seu computador e ligue-o novamente usando o interruptor de força na parte traseira, e tente novamente a operação de LOAD.

Na alternativa D, você está na pista certa, ao escutar o "BIP" espere mais 15 segundos. Este tempo irá variar de acordo com o tamanho do programa. Podem acontecer duas coisas, ou você obterá uma mensagem de erro (alternativa C) ou reaparecerá o sinal de solicitação de entrada e o cursor. Se eles aparecerem, pare e rebobine a fita. Marque a posição do controle de volume do gravador, de modo a usar o mesmo volume sempre que for carregar esta fita para memória.

Então digite:

RUN e tecle [CR]

PROGRAMAS EM FITA

Uma vez que tenha sido ajustado corretamente o controle de volume

1 - rebobine a fita;
2 - comece a tocar a fita;
3 - digite LOAD.

Depois que digitar [CR] , o cursor desaparecerá. Nada acontecerá de 5 a 20 segundos, e então o computador emitirá um "BIP".

Isso significa que as informações da fita começaram a entrar na memória. Depois de algum tempo, o que vai depender da quantidade de informação contida na fita, (mas geralmente menos do que alguns minutos), o computador emitirá outro "BIP" e o caráter de solicitação de entrada e o cursor reaparecerão.

4 - Pare o gravador e rebobine a fita. As informações foram transferidas e, por enquanto, você não precisará do gravador.

5 - Digite RUN e tecle [CR] e seu programa começará a ser executado.

Se seu EXATO estiver na linguagem CCE BASIC, a fita que você carregar também deverá estar em CCE BASIC. Tentar carregar uma fita em linguagem diferente dará resultados imprevisíveis. Poderá aparecer na sua tela estranhas mensagens de erro e caracteres estranhos, você poderá perder o controle do teclado, ou ainda poderá ocorrer uma série de fatos estranhos. Se isso acontecer com você, desligue o computador e ligue-o novamente, de modo a poder voltar ao normal.

Usuários de computador usam termos diferentes para descrever o processo de obter informações de uma fita e colocá-las em memória. Diz-se que o computador "lê" a fita. Diz-se que a informação "entrou" ou "foi lida" pelo computador. O ato de ler uma fita, também é chamado de "carregar" uma fita no computador e diz-se que a informação foi "carregada" no computador. Todas essas expressões são formas diferentes de exprimir a mesma coisa.

UMA SUGESTÃO ÚTIL

O que é que o computador acha de tão interessante a respeito desta fita: escute uma delas. Não soará como música a seus ouvidos, mesmo assim você poderá reconhecer alguns dos sons que o computador procura. A informação começa com um tom firme, depois a um rápido "BIP" seguido do mesmo tom. Este é um tom na frequência de 1000 ciclos por segundo. Este tom fica a duas oitavas abaixo do "dó" médio na escala musical. Depois deste tom, vem uma sequência sonora que lembra uma tempestade. Quando você se acostumar ao som de uma boa fita, poderá verificar rapidamente, de ouvido se a fita é de computador ou não. Se você puder interpretar o que contém a fita, simplesmente ouvindo-a, então você é um mutante e irá longe no mundo dos computadores.

USANDO UM DRIVE

Pule esta parte se não estiver usando Disquetes.

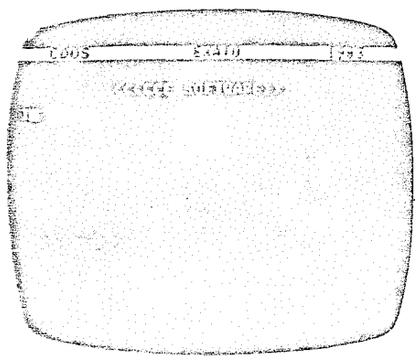
O DRIVE é muito mais rápido e fácil de ser usado do que um gravador, entretanto disquetes e drives são objetos delicados e alguns cuidados tem que ser tomados para protegê-los. No seu manual CDOS você encontrará informações sobre os cuidados a serem tomados em relação a disquetes e acionadores de discos (drives). Leia cuidadosamente caso não o tenha feito.

A última parte do primeiro capítulo do manual de CDOS é chamada: INSERÇÃO E REMOÇÃO DE DISQUETES. Tire o disquete SISTEMA MESTRE de sua embalagem e coloque-o com a etiqueta voltada para cima e a fenda oval virada para a parte posterior do drive, assim como está descrito no manual CDOS.

Uma das características que tornam o disquete tão fácil de ser usado, é sua capacidade de armazenar e recuperar vários grupos diferentes de informações. Os grupos de informações são armazenados no disco com seus respectivos nomes chamados de nomes de arquivo. Assim por exemplo, um programa que armazena endereços no disquete poderá ser chamado ENDERECOS.

Os programas de controle de arquivos que guardam, recuperam e realizam uma série de outras tarefas de manutenção, constituem o CDOS (Sistema Operacional de Disco CCE). O processo de somar as habilidade do CDOS ao CCE BASIC (ou a qualquer outra linguagem usada por seu EXATO) é chamada "carregar CDOS" ou "inicializar o sistema CDOS". (Em inglês: booting dos).

Há várias maneiras de carregar o CDOS, uma delas é simplesmente desligar seu EXATO e ligá-lo novamente. A luz vermelha do drive acenderá e ele fará o mesmo som de giro que fez quando o computador foi ligado a primeira vez. Desta vez o disquete parará de girar sozinho. Quando o giro cessar e a luz vermelha apagar, o título EXATO CCE sumirá e uma mensagem aparecerá na tela.



Quando obtiver esta mensagem, você saberá que o CDOS está carregado.

O caráter de solicitação de entrada e o cursor que aparecem no canto inferior esquerdo da tela indicam que o EXATO está em liguagem CCE BASIC e está pronto para receber instrucões.

A outra maneira de inicializar o CDOS é digitando:

SHIFT

P R # 3 & 6 CR

No seu EXATO, se o cartão de controle não estiver ligado no slot nº 6, digite:

SHIFT

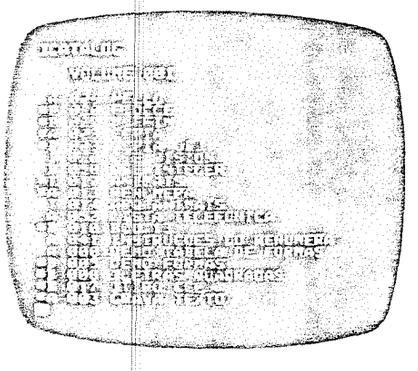
P R # 3 "nº do slot" CR

e o CDOS será carregado.

O "Sistema Mestre" é um disquete muito especial. Contém programas que você necessitará para seu aprendizado além de outros programas de grande utilidade. Para saber quais os programas que estão no disco, digite:

CATALOG e CR

e uma lista dos arquivos aparecerá na tela conforme a foto abaixo:



Para executar um destes programas digite :

RUN "NOME DO PROGRAMA" [CR]

e o programa será executado. Por exemplo, localize no catálogo o programa demonstração de cores e digite:

RUN DEMONSTRACAO DE CORES

depois pressione [CR] e aparecerá o menú do programa.

O "MENU"

Para maior facilidade em geral, são feitas listas de opções na execução de um programa. Os usuários de computador chamam estas listas de opções de "menu". Ela funciona como um cardápio de restaurante.

Tente, da mesma forma, escolher uma das demonstrações a cores, digitando seu número (seguido de [CR]).

Quando estiver vendo uma das demonstrações, simplesmente tecla [CR] e volte ao "menu".

PARANDO A EXECUÇÃO

Para parar a execução use:

CTRL

C

Isso fará com que apareçam o sinal de solicitação de entrada e o cursor.

O sinal de solicitação de entrada lhe dirá que está tudo pronto para se digitar informações para o computador. É, por isso, que é chamado de solicitação de entrada.

Uma vez que a execução parou, ela pode ser iniciada novamente digitando-se RUN e, é claro, seguido do CR, você não será mais lembrado disso, pois supomos que já tenha aprendido.

Use:

CTRL

C

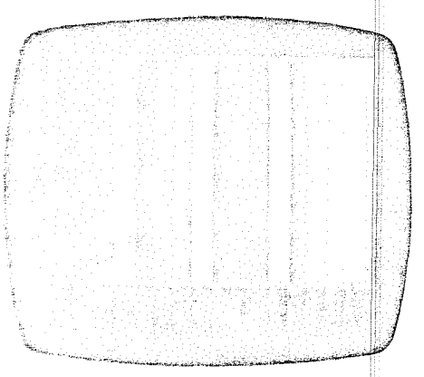
para parar a execução, e

RUN

para fazê-la iniciar novamente. Tente repetir isso algumas vezes.

AJUSTANDO A TV A CORES

Se o "menu" não está na sua tela, carregue o CDOS e execute o programa "TESTE CORES". Um dos itens do menu é chamado NOME DAS CORES. O usaremos para ajustar as cores da TV. Digite o número 1 (NOME DAS CORES), e tecle CR. Aparecerá uma série de barras coloridas, sob cada uma destas barras há uma abreviação de quatro letras que corresponde ao nome da cor. Os nomes completos são:



Ø - PRETo
1 - MAGEnta (vermelho azulado)
2 - AZUL escuro (1)
3 - VIOLeta
4 - VERde escuro (1)
5 - CINZa
6 - AZUL médio (2)
7 - AZUL claro (3)
8 - MARRom
9 - LARAnja
1Ø - CINZa
11 - ROSA
12 - VERDe
13 - AMARelo
14 - AGUA marinha
15 - BRANco

Se você tiver uma TV ou monitor branco e preto, juste o brilho e contraste. É claro que, se a imagem está tremendo, pare-a da maneira como o faria com qualquer TV. Se a sua TV for colorida, o trabalho é um pouco maior para o ajuste.

Lembre-se que esta questão de cores é muito subjetiva e que você pode fazer o que quiser com elas. As instruções a seguir, lhe darão o quadro que preferimos, usando as cores padrão. Mas você deve agradar a seus próprios olhos, e além disso, os ajustes variarão com diferentes quantidades de luz na sala. Desligue qualquer ajuste automático de cores. Em alguns aparelhos está marcado com "Auto Color" ou simplesmente "Auto". Abaixe totalmente o volume de som do aparelho (mas não o deslique). Agora são importantes 4 controles: Imagem, Brilho, Cor e Matriz ou Tint. Alguns aparelhos tem um botão marcado "contraste" ao invés de "Imagem" - mas o botão realiza a mesma tarefa. Ajuste o botão de controle de imagem até sua posição mais turva e então o botão de brilho até que o fundo fique totalmente negro. Coloque o controle de cor até a metade de seu curso. Agora ajuste o contraste para torná-las mais brilhantes. Não as torne tão brilhantes de modo que elas cheguem a se embaralhar.

Agora ajuste o botão de cores. Girando para um lado a TV ficará branco e preto. Este ajuste é útil quando aparece apenas texto na tela. Para o outro lado, você poderá ajustar as cores a seu gosto. Finalmente, ajuste o botão de matriz até que todas as cores concordem com seus nomes. O violeta, rosa e o amarelo são indicadores especialmente sensíveis. Assegure-se também de que os três tons de azul estejam distintos entre si.

Quando as cores da TV estiverem OK, tecle [CR] e o "Menu" reaparecerá. Agora teste o ítem 2, que mostra as listas coloridas com seus números de código. Tente também as outras demonstrações.

O PROGRAMA GOBBLER

Para executar o programa GOBBLER você deve possuir um drive e um disquete Sistema Mestre, o qual tem este programa gravado.

Coloque o disco no drive e ligue o EXATO, assim que os CDOS estiver carregado, o cursor aparecerá na tela. Então digite CATALOG. Serão listados na tela os nomes de todos os programas gravados neste disco. Você perceberá que há uma letra B antes do nome deste programa, isto quer dizer que ele foi gravado em linguagem de máquina (Sistema Binário). Para executar este programa digite:

BRUN GOBBLER

e ele será carregado e iniciará a execução automaticamente. Aparecerá desenhado um labirinto na tela e os bichinhos ganharão movimento, mas só haverá contagem de pontos após se pressionar a barra de espaço, feito isto você poderá controlar os movimentos do "come-come" usando as teclas [<-] (para esquerda), [->] (para direita), [A] (para cima) e [Z] (para baixo).

O objetivo deste jogo é fazer com que o "come-come" coma todos os quadrinhos da tela sem que os fantasmas o peguem. Ao comer os quadrinhos que ficam piscando nas pontas da tela, os fantasmas ficaram de olhos fechados e o come-come poderá comê-los isto lhe dará mais pontos.

A TECLA [RESET]

Esta tecla permite salvar o sistema em determinadas situações, porém, se pressionada inadvertidamente, pode trazer problemas desagradáveis.

Para que tal acidente possa ser evitado, abra a tampa do EXATO e note que, logo abaixo do teclado existem três pinos e um Jumper entre o primeiro e o segundo pino. Se você puser este Jumper entre o segundo e o terceiro pino, [RESET] só funcionará se a tecla [CTRL] estiver pressionada. Não execute esta operação se o seu EXATO estiver utilizando o CP/M.

CAPÍTULO 2

CONHECENDO CCE BASIC

CONHECENDO O CCE BASIC

Se você estiver em CCE BASIC, o caráter de solicitação de entrada seguido do cursor, aparecerão no canto esquerdo da tela cada vez que você teclar [CR].

Fique em CCE BASIC e se tiver um drive, carregue o CDOS.

UMA OLHADA NA INSTRUÇÃO PRINT

Agora que você já tem o caráter de solicitação de entrada e o cursor na tela (e se você tem um drive o CDOS foi carregado), você está pronto para usar a linguagem CCE BASIC então digite:

```
PRINT "OLA"
```

e aparecerá na linha seguinte a palavra OLA.

Se a palavra não aparecer na tela, pergunte-se, será que esqueci o [CR]? Se você digitar errado a palavra PRINT, receberá a seguinte mensagem de erro:

```
? SINTAX ERRO
```

Se você esquecer uma ou ambas as aspas aparecerá um 0 (zero) na linha seguinte.

Se a aspa final foi o último caráter antes do [CR], você não tem que digitá-la: a execução da instrução será a mesma com ela ou sem. Mesmo assim, é uma boa idéia sempre colocar as aspas finais. O hábito de colocá-las será importante mais adiante e além disso este manual supõe que você as use. O comando

```
PRINT "OLA"
```

é uma instrução ao computador, dizendo-lhe para mostrar na tela todos os caracteres entre aspas, nesse caso uma palavra de saudação. Você pode usar o comando PRINT para dizer ao computador que exiba qualquer mensagem que você desejar. Entretanto, se você digitar mais que 240 (duzentos e quarenta) caracteres, o computador começará a emitir um BIP , fará aparecer uma barra diagonal no fim da linha, você terá que digitar a instrução novamente na linha seguinte.

Agora tente o comando:

PRINT "150"

obedientemente o computador exibirá o número 150 na linha seguinte. Mas digite:

PRINT 150

e, novamente o computador exibirá o número sem qualquer confusão ou mensagem de erro por causa da falta das aspas. Na realidade o EXATO lhe permite exibir qualquer número sem colocá-lo entre aspas.

Sem mais estudos o EXATO pode ser usado como uma calculadora. Digite o seguinte:

PRINT 3+4

aparecerá a resposta 7 na linha seguinte.

O EXATO pode realizar 5 operações matemáticas elementares:

1- ADIÇÃO
Indicada pelo sinal usual de soma (+).

2- SUBTRAÇÃO
Usa o sinal negativo convencional (-).

3- MULTIPLICAÇÃO
É usado um asterisco (*). Para descobrir quanto é 7 vezes 8 (no caso de ter esquecido), digite PRINT 7*8 , e sua memória se refrescará.

4- DIVISÃO
Como de costume use a barra inclinada (/). Para dividir 63 por 7, digite PRINT 63/7 , e aparecerá a resposta. Tente dividir 3 por 2. A resposta é um e meio e o EXATO lhe dará a resposta 1.5 .

5- POTENCIAÇÃO
O símbolo usado é (^), isto é SHIFT N
Frequentemente é necessário multiplicar um número por si mesmo um dado número de vezes. Ao invés de escrever:

PRINT 4*4*4*4*4

você pode digitar:

PRINT 4^5

este sinal não tem nada de especial é apenas a maneira de

se dizer quatro elevado a quinta potência e o computador entender.

FORMATO NUMERICO NO CCE BASIC

Digita:

```
PRINT 45.340
```

na linha seguinte aparecerá 45.34 e o zero será suprimido. O EXATO não exibe os zeros iniciais nem os finais após o ponto decimal.

Números muito pequenos (entre Ø.ØØØØØØ ØØØØØØØØØØØØØØØØØØ ØØØØØØØØØØØØ3 e -Ø.ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ3) serão convertidos em zero pelo computador. Um método mais fácil de escrever estes números é: 3*1Ø^-39 e -3*1Ø^-39. não acredite nisso experimente você mesmo.

Agora digite:

```
PRINT 985788.6898
```

Surpresa! Os dois últimos dígitos se perdem e o número que resta é a melhor aproximação que o computador acha. Esse processo é conhecido por "arredondamento". Experimente digitar:

```
PRINT 788.6898
```

ele não arredondou o número, mas exibiu-o exatamente do modo que você escreveu. Loucura ? Sim, mas há um método nesta loucura aparente: os números só serão arredondados se tiverem mais de nove dígitos. Qualquer número com menos de dez dígitos não será arredondado. O computador faz o melhor possível, mas só dispõe de nove dígitos para trabalhar.

Se você digitar o comando PRINT com um número comprido como por exemplo: 1234567890.Ø, o EXATO exibe 1.23456789 E+Ø9.

Os números 1234567890 e 1.23456789 E+Ø9 têm o mesmo valor. O número exibido pelo computador é uma "notação científica" se você precisar de números como este, provavelmente saberá como interpretá-los. Se estiver curioso a respeito destas notações o manual REFERÊNCIA PARA PROGRAMAÇÃO DO CCE BASIC traz mais informações. Experimente mais alguns números.

Quantos dígitos pode ter um número sem ponto decimal, antes que o EXATO o transforme numa notação científica ?

Não se preocupe se esta notação parece complicada. Provavelmente você não a utilizará neste momento. Lembre-se que qualquer número será exibido exatamente como digitá-lo. Se o fizer entre aspas, entretanto, o EXATO não poderá usá-lo para operações matemáticas.

MAIS A RESPEITO DO [CR]

Até agora, você teclou [CR] como um autômato, após cada linha. Achamos que podemos dizer-lhe porque esta tecla é tão usada. A razão é simples, sem o [CR] , o computador não fica sabendo que você acabou de dar a instrução.

Por exemplo, você pode começar a digitar PRINT 4+5, se o computador se adiantasse e exibisse o número 9, você poderia se chatear porque tinha planejado digitar PRINT 4+5+346, o que lhe daria uma resposta totalmente diferente. Já que o computador não pode adivinhar quando você acabou de digitar uma instrução, você precisa dizer-lhe, e o faz pressionando [CR] .Lembre-se que após uma instrução, deve-se sempre digitar [CR] . Se você fez todos os exemplos, e teclou [CR] após cada instrução isso já deve ter se tornado um hábito.

Esperamos sinceramente que você tenha testado todos os exemplos pois aprender a programar é muito parecido com aprender a andar de bicicleta, tocar piano ou jogar futebol. Você pode ler todos os livros do mundo que falam sobre andar de bicicleta e ser um perito teórico, mas todo este conhecimento será de pouca valia quando montar numa bicicleta pela primeira vez. Quando você tiver aprendido a pedalar através da prática, poderá ir onde quiser.

O mesmo pode ser extrapolado para a programação. Você pode ler este manual e achar que entendeu tudo, mas não será capaz de programar. A verdade é que vai aprender a programar se fizer cada exemplo, do modo em que é apresentado.

CARACTERÍSTICAS DE EDIÇÃO

Ninguém é digitador perfeito. Todos fazemos erros e o EXATO tem várias características que ajudam na correção de erros e portanto economizando o esforço de reescrever toda a linha. São elas que entram em ação as teclas marcadas com flechas para a direita e para a esquerda.

A tecla [<-] funciona como uma tecla de retrocesso na máquina de escrever comum e por isso damos o mesmo nome a ela. Alguns testes tornam obvio seu funcionamento, digite:

PRINT COMPUTADO"

o computador exibirá Ø (zero), por causa da falta das aspas.

Agora se tivesse digitado:

PRINT "COMPUTADO"

o computador teria exibido: COMPUTADO.

Não acredite neste manual, experimente. Agora sem teclar [CR], digite a instrução errada.

PRINT "COMPATADO"

já que você não teclou [CR], ainda não aconteceu nada. o cursor está parado ao lado das últimas aspas.

Para trocar COMPATADO por COMPUTADO, podemos usar a [<-]. Note que cada vez que você digita esta tecla, o cursor move-se para a esquerda um espaco. A palavra retrocesso é um verbo, portanto retroceda o cursor até a letra A. Digite U. Como vê, o U substitue A. Agora tecle [CR] e você obteve:

COMPU

da palavra COMPUTADO . Isto aconteceu porque você retrocedeu sobre "TADO". Qualquer caráter na linha em que você estiver digitando e que for passada sobre ele a tecla de retrocesso, não é mandado para o computador quando você teclar [CR].

Uma solução seria corrigir o A, retrocedendo até ele e então digitar o resto da linha novamente. Experimente. Funciona ! Há entretanto uma maneira mais fácil. Ao teclar [->],o cursor se move para a direita. Ao mover-se sobre um caráter, surte o mesmo efeito de se digitá-lo novamente. Chamamos esta tecla de "Avanço". Digite

PRINT "COMPATADO"

Então retroceda até o A e o mude para U. Para completar, simplesmente pressione a tecla de avanço 5 (cinco) vezes e então tecle [CR] . Funcionou ? O uso da tecla retrocesso e da tecla avanco irá economizar-lhe muito tempo.

Experimente usá-las várias vezes em seus próprios "erros" de modo que o uso das mesmas se torne familiar.

COLOCANDO CORES NA TELA

Para colocar gráficos coloridos na tela precisamos de um modo de descrever quais das 16 (dezesseis) cores disponíveis e onde as desejamos. Para especificar onde vai determinada cor, dividimos a tela em 40 colunas verticais, numeradas de 0 a 39. A coluna 0 (zero) fica na extremidade esquerda da tela e a 39ª na extremidade direita.

Você pode ficar imaginando porque os números não vão de 1 a 40, mas de 0 a 39. Ao ganhar experiência em programação, descobrirá que a escolha que fizemos é a mais prática.

A tela também é dividida em 40 linhas horizontais, novamente numeradas de 0 a 39. As linhas horizontais começam com a linha 0 (zero) no alto da tela e vão aumentando até a linha 39 na base da mesma. Essas linhas interceptam as colunas, dividindo cada coluna em 40 "blocos" numerados de 0 (quadradinho superior) a 39 (o mais inferior).

Aqueles que apreciam a terminologia formal, reconheceram que este é simplesmente um sistema de Coordenadas Cartesianas Retangulares. Os que não gostam de termos técnicos podem pensar em termos de colunas de "tijolinhos".

Para usar cores na tela, digite a seguinte instrução:

```
GR
```

sem dúvida você se lembrou do [CR].

Quando você usa este comando a tela fica limpa restando apenas na base, quatro linhas para o texto. O "GR" é a abreviação de "Gráfico". Para reverter a situação anterior (antes de ter digitado GR), use o comando:

```
TEXT
```

Quando você digitar este comando, a tela subitamente se transforma numa porção de caracteres "■" . Isso é normal. Tente digitar a instrução TEXT e então, voltar ao gráfico, digitando a instrução GR.

Antes de poder colocar um ponto de cor na tela, você precisa dizer ao computador qual a cor desejada. Há 16 cores disponíveis. Você já viu anteriormente: são numeradas de 0 a 15, como foi mostrado no Programa TESTE CORES (2 - NUMERO DAS CORES).

Suponhamos que você queira um ponto verde em algum lugar. Primeiro você precisa digitar a ordem GR e então digite:

```
COLOR = 12
```

Isso significa que qualquer ponto (bloco gráfico ou segmentado) de cor que você colocar, será verde. Na realidade, a não ser que seja dado um outro comando, tudo o que for exibido na tela será verde, exceto é claro a pequena área reservada com o texto para instruções na base da tela. Para colocar um ponto colorido no canto superior esquerdo da tela (na coluna 0 da linha 0) digite:

```
PLOT 0,0
```

Para colocar um ponto da mesma cor no canto superior direito, você tem que especificar coluna 39, linha 0. Portanto digite:

```
PLOT 39,0
```

Note que sempre se fornece primeiro o número da coluna. Agora coloque um bloco laranja no canto inferior esquerdo. Primeiro mude a cor. Lembre-se, você deve fazer estes exercícios e pensar neles. Portanto, mãos a obra e digite:

```
COLOR = 9
```

nada acontece na tela (mesmo que tenha lembrado-se de teclar CR).

Mas o computador terá registrado, e quando você usar outro comando PLOT a resposta aparecerá em laranja e não em verde. Agora que você alterou a cor, você pode colocar um ponto no canto inferior esquerdo (Coluna 0, linha 39).

```
PLOT 0,39
```

Funcionou ? Você não esqueceu de teclar o CR? Laranja é sua cor predileta?

Agora coloque um ponto magenta no canto inferior direito. Tente sozinho.

MENSAGENS DE ERRO NOS GRÁFICOS

Quando estiver usando o comando PLOT, há duas mensagens de erro que podem aparecer. Você já sabe disso se digitou PLAT ou PLOP ao invés de PLOT e recebeu a mensagem.

```
?SINTAX ERRO
```

Outra mensagem de erro ocorre quando você assinala um número acima ou abaixo daqueles permitidos para coordenadas no comando PLOT. Digite:

```
PLOT 13, 85
```

e você terá a mensagem:

```
?VALOR ILEGAL ERRO
```

Essa mensagem significa que você tentou assinalar um ponto fora das dimensões da tela. Os números mais altos que você pode usar em um comando PLOT são: 39 para as colunas e 47 para as linhas. O uso de números acima de 39 para as linhas em um comando como este:

```
PLOT 20, 45
```

exibirá caracteres peculiares na área de texto, na base da tela.

Usando caracteres com valores negativos em um comando PLOT, também provocará o surgimento da mensagem:

```
?VALOR ILEGAL
```

DESENHANDO LINHAS

Suponhamos que você queira desenhar uma linha horizontal em azul claro que vá da coluna 5 a coluna 9, ao nivel da linha 14. Você poderia digitar:

```
COLOR = 7
PLOT 5,14
PLOT 6,14
PLOT 7,14
PLOT 8,14
PLOT 9,14
```

Note que não aparecem as junções entre os blocos ligados e eles formam uma linha contínua. Entretanto, há uma maneira mais simples de fazer linhas horizontais.

Suponhamos que você queira desenhar uma linha horizontal verde escuro no meio da tela. Usando o processo mais longo, seriam necessários 40 comandos:

```
COLOR = 4
PLOT Ø,2Ø
PLOT 1,2Ø
PLOT 2,2Ø e assim por diante até
PLOT 39,2Ø
```

O método mais fácil é digitar somente:

```
COLOR = 4
HLIN Ø, 39 AT 2Ø
```

Pressione [CR] e aí está: uma linha horizontal instantânea, da coluna Ø a coluna 39, ao nível da linha 2Ø.

Agora tente colocar uma linha magenta da coluna 19 até a 28, ao nível da linha 18. Tente algumas outras. Se fizer umas 6 linhas horizontais diferentes, você vai ficar craque.

Note que quando você coloca um ponto ou uma linha em um lugar onde já existe algo, a nova cor prevalece sobre a anterior. Para limpar a tela de todos os gráficos traçados, use o comando GR.

Há um dispositivo para linhas verticais automáticas semelhante ao existente para linhas horizontais. Para traçar uma linha vertical na coluna 7 da linha 12 à 33 digitamos:

```
COLOR = 9
VLIN 12, 33 AT 7
```

experimente este comando.

Faça outras tentativas mudando os números das linhas e colunas. Você poderá testar sua habilidade para traçar linhas verticais e horizontais traçando uma borda magenta em volta da tela com 5 comandos. Depois faça um "x" verde no centro da tela.

Treine o uso de PLOT, HLIN e VLIN durante algum tempo, antes de prosseguir.

OS CONTROLADORES DE JOGO

Antes de iniciarmos verifique se os controladores (PADDLE) ou o JOYSTICK estão corretamente instalados.

Agora digite:

```
PRINT PDL ( Ø )
```

e deverá aparecer um número, caso não apareça, movimente o outro controlador ou mude o sentido do movimento da alavanca do JOYSTICK.

Mexa um pouco o controlador e digite novamente

```
PRINT PDL ( Ø )
```

Experimente ficar mudando o controlador e digitando:

```
PRINT PDL ( Ø )
```

várias vezes. Quais são os números maiores e menores que você pode conseguir ? Qual é a menor mudanca que você pode fazer ?

Você pode descobrir a posição do outro controle digitando:

```
PRINT PDL ( 1 )
```

A abreviação "PDL" vem da palavra "PADDLE", que em inglês significa alavanca. Há muitos outros usos para estes controles.

PDL é uma função. Em linguagem CCE BASIC, é uma instrução que utiliza um ou mais números, executa algumas operações sobre estes números de modo a fornecer um valor único. Os números que a função usa são chamados de ARGUMENTOS e são sempre colocados entre parênteses após o nome da função. PDL é uma função que tem um único argumento. Costuma-se dizer que o número gerado por ela foi retornado ao programa.

VARIÁVEIS E CARACTERÍSTICAS DE CALCULADORA

Em muitas calculadoras simples você pode guardar um número para uso ou referência posterior. Para fazê-lo, você o coloca num local especial que chamaremos de compartimento. Geralmente isso é feito apertando a tecla [M] (memória). No EXATO você pode fazer o mesmo. Por exemplo, para guardar o valor 77, você tecla:

```
M = 77
```

o valor 77 não é exibido na tela, mas sim armazenado na variável chamada M. Se agora você digitar:

```
PRINT M
```

o computador exibirá o valor de M. Agora escreva:

```
M = 324
```

e faça com que o computador exiba o valor de M. O valor será 324. O que aconteceu com o 77 ? Desapareceu para sempre. As variáveis só podem armazenar um valor por vez. Quando você coloca um novo valor em M, o valor anterior é substituido.

Digite:

```
PRINT "M"
```

O que aconteceu ? Há uma grande diferença entre M e "M".

É como a diferença entre as seguintes frases:

```
Ratos têm 4 patas
"Ratos" têm 5 letras
```

No primeiro caso, nós nos referimos aos pequenos animais peludos e de longas caldas, no segundo, à palavra propriamente dita. Isso explica como as aspas são usadas em computadores. Quando dizemos:

```
PRINT "M"
```

significa que pretendemos exibir a própria letra "M". Quando digitamos:

```
PRINT M
```

significa que queremos exibir o valor armazenado na variável M. Você pode armazenar o resultado de uma operação aritmética em uma variável. Por exemplo:

```
M = 4 + 5
```

Você poderá verificar que o resultado foi armazenado, ordenando que o computador exiba o valor de M. Você também pode usar o valor de M para operações subsequentes. Por exemplo, experimente isso no computador.

```
PRINT M + 2
```

Foi exibido o valor que você esperava ? Experimente outros cálculos usando M.

O que acontece?

Se você estudar esses nomes, notará que todos eles começam com "GO" o CCE BASIC usa apenas os dois primeiros caracteres de cada nome de variável para diferenciá-lo de outros nomes de variáveis. Assim o nome:

```
GOTA
```

refere-se à mesma variável que GOLA, GOMA e assim por diante.

Aqui vai um truque útil. Vamos imaginar que você tenha algum valor armazenado na variável PRECO e você queira aumentar de 5 este valor.

Uma maneira de fazer isso, seria somar 5 ao valor da variável PRECO e então armazenar o valor resultante na mesma variável.

Por exemplo:

```
PRECO = 28
PRINT PRECO
PRINT 28 + 5
PRECO = 33
```

Mas veja como é mais fácil digitar:

```
PRECO = 28
PRECO = PRECO + 5
```

Experimente as seguintes ordens nesta sequência:

```
PRECO = 2
PRINT PRECO
PRECO = PRECO + 3
PRINT PRECO
PRECO = PRECO * 6
PRINT PRECO
PRECO = PRECO / 10
PRINT PRECO
```

No fim dessa sequência de ordens, provavelmente obterá um valor? Estará certo?

É isso que você esperava? Experimente esta sequência:

```
PERAS = 55
BANANAS = 11
QUOCIENTE = PERAS / BANANAS
PRINT QUOCIENTE
```

Primeiro pense na resposta que você esperava, depois veja se tem razão. Se não estiver certo, descubra por quê. Finalmente experimente esta:

```
PATO = 128
PRINT "PATO"
PATO = PATO /2
PRINT "PATO"
PRINT PATO
```

O que você esperava? O que obteve?

PRIORIDADES

Antigamente, em jantares formais, as pessoas eram servidas de acordo com regras rigidamente estabelecidas: em primeiro lugar o convidado de honra, então as convidadas do sexo feminino (na ordem de hierarquia de seus maridos), então os convidados do sexo masculino (em ordem de graduação) e finalmente o anfitrião. Não importando onde estavam sentados, o garçon escolhia para servir as pessoas na sequência certa. Podemos dizer que havia uma certa prioridade entre os participantes.

Em um cálculo simples como:

```
PRINT 4 + 8/2
```

você não pode dizer se a resposta certa deveria ser 6 ou 8 até que saiba a ordem (ou prioridade) para resolver o cálculo.

Se somar 4 ao 8, obtém 12. Se dividir 12 por 2, obtém-se 6. Esta é uma resposta possível. Entretanto, se você somar 4 a 8 dividido por 2, você terá 4 mais 4, ou 8. Esta é outra resposta possível. Oito é a resposta que o computador fornecerá.

Abaixo descrevemos como o computador obtém a ordem pela qual irá realizar a operação matemática:

1- quando o sinal de subtração é usado para indicar um número negativo, por exemplo:

```
-3 + 2
```

O EXATO aplicará primeiro o sinal de menos a seu número de variável adequado. Assim -3 + 2 é -1. Se o EXATO fizesse a soma primeiro, -3 + 2 seria -5. Mas isso não acontece. Outro exemplo é:

```
CALCULO = 6
PRINT -CALCULO + 10
```

A resposta é 4 (note que na expressão 5-3 o sinal de menos indica a subtração e não um número negativo);

2- depois de aplicar todos os sinais negativos o computador faz as potenciações. A expressão:

```
4 + 3 ^ 2
```

é resolvida elevando 3 ao quadrado (3 vezes 3 é nove) e então somando 4, obtendo um total geral de 13. Quando há uma série de potenciações, estas são feitas da esquerda para a direita. de modo que:

> 2 ^ 3 ^ 2 é resolvido multiplicando-se 2 por 2 três vezes (2 * 2 * 2) que resulta em 8 e então multiplicando o 8 por si mesmo mais uma vez. A resposta é 64.

3 - após terem sido calculadas todas as potenciações, são realizadas todas as multiplicações e divisões, da esquerda para a direita.

As operações matemáticas de igual prioridade são sempre efetuadas da esquerda para a direita. A multiplicação (*) e a divisão (/) têm igual prioridade;

4 - por fim, são feitas todas as somas e subtrações da esquerda para a direita. A adição (+) e a subtração (-) tem a mesma prioridade.

Vamos resumir a ordem de prioridades do computador na realização de operações matemáticas:

1º : - (sinal de menos usado para indicar números negativos);

2º : ^ (potenciação, da esquerda para a direita);

3º : * / (multiplicação e divisão, da esquerda para a direita);

4º : + - (adição e subtração, da esquerda para a direita).

Abaixo, você encontrará algumas expressões aritméticas para resolver.

Em cada um dos casos, primeiro resolva a expressão de cabeça (ou com ajuda de uma calculadora portátil ou lápis e papel) e depois experimente no computador.

Se a sua resposta não for igual a do computador, tente descobrir a razão.

Daremos aqui apenas as expressões matemáticas. Você terá que colocar um PRINT na frente de cada uma delas de modo a conseguir seu valor.

A menos que você tenha bastante experiência em cálculos c[illegible] computadores, você deve realmente fazer esse exercício. Não [illegible] faça todos de uma vez e depois confira com o computador. Faça um exemplo e confira, depois passe para o próximo e assim por diante.

```
3 + 2
4 + 6 - 2 + 1
8 * 4
4 ^ 2 + 1
6 / 4 + 1
5 - 4 / 1
6 * -2 + 6 / 3 + 8
4 + -2
2 ^ 2 ^ 3 + 1
2 * 2 * 3 + 1
2 * 2 + 1 * 3
2 * 2 + 1 + 3
8 / 2 / 2 / 1
8 * 2 / 2 + 3 * 2 ^ 2 * 1
20 / 2 * 5
```

Neste manual não são fornecidas as respostas. As respostas corretas serão dadas pelo computador.

COMO EVITAR AS PRIORIDADES

Suponhamos que você queira dividir 12, por 4 mais 2. Se digitar:

```
12 / 4 + 2
```

Você terá 12 dividido por 4, e o resultado será somado a 2. Mas não era isso que você queria. Para consegui-lo digite:

```
12 / (4 + 2)
```

Os parênteses modificam as prioridades. A regra seguida pelo computador é simples, faça primeiro o que estiver entre parênteses. Se houver parênteses dentro dos parênteses, os mais internos devem ser resolvidos primeiro. Veja este exemplo:

```
12 / (3 + (1 + 2) ^ 2)
```

Nesse caso, é efetuado o parêntese mais interno. Primeiro soma-se 1 + 2. Agora a expressão fica assim:

```
12 / (3 + 3 ^ 2)
```

Mas você sabe que 3 + 3 ^ 2 é o mesmo que 3 + 9 ou 12, portanto a expressão foi simplificada para 12 / 12, cujo resultado é 1.

Num caso como (9 + 4) * (1 + 2) onde há mais de um conjunto de parênteses, e que não estão um dentro do outro, você simplesmente trabalha da esquerda para direita. A expressão fica 13 * 3, ou seja 39. Aqui há algumas outras expressões para resolver. Ressaltamos novamente que se você não estiver familiarizado com computadores, serão valiosos os poucos minutos que você gastar resolvendo estes exercícios e treinando-os.

Você será recompensado por poder usar melhor o computador. A maior parte dessas regras para prioridades e parênteses, servem para a maioria de sistemas de computadores, em todo o mundo, e não apenas para o EXATO.

```
44 / (2 + 2)
(44 / 2) + 2
(3 + -2) * 2
3 + (-2 * 2)
100 / (200 / (1 * (9 - 5) ))
32 / (1 + (7 / 3) + (5 / 4))
```

CAPÍTULO 3

PROGRAMAÇÃO ELEMENTAR

EXECUÇÃO ADIADA

Até agora quando você digitava:

PRINT 3 + 4

e digitava a tecla [CR], imediatamente o computador executava o que você ordenava. Quando o computador atua de acordo com a instrução que recebeu, diz-se que executou a instrução. Assim, você tem usado o computador para a execução IMEDIATA de cada instrução digitada.

Você vai aprender como guardar instruções para execução posterior (execução adiada). Para ter certeza de que a memória do computador esteja limpa de programas anteriores, digita-se NEW. Como quase tudo que você já estudou, NEW deve ser seguido de um [CR].

Para mandar o computador armazenar uma instrução, apenas escreva um número antes de digitá-la. Por exemplo, se você digitar:

100 PRINT 3 + 4

nada acontece, mesmo se você apertar a tecla [CR]. O EXATO armazenou a instrução. Para ver o que foi armazenado digite LIST e [CR].

A não ser que você tenha digitado errado alguma letra e tenha recebido a mensagem: ?SINTAX ERRO; aparecerá na tela:

100 PRINT 3 + 4

Agora escreva o comando RUN e tecle [CR]; o computador exibiu o número 7 na tela.

Digitando RUN você fez com que ele executasse a instrução que foi armazenada, mas o computador não a esqueceu. Você pode executar (RUN) novamente a mesma instrução quantas vezes quiser; experimente.

Além disso, o computador não esquece a instrução quando você limpa a tela. O comando HOME é um novo método de limpar a tela.

O comando HOME tem o mesmo efeito de:

[ESC]

[SHIFT] [@ P]

que você aprendeu antes, mas pode ser usado tanto na execução adiada como na imediata.

Para experimentar isto digite:

```
100 HOME
```

Agora quando você digitar RUN o computador executará a instrução armazenada e limpará a tela. Digite NEW, depois LIST e veja o que acontece. Digitando NEW, você fez com que a instrução armazenada ficasse permanentemente anulada.

Digite RUN e não aparece nada na tela, isso acontece porque a sua instrução antiga foi anulada pelo comando NEW.

É possível armazenar muitas instruções dando a cada uma delas, um número diferente.

Experimente digitar isto.

```
1 PRINT "HELLO"
2 PRINT 4 ^ 3
3 PRINT 67 / 12
```

até agora não aconteceu muita coisa, mas agora digite RUN e observe as respostas obtidas.

Este número colocado antes de cada instrução é chamado "número da linha" e serve para informar ao computador a ordem de execução. O computador armazena e executa as instruções na ordem crescente dos números das linhas. Para verificar esse fato, apague as instruções que armazenou (digitando NEW). Agora digite:

```
1 PRINT "X"
0 PRINT "E"
3 PRINT "O"
2 PRINT "A"
```

Note que o número 0 (zero) é um número de linha permitido. O maior que você poderá usar é 63.999. Agora execute as instruções.

O resultado deve ter sido este:

```
E
X
A
O
```

Para descobrir o que aconteceu dentro da memória digite LIST.

Note que não é obrigatório ter que listar um conjunto de instruções antes de executá-las, entretanto é bom fazê-lo.

O conjunto de instruções que são executadas quando você digita RUN é chamado programa.

O programa deveria exibir:

```
E
X
A
T
O
```

mas parece que foi esquecida uma instrução PRINT. Como você poderá remediar isto ?

Apenas reescrever as instruções da linha 3 e adicionando uma nova instrução 4. Para fazer esta correção digite:

```
4 PRINT "O"
3 PRINT "T"
```

veja o que aconteceu. Liste o programa.

Note que qualquer que seja a ordem na qual as instruções foram colocadas, o EXATO as armazena com seus números de linha em ordem crescente. Execute este programa.

Foi uma chateação ter que redigitar esta instrução de modo a poder introduzir mais uma. Portanto é uma boa tática em programação, deixar alguns espaços entre os números de linhas, e também antes da primeira linha.

Digite NEW para eliminar aquele programa e coloque este:

```
100 PRINT "G"
110 PRINT "T"
```

quando você o executou, ele não exibiu a palavra "GATO" verticalmente. Digite:

```
105 PRINT "A"
120 PRINT "O"
```

e execute-o novamente.

De agora em diante este manual começará todos os programas num número de linha razoavelmente alto e deixará bastante espaço entre cada linha sucessiva de modo que haja espaço adequado para inserir instruções intermediárias.

EDIÇÃO ELEMENTAR

Anteriormente você descobriu que a instrução PRINT PDL (Ø) exibiria o número correspondente à posição presente de um dos controladores de jogo. Foram necessárias uma série de instruções PRINT para descobrir alguma coisa sobre o controle. Agora que você já pode escrever programas, as coisas ficaram mais fáceis. Limpe a memória do computador com um "NEW" e digite:

```
1ØØ PRINT PDL (Ø)
```

cada vez que digitar RUN, esse pequeno programa será executado, assim você poderá ver a posição do controlador de jogo.

Podendo ser executado várias vezes, o programa armazenado está lhe economizando trabalho. Antes, você tinha que redigitar toda uma instrução ou grupo de instruções. Agora, simplesmente digite RUN.

A execução adiada traz mais uma vantagem. Você pode modificar parte de um programa e manter o resto inalterado, sem precisar redigitar tudo. Por exemplo:

```
NEW
1ØØ P = PDL (Ø)
11Ø PRINT P
12Ø PRINT "MOVA O CONTROLADOR DE JOGO"
13Ø PRINT "PARA OUTRA POSIÇÃO"
```

Execute o programa várias vezes, alterando o ajuste do controlador de jogo entre as execuções.

Verifique se o programa responde a ambos controladores de jogo. Deve funcionar para apenas um deles. Você pode aproveitar essa oportunidade para marcar esse controlador com o número zero.

Esse mesmo programa pode ser usado com uma pequena variação, para observar o outro controlador de jogo. Liste o programa como está agora e depois digite:

```
2ØØP = PDL (1)
```

Quando você imorime uma instrução com o mesmo número de linha de uma já existente em um programa, a nova linha substitue a antiga. Liste o programa e veja o que mudou. Execute-o várias vezes para ver o que acontece. Mexa outros controladores de jogo entre as execuções. Esse programa responde a ambos controladores? Marque um número 1 no controlador que esse programa responde.

Este método de modificar um programa é um exemplo de edição de programa. Se você quiser apagar a linha 230 no programa anterior, você deve digitar:

230 e então CR

Ou então poderia usar a instrução DELete (apagar). Para apagar a linha 230 de seu programa você deve digitar:

DEL 230, 230

As vantagens do DEL (Cancelamento) não ficam aparentes até que lhe revelemos que várias linhas consecutivas de um programa podem ser apagadas com esta instrução como:

DEL 200, 230

que apagará qualquer instrução cujo número de linha seja igual ou maior a 200 e menor ou igual a 230. Experimente este comando e então liste o programa para ver o que aconteceu. A capacidade de cancelar blocos de instruções será útil quando estiver trabalhando com programas extensos.

Como visto, há vários comandos que o auxiliam a lidar com programas inteiros. São eles:

NEW que apaga programas inteiros.
LIST que os lista.
RUN que os executa, começando com a instrução que tem o menor número de linha.

Também é possível começar a execução em outro ponto e li ar apenas partes do programa. Estas possibilidades serão ex icadas mais adiante.

DESVIOS ELEMENTARES

Agora você já está começando a evoluir, portanto discutiremos LOOPS (rotinas de programação que podem ser repetidamente executadas enquanto prevaleça certa condição).

A melhor maneira de ver como funciona a função PDL é entender a utilização dos LOOPS e usar uma instrução que até agora não foi discutida. É muito simples, digite as seguintes linhas (após digitar NEW de modo a apagar programas antigos):

```
11Ø PRINT PDL (Ø)
12Ø GOTO 11Ø
```

A linha 11Ø deste programa imprime (PRINT) o número que representa o valor atual do controlador de jogos. A linha 12Ø faz o que parece dizer: (GO TO = vá para) faz com que a execução do programa volte para a linha 11Ø. O que acontece então? O programa imprime o valor atual do controlador de jogo, depois executa a linha 12Ø que manda repetir a linha 11Ø e assim por diante, para sempre. Isto é um LOOP. O loop é uma estrutura de programação que existe quando em um programa há uma instrução que faz com que outras sejam reexecutadas. Execute o programa. Mexa com o controlador de jogos. No próximo segmento lhe diremos como parar este programa. Enquanto isso, admire o fato de que se você digitou RUN quando foi instruído a fazê-lo, há 3 sentenças atrás seu computador já executou algumas centenas de vezes a instrução PRINT PDL (Ø). Agora que você já tem uma base, sua habilidade em usar o computador aumentará fantasticamente nos próximos capítulos.

ALGUMAS DICAS PARA FACILITAR

Mas primeiro, você provavelmente deve estar pensando como parar o programa dos controladores. Você já deve ter notado como os números se movem na tela ao mover o controle, isto ocorre porque os números são exibidos na base da tela e, cada número novo que é exibido empurra todos os outros números uma linha para cima. Esse processo é chamado "scrolling" (enrolamento). Você veio observando o fenômeno todo esse tempo, mas a uma velocidade bem menor. Para parar de rodar o programa use simplesmente:

CTRL

C

O comando CTRL-C faz com que você saiba onde parou a execução do programa. Exibindo "PAUSA EM 11Ø" ou qualquer número de linha onde parou o programa.

A propósito, há uma exceção à regra a respeito do fato de digitar a tecla CR após cada comando. Geralmente não é necessário digitá-los quando se para um programa com CTRL C.

Se quiser, também pode usar a tecla [RESET] para parar os programas. mas nesse caso você não receberá a mensagem que indica a última linha executada do programa.

Quando você para um programa com [CTRL] [C] ou [RESET], você pode prosseguir sua execução digitando o comando:

```
CONT
```

que significa "continue". Experimente, e depois digite este programa:

```
NEW
100 X = PDL (0)
110 PRINT "CONTROLADOR JOGO 0 ESTA EM"
120 PRINT X
130 Y = PDL (1)
140 PRINT "E O CONTROLADOR 1 ESTA EM"
150 PRINT Y
```

Anteriormente dissemos que quando você digita RUN, o programa começa a ser executado pelo menor número de linha. É verdade, entretanto, se você quiser começar a processá-lo em alguma outra linha, como por exemplo na linha 130, simplesmente digite:

```
RUN 130
```

Você também pode especificar o número de linha no comando LIST. Se digitar

```
LIST 130
```

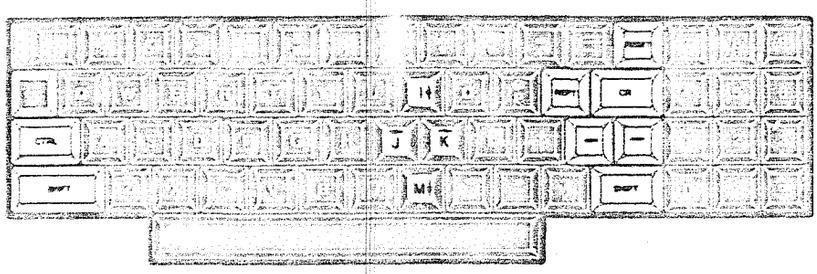
o EXATO listará a linha 130. Se você digitar

```
LIST 110, 130
```

O EXATO listará todas as linhas de seu programa, começando na linha 110 e continuando até a 130. Essa característica não pode ser obtida com o comando RUN.

O CURSOR

Quando são digitadas as teclas de retrocesso ou de avanço, elas movem o cursor, mas como você já aprendeu no segundo capítulo elas também apagam ou repetem caracteres. É possível mexer o cursor sem afetar nada (exceto, é claro, a posição do cursor). Você faz isso simplesmente movimentando o cursor. Para utilizar esses movimentos são usadas 3 teclas: [ESC], [I], [J], [K] e [M]. Veja como usá-las.



Primeiro você coloca seu computador no modo de edição teclando [ESC] , depois use [I] para subir o cursor, [J] para movê-lo para esquerda, [K], para direita e [M], para baixo. Para mexer repetidamente o cursor, segure abaixada uma das teclas indicativas de direção (I,J,K,M), e ao mesmo tempo pressione a tecla [RET]. Enquanto ambas teclas estiverem pressionadas, o cursor correrá na tela. O cursor vai parar ao chegar a base da tela e o display da mesma se moverá para cima, uma linha por vez. Se atingir a extremidade direita da tela, o cursor desaparecerá e aparecerá no início da próxima linha. Pratique mover o cursor em volta da tela com o auxílio destas 5 teclas.

Quando se cansar de movimentos simples do cursor, pressione a barra de espaço e você se encontrará no modo normal.

Os movimentos simples do cursor realizados pelas teclas [J] e [K] , parecem iguais aos movimentos causados pelas teclas de retrocesso e de avanço , mas como verá no resultado de listagem, o efeito é diferente. Os movimentos simples do cursor não causam nenhuma alteração no texto sobre o qual passam, enquanto que a tecla de retrocesso apaga e a de avanço repete os caracteres sobre os quais passam.

Esses movimentos simples do cursor têm uma aplicação, e portanto, no final das contas, não são simples. Digite por exemplo:

```
130 PRUNT "O EXATO E UM MICRO"
140 PRINT "DE OTIMA QUALIDADE"
```

teclando [CR] após cada linha. Seu EXATO parece aceitar as instruções, mas quando tentar executar o programa, você receberá pelos seus esforços a seguinte mensagem:

```
?SINTAX ERRO EM 130
```

Para fazer a correção necessária, você pode usar esse truque para repetir toda a instrucão:

primeiro liste o programa e então pressione [ESC]. Agora tecle I tantas vezes quantas forem suficientes para mover o cursor até a linha que contém a instrução errada e então use a tecla de avanço para repetir todos os caracteres anteriores ao U em "PRUNT". Tecle um I sobre o U e continue com a tecla de avanço até o fim da linha. Feito isso tecle [CR] e liste o programa para ver se a linha foi adequadamente corrigida. O computador sempre se compadece dos pobres digitadores.

Quando você precisar corrigir parte de uma linha que aparece em algum lugar da tela, para agilizar o processo podem ser usadas as teclas de avanço e os movimentos simples do cursor. Experimentar essas características do computador servirá para economizar muito trabalho mais tarde.

Estas teclas só funcionarão nas linhas sobre as quais puderem passar, isto é, as que estejam presentes na tela. Se digitar uma linha de programa e depois efetuar um movimento simples de cursor antes de teclar [CR], a linha (não os caracteres sobre os quais irá se mover o cursor) será afetada pelo movimento da tecla de retrocesso. Entretanto, a tecla "avanço", repete os caracteres sobre os quais o cursor passa.

COMO FIXAR O QUE ESTA ESTUDANDO

Muitas vezes há perguntas que você pode fazer a respeito da linguagem CCE BASIC que não são diretamente respondidas por este manual. Nesta instrução por exemplo:

```
PRINT "BERNARDO"
```

você precisa deixar um espaço depois da palavra "PRINT"? Aqui não terá a resposta, recomendamos que ligue o seu computador e experimente ambas as possibilidades. Geralmente uma experiência simples responderá sua pergunta e, já que resolveu experimentar por si mesmo, você se lembrará muito melhor do que se tivesse simplesmente lido o que fazer.

UM ACIDENTE PODE ACONTECER

Anteriormente neste capítulo você aprendeu como apagar uma linha digitando seu número e teclando [CR]. Este é o método favorito de introdução de erros em seus programas. Suponhamos que queira eliminar a linha 1100 de um programa, mas se enganou e digitou 110 e [CR]. Parabéns, você apagou a linha 110. Isto acontece. Ou se está para corrigir a linha 450 e então digita este número, pensa melhor e resolve, no fim das contas, não alterar nada. Não tecle [CR]! Retroceda sobre o número ou use o comando especial [CTRL] [X] usando este comando você coloca uma barra inclinada no fim da linha que estava digitando e será como se nunca a tivesse digitado.

LOGICA ARITMETICA

O computador pode distinguir entre o que é verdadeiro e o que é falso. Já que isto é mais do que a maioria de nós podemos fazer, devemos dar-lhe algumas palavrinhas de explicação.

O símbolo ">" significa "maior que".

A afirmação 6 > 2 (que significa, seis é maior que dois) certamente é verdadeira. O computador se utiliza do número 1 (um) para indicar a verdade.

Se digitar:

```
PRINT 6 > 2
```

o computador responderá com 1.

A afirmação 55 > 78 é falsa. O computador usa o número 0 (zero) para indicar a não verdade.

Se digitar:

```
PRINT 55 > 78
```

o computador responderá com um 0.

O símbolo "<" significa "menor que".

Aqui está o conjunto completo de símbolos usados em afirmações.

```
>  - maior que
<  - menor que
=  - igual a
>= - maior ou igual a
<= - menor ou igual a
<> - diferente de
```

Para digitar os símbolos de "maior ou igual a" e "menor ou igual a" no teclado do seu EXATO, primeiro é necessário digitar > ou < e depois o =. Para digitar o símbolo "diferente de" você precisa digitar < e depois o >.

Pense, e depois teste, para verificar quais destas afirmações são verdadeiras e quais são falsas:

```
5 <> 5
6 > 2
8 > 8
8 <= 8
9354 = 4359
5 < 8
-5 >= -4
-8 < -7
-2 >= -5
```

As afirmações podem incluir variáveis e expressões, bem como números. Por exemplo:

```
PRINT (45*6) <> (45+6)
```

Exibirá o valor 1 já que 270 não é igual a 51 (lembre-se de que "1" significa que a afirmação está correta).

Você já viu que o EXATO pode diferenciar o certo do errado em afirmações numéricas simples. Entretanto, afirmações como GATO > CAO podem ser verdadeiras ou falsas dependendo do valor das duas variáveis. Se:

```
GATO = 5 e CAO = 9
```

então a afirmação GATO > CAO é falsa, mas se:

```
GATO = -8 e CAO = -15
```

então será verdadeira.

Afirmações para o computador têm valores numéricos zero ou um. E podem ser usadas em expressões aritméticas e ao invés de 1 e 0 poderá ter como resposta outros valores.

Por exemplo,

```
PRINT 3 + (4>2)
```

exibirá o valor 4. A afirmação:

```
T = 4 <> 3
```

dá ao T o valor 1, já que 4 não é igual a 3 e assim, a afirmação 4 <> 3 tem valor 1. A afirmação:

```
PATO = 67 = 19
```

a princípio parece muito estranha, mas é facilmente compreendida já que 67 não é igual a 19, a afirmação 67 = 19 é falsa e tem o valor zero. O valor zero é conferido à variável PATO (PATO = Ø).

Como já vimos, o EXATO usa 1 para dizer verdadeiro e Ø para falso. Se alguma coisa não é verdadeira, é falsa. Se algo não é falso, é verdadeiro. Este pode não ser sempre o caso na vida cotidiana mas é sempre o caso com os computadores. Experimente isto:

```
PRINT NOT 1
```

e então:

```
PRINT NOT Ø
```

O computador concorda: Não verdadeiro, é falso, e não falso, é verdadeiro. É claro que você pode usar expressões dos números 1 e Ø.

Assim:

```
PRINT NOT (45 > 3)
```

a sentença:

Triângulos tem três lados.

é verdadeira. E a sentença:

Este manual está escrito em português.

também é verdadeira.

Considere a sentença

Triângulos tem três lados e o manual está escrito em português.

Esta sentença é verdadeira ou falsa? É verdadeira. Considere a sentença:

Triângulos tem 8 lados e o livro está escrito em português. A sentença, como um todo, é falsa.

Triângulos tem 8 lados e o livro está escrito em japonês. Esta sentença também é falsa. Em geral, quando combina-se duas sentenças ou afirmações unindo-as com a letra E, descobre-se que:

a. a sentença nova é verdadeira, se ambas sentenças originais forem verdadeiras;

b. a nova sentença é falsa se pelo menos uma das sentenças originais era falsa.

O computador pode determinar se uma afirmação contendo várias afirmações ligadas por "E", é verdadeira ou falsa. Teste-o com as seguintes instruções. Tente predizer as respostas:

```
PRINT 1 AND 1
PRINT 1 AND Ø
PRINT Ø AND 1
PRINT Ø AND Ø
PRINT (3 > 2) AND Ø
PRINT (NOT Ø) AND (4 = 5)
```

Esta sentença é verdadeira ou falsa?

Um triângulo tem três lados ou este livro está escrito em latim.

É verdadeira. Um triângulo tem 3 lados, mesmo se o livro não estiver escrito em latim; portanto, a sentença como um todo, é verdadeira. Quol erat demonstrandum.

Em geral, quando junta-se 2 sentenças com um OU, descobre-se que:

a. a nova sentença é verdadeira se uma ou ambas sentenças originais eram verdadeiras;

b. a nova sentença é falsa se ambas sentenças originais eram falsas.

O computador pode determinar se uma afirmação contendo várias afirmações ligadas por OU é verdadeira ou falsa. Tente estas no seu computador, depois de verificar quais serão as respostas.

```
PRINT 1 OR 1
PRINT 1 OR Ø
PRINT Ø OR 1
PRINT Ø OR Ø
PRINT (4 <>5) OR (4=5)
PRINT 1 OR (Ø AND 1)
PRINT ((3>4) OR (54 < 337)) AND (NOT Ø)
```

No próximo capítulo AND, OR e NOT se tornarão de extrema utilidade.

Você também já descobriu que na instrução:

```
PRINT 1 OR Ø
```

O computador considera 1 como verdadeiro e Ø como falso. Agora experimente:

```
PRINT 23 OR Ø
```

e isto:

```
PRINT - 247 AND 32707.61
```

Nas afirmações, o computador não considera apenas 1, mas qualquer número diferente de zero, como verdadeiro. Entretanto, quando ele obtém o valor de uma afirmação, esse valor será sempre Ø ou 1.

Enquanto você não tiver conhecimento das regras de prioridade para AND, OR e NOT, recomendamos que use parênteses para tornar claras as suas instruções.

PRIORIDADES DE EXECUÇÃO

1. ( ) : parênteses
2. NOT : para valores negativos.
3. ^ : potenciação
4. *, /: multiplicação e divisão.
5. -,+ : adição e subtração
6. >, <, =, >=, <=, <>
7. AND
8. OR

A INSTRUÇÃO IF

Suponhamos que se queira exibir na tela números inteiros de 1 a 10, um número por linha. Um método óbvio de fazê-lo é:

```
NEW
210 PRINT 1
220 PRINT 2
230 PRINT 3
```

e assim por diante. Mas isso requereria 10 instruções e se quisesse exibir números inteiros de 1 a 200, seriam necessárias 200 instruções. Usando o que já aprendeu, você pode exibir números inteiros de 1 em diante usando somente 4 instruções com o auxílio de um loop.

```
200 N = 1
210 PRINT N
220 N = N + 1
230 GOTO 210
```

Há uma maneira de controlar a extensão de um loop, se não ele nunca vai parar. O que você quer é uma instrução que realize um GOTO se N é, por exemplo, menor do que 11, mas não faça o GOTO se N for maior do que 11. A resposta para o que deseja é a instrução IF . Se uma condição se realiza, o computador pulará a instrução GOTO e executará a instrução da próxima linha. Se não houver instrução na linha seguinte. Ele parará a execução.

Aqui está um programa que exibe de 1 a 10 e então para:

```
200 N = 1
210 PRINT N
220 N = N + 1
230 IF N < 11 THEN GOTO 210
```

Em geral, a instrução IF, funciona assim:

SE (expressão aritmética) ENTÃO (qualquer instrução)

Primeiro é calculada e avaliada a expressão aritmética. Se o resultado for zero (falso), todo o resto da linha no programa é ignorado e a execução passa para a linha seguinte. Se o resultado for diferente de zero (verdadeiro), o restante da linha do programa é executada.

A instrução IF é muito poderosa, e aparecerá em quase todos os programas que fizer. Só para se divertir, digite o seguinte programa:

```
NEW
400 GR
410 LINHA = 1
420 COLOR = LINHA
430 HLIN 0, 39 AT LINHA
440 LINHA = LINHA + 1
450 IF LINHA < 16 THEN GOTO 420
```

ARMAZENANDO PROGRAMAS

(Se não estiver usando o drive, pule este segmento).

Nesse ponto, você pode estar querendo armazenar no disquete alguns dos programas que está usando. Simplesmente digite seu programa (após ter digitado NEW) e depois, no final digite SAVE seguido do nome que quiser usar ao se referir ao programa, e então, é claro o [CR]. Por exemplo, se quiser gravar (SAVE) o programa acima e chamá-lo de TESTE, você deverá digitar:

```
SAVE TESTE
```

e o programa ficará armazenado no disquete sob o nome de TESTE. Uma vez que o programa foi armazenado, digite CATALOG seguido de [CR], para ver o nome de seu programa listado com os outros programas no disquete. Então será capaz de executar (RUN) o programa chamado TESTE, armazenado no disquete sempre que quiser, digitando

```
RUN TESTE
```

Experimente armazenar o programa de sua escolha. Se, acidentalmente digitar um comando errado (e obter uma resposta ?SINTAX ERRO), simplesmente redigite o comando corretamente.

Às vezes deseja-se carregar (LOAD) um programa na memória do computador sem executá-lo (RUN). Por exemplo, se você quiser modificar o programa antes de executá-lo (RUN). O comando LOAD é útil nesse caso. Para usá-lo, simplesmente digite LOAD seguido do nome do programa que deseja carregar. Por exemplo, se você quisesse carregar o programa chamado TESTE, você digitaria:

```
LOAD TESTE
```

Se você alterou o programa e então deseja executar a nova versão que está na memória, mas não a que está armazenada no disquete, lembre-se de digitar apenas RUN.

Se você se esquecer e digitar:

```
RUN TESTE
```

a versão antiga armazenada no disquete, será recarregada apagando nova versão em memória.

Você pode usar os comandos LOAD e SAVE para copiar os programas de um disquete para outro, (carregando) LOAD 1 programa de um disquete e (armazenando) SAVE o programa em outro disquete. Pratique usar os comandos SAVE e LOAD.

ARMAZENANDO PROGRAMA EM FITA

(Pule esse segmento se não estiver usando um gravador)

Para armazenar um programa numa fita cassete para uso posterior, coloque uma fita virgem no gravador e rebobine-a até o início da fita.desta maneira o seu programa será mais facilmente encontrado.

No gravador, pressione a tecla PLAY, enquanto pressiona a tecla RECORD. Ambas devem permanecer abaixadas. Voltando ao computador digite:

```
SAVE
```

Quando teclar CR , o cursor desaparecerá. Após 10 ou 15 segundos, o computador emitirá um bip para avisá-lo de que a gravação começou. Outro bip soará quando a gravação acabar e o cursor reaparecerá. Aperte o botão STOP do gravador, rebobine a fita até o começo e você estará pronto para voltar à programação. Seu programa em memória não ficou afetado pelo fato de tê-lo armazenado (SAVE).

MAIS PROGRAMAS GRÁFICOS

Antes, você colocou quatro pontos coloridos nos cantos da tela. Agora digite este programa:

```
NEW
190 GR
200 COLOR = 9
210 PLOT 0, 0
220 PLOT 0, 39
230 PLOT 39, 39
240 PLOT 39, 0
```

Liste o programa para verificar se foi corretamente digitado e então execute-o. Rápido, não é? Para mudar as cores, apenas altere a linha 200 e execute o programa novamente. Tente listar o programa. Note que a listagem não cabe na estreita janela na base da tela. Isso continuará a acontecer, a não ser que você digite:

```
TEXT
```

para sair do modo gráfico.

O programa seguinte faz com que a tela fique cor de laranja

```
NEW
200 GR
210 COLOR = 9
220 COLUNA = 0
230 VLIN 0, 39 AT COLUNA
240 COLUNA = COLUNA + 1
250 IF COLUNA < 40 THEN GO TO 230
```

Aqui temos uma explicação passo a passo do que acontece quando você executa este programa.

A linha 200 ajusta o comutador em modo gráfico. A cor é escolhida na linha 210. O programa deve começar na coluna 0 da tela e seguir seu caminho até a coluna 39. A linha 220 assegura que o programa comece na coluna 0. Agora que a coluna 0 está preenchida com a cor desejada, a linha 240 aumenta de uma unidade a coluna. O valor da COLUNA agora é 1. A linha 250 verifica se o novo valor da COLUNA é menor que 40. Se for menor que 40, o programa volta a linha 230 para desenhar uma nova linha vertical na coluna indicada. Entretanto, quando o valor da coluna atinge 40, na tela do EXATO, a coluna da extrema direita é a 39, o programa não volta a linha 230, mas sua execução para, porque alcançamos o fim do programa.

Para eliminar a necessidade de digitar RUN a cada vez que desejar encher a tela de cor, digite:

```
260 GOTO 210
```

Observe o que acontece. Quando vai parar este programa? Liste o programa e assegure-se de que o entendeu antes de prosseguir a leitura deste manual.

Quando você parar de usar o programa de cores (tela colorida), limpe a tela e digite o programa abaixo. Ele usa uma nova e importante instrução: a instrução REM, "REM" significa REMark (Observação).

Essa instrução possibilita que você insira observações no programa. O computador ignora qualquer instrução REM, elas são estritamente para o uso dos usuários. Veja como é fácil entender esse programa, onde os "REM" são livremente usados:

```
200 REM AJUSTE MODO GRAFICO
210 GR
220 REM ESCOLHA COR
230 COLOR = 1
240 REM LEIA CONTROLADOR ZERO
250 X = PDL(0)
260 REM DETERMINA QUE O VALOR MAXIMO DE X SEJA 36
270 X = X/7
280 R  LEIA CONTROLADOR UM
290 Y  PDL (1)
300 REM LIMITA VALOR MAXIMO DE Y
310 Y = Y/7
320 REM EXIBIR O PONTO
330 PLOT X,Y
340 GOTO 250
```

Após ter digitado RUN, opere os controladores de jogo. O programa é chamado "DESENHOS". É necessária a divisão por 7 já que a função PDL fornece valores entre 0 e 255, enquanto que a tela pode apenas aceitar valores de colunas e linhas de 0 a 39. Ao dividir por sete, terá valores de 0/7 = 0 a 255/7 = 36,4285715. Então este valor é arredondado automaticamente, para baixo, isto é, o número inteiro mais próximo cujo valor seja menor ou igual ao valor dado. Em outras palavras, as coordenadas X e Y são arredondadas para números inteiros de 0 a 36 de modo que as coordenadas X e Y possam ser (traçadas) PLOT. Esse método não utiliza a altura ou largura total da tela. Para se conseguir a largura total da tela, ao invés de:

```
270 X = X/7
```

você poderia usar 2 linhas:

```
270 IF X > 239 THEN X = 239
275 X = X/6
```

Para usar a altura total da tela, você poderia fazer o mesmo, usando a coordenada Y.

A instrução IF limita o valor de X em 239. Nos cálculos feitos pelo EXATO 239/6 é arredondado de 39.833333 para 39. É muito comum o uso da instrução IF para limitar a extensão de uma variável.

INSTRUÇÕES FOR E NEXT

"Loops" executados por aviões ou computadores, têm uma base e um topo, ou seja, um começo e um fim. No programa:

```
NEW
100 NUMERO = 0
110 PRINT NUMERO
120 NUMERO = NUMERO + 1
130 IF NUMERO <= 12 THEN GOTO 110
```

a linha 110 é o topo do "loop" e a 130 é a base. O programa exibe os números inteiros de 0 a 12. O número 12 é o limite do "loop". Outra maneira de fazer o "loop" é usar uma instrução que ainda não vimos: a instrução FOR. Podemos usar essa instrução no programa anterior:

```
200 FOR NUMERO = 0 TO 12
210 PRINT NUMERO
220 NEXT NUMERO
```

Para executar esse programa use:

```
RUN 200
```

Se você simplesmente digitar RUN, o programa será executado a partir da linha 100 (número de linha mais baixo disponível).

A linha 200 contém a nova instrução FOR. Começa igualando a variável NUMERO à 0. Esta é exatamente a mesma função que a linha 100 executou. Então é executada a linha 210. O fim do loop FOR é a linha 220. A variável NUMERO é acrescida de 1 e então comparado ao limite superior 12 especificado na instrução FOR. Se NUMERO não estiver acima do limite, a execução continua na instrução imediatamente seguinte ao FOR. Mas se a variável estiver acima do limite, o programa desvia para a primeira instrução após o NEXT. Nesse caso, não há instruções após o NEXT, portanto, ele para a execução.

A vantagem mais óbvia do método FOR/NEXT de construção de "loops" é que economiza uma instrução. É a mais importante é que você não precisa pensar tanto ao criar um loop. Se você queria desenhar uma série de linhas horizontais na tela, usando cada uma das 15 cores poderia digitar:

```
3000 GR
3010 FOR N = 0 TO 15
3020 COLOR = N
3030 HLIN 0,39 AT N
3040 NEXT N
```

Outra vantagem, é que é muito mais simples ler uma única instrução FOR, do que interpretar 3 instruções para descobrir o que um "loop" está fazendo. Tudo o que tem a fazer para descobrir a base de um loop FOR/NEXT é procurar um NEXT que tenha a mesma a variável do FOR.

Seria bom mencionar que, embora você já saiba como funciona a instrução FOR, não é obrigatório usá-la. Ela não acrescenta novos conhecimentos. Simplesmente torna o programa mais fácil de ser digitado e interpretado.

Nesse ponto, se você tem feito todos os programas no seu computador, deverá retirar a porção do programa entre as linhas 3000 e 3040. Portanto digite :

```
DEL 3000, 3040
```

e então liste o programa para verificar os resultados.

Para exibir somente números pares de 0 a 12, você poderia usar o programa:

```
100 PAR = 0
110 PRINT PAR
120 PAR = PAR + 2
130 IF PAR <= 12 THEN GOTO 110
```

O segredo está na linha 120 onde 2 é somado à PAR, dizemos que o "loop" avança dois. Para avançar 2 em um "loop" FOR digite:

```
300 FOR PAR = 0 TO 12 STEP 2
```

O resto do programa ficará semelhante as linhas de 210 a 220 na página anterior, exceto que a palavra NUMERO terá que ser alterada, sempre que isso ocorrer, para a palavra PAR. O valor de STEP pode ser qualquer número dentro do âmbito do computador. Ele também pode ser negativo, por exemplo:

```
200 FOR COISA = 39 TO 15 STEP - 3
```

Digite esta linha e depois:

```
RUN 200
```

Você deve praticar durante algum tempo o uso da instrução FOR, se quiser mesmo fixá-la. De agora em diante, muitos programas usarão esta instrução.

Juntamente com as conveniências da instrução FOR, temos algumas limitações. Por exemplo os loops FOR/NEXT podem estar aninhados mas não podem se cruzar. Aqui há alguns exemplos que demonstram a idéia:

```
NEW
300 GR
310 FOR M = 1 TO 15 <-------------------------------
320 COLOR = M                                       !
330 FOR LINHA = 0 TO 39 <----------------           !
340 HLIN 0, 39 AT LINHA                  !          !
350 NEXT LINHA <-------------------------           !
360 COLOR = M - 1                                   !
370 FOR COLUNA = 0 TO 39 <---------------           !
380 VLIN 0, 39 AT COLUNA                 !          !
390 NEXT COLUNA <------------------------           !
400 NEXT M <----------------------------------------
```

Esse programa é um exemplo de loop em 2 niveis internos. Pense nisso e execute este programa antes de passar para o próximo. Lembre-se que quando fizer programas usando a instrução FOR para cada uma delas deve haver um NEXT correspondente.

UM PROGRAMA ERRADO

```
NEW
500 FOR N = 10 TO 20 <--------------------
510 PRINT N                               !
520 FOR J = 30 TO 40 <-----------------    !
530 PRINT J                           !   !
540 NEXT N <--------------------------!---
550 NEXT J <--------------------------
```

Esse programa não vai funcionar. Seus loops estão cruzados, o que não apenas dá uma mensagem de erro, como também não tem sentido. Sempre que você estiver fazendo loops cruzados, seu pensamento se embaralha. Se tiver certeza do que está fazendo e, mesmo assim quiser usar loops cruzados, use loops feitos com a instrução IF. Estes você pode cruzar quanto quiser.

LOOPS INTERDEPENDENTES

```
NEW
300 GR
310 M = 0
320 FOR COLUNA = 0 TO 35 STEP 5  <------------------
330 FOR LINHA = 0 TO 30 STEP 10  <--------------    !
340 M = M + 1                                  !    !
350 IF M > 15 THEN M = 0                       !    !
360 COLOR = M                                  !    !
370 FOR FILEIRA = LINHA TO LINHA + 9  <-----   !    !
380 HLIN COLUNA,COLUNA + 4 AT FILEIRA      !   !    !
390 NEXT FILEIRA  <-------------------------   !    !
400 NEXT LINHA  <------------------------------    !
410 NEXT COLUNA  <-------------------------------------
```

Este programa tem um aninhamento de três níveis e faz com que a tela fique xadrez. Note que COLOR não pode ser usado como variável FOR/NEXT. COLOR é uma palavra reservada do CCE BASIC. Experimente executar o programa no modo texto removendo a linha 300. O que acontece?

CHAMANDO A ATENÇÃO

Se você já está cansado de letras brancas sobre um fundo preto, vai adorar esta seção. Digite:

INVERSE

e dê uma olhada no cursor. Ele deve estar em preto sobre um fundo branco. Agora digite um programa simples como:

```
100 PRINT "PRETO E BRANCO"
```

e execute, não é mais interessante ?

Agora digite:

FLASH

e rode o programa novamente. Você verá a mesma frase "piscando'.

Note que INVERSE e FLASH afetam apenas onde é exibido pelo computador. Não são alterados os caracteres que aparecem na tela quando você os digita. Estes comandos podem ser usados tanto na execução imediata quanto na adiada. Experimente-os, depois de ter usado os comandos FLASH e INVERSE durante algum tempo, você pode ter decidido que, afinal das contas, branco e preto não é tão macante assim. Se for assim, digite NORMAL para fazer o modo texto voltar ao normal.

FORMATANDO A TELA

Como experiência, digite este programa e veja o que acontece quando você o processa:

```
NEW
100 PRINT "CCE"
110 GOTO 100
```

Pare o programa com CTRL C e então altere a linha 100 colocando uma vírgula no fim da mesma.

```
100 PRINT "CCE",
```

e execute novamente o programa. Como se pode ver, ao colocar a vírgula o programa exibe as palavras em colunas. Agora substitua a vírgula por um ponto e vírgula.

```
100 PRINT "CCE";
```

e execute o programa. Dessa vez o resultado é comprimido. Isto é, não há espaço entre o que você instruiu o computador a fazer. Ele exibe "CCE" após "CCE" até que rapidamente a tela fique cheia.

Altere o programa, inserindo a instrução:

```
90 V = 99
```

e altere a linha 100 para:

```
100 PRINT V
```

Processe-o. Agora altere a linha 100 para:

```
100 PRINT V,
```

e processe-o novamente. Então altere a linha 100 para:

```
100 PRINT V;
```

e observará que a vírgula e o ponto e vírgula também podem ser usados com valores numéricos. A capacidade de colocar números uns após os outros, sem espaço entre eles, às vezes torna-se bastante útil. A vírgula e o ponto e vírgula podem ser usados dentro de uma instrução PRINT.

Limpe a memória com um NEW e digite:

```
100 TIROS = 2
110 BALAS = 3
120 PRINT TIROS,BALAS
```

Você pode fazer uma saída mais clara, incluindo mensagens na instrução PRINT. Por exemplo, altere a linha 120 para:

```
120 PRINT "OS TIROS E AS BALAS SAO: ";TIROS,BALAS
```

Note que, provavelmente você deseja obter um espaço após os dois pontos a fim de que o número de tiros não fique próximo demais.

Se você acha que fica feio o espaço grande entre o número de tiros e balas, você poderá usar a instrução:

```
120 PRINT "OS TIROS E AS BALAS SAO: ";TIROS;" ";BALAS
```

Nesta versão, um espaço em branco é colocado entre o número de tiros e balas. Talvez o modo mais estético de fazer isso seja:

```
120 PRINT "TIROS ";TIROS;" BALAS "; BALAS
```

essa instrução lhe dará um display mais claro.

Digamos que você quisesse exibir a palavra AQUI começando na décima coluna (lembre-se que a largura máxima é de 40 colunas) poderá usar esta instrução:

```
120 PRINT "         AQUI"
```

(foram deixados 9 espaços em branco antes da palavra AQUI). Assim como nas máquinas de escrever convencionais, você pode aplicar um tabulador no EXATO é a instrução TAB, usada da seguinte maneira:

```
120 PRINT TAB(10)"AQUI"
```

tem o mesmo efeito que colocar 9 espaços em branco entre as aspas, como fizemos antes. Experimente. Você vai gostar.

Combinando o TAB com o "loop" FOR/NEXT, você pode programar alguns efeitos visuais bonitos. Por exemplo:

```
NEW
200 FOR N = 1 TO 24
210 PRINT TAB(N)"X"
220 NEXT N
```

Há 24 linhas horizontais. Esta, a propósito, é a razão do limite superior do "LOOP", no programa acima, ser 24. TAB não pode ser usado para retroceder (para a esquerda) numa linha. Só são efetuados movimentos para a direita. Para formar uma determinada linha, você pode tabulá-la na vertical (VTAB). A linha superior é a 1 e a inferior é a 24. Ao contrário do TAB, VTAB não é usado dentro de uma instrução PRINT.

Se não quiser usar uma instrução PRINT, você pode tabular na horizontal usando o HTAB. O HTAB também pode fazer com que o caráter exibido comece a direita ou a esquerda da posição atual. O caráter na posição mais a esquerda em dada linha está na posição 1, enquanto que o mais a direita está na 40. Digite este pequeno programa que demonstra o uso do HTAB e VTAB:

```
NEW
590 HOME
600 FOR X = 1 TO 24 <---
610 FOR Y = 1 TO X <-- !
620 HTAB X          ! !
630 VTAB Y          ! !
640 PRINT "EXATO"   ! !
650 NEXT Y <--------- !
660 NEXT X <----------
670 GOTO 600
```

Antes de executar esse programa, tente (não é fácil) imaginar o que ele fará.

O resultado além de bonito, é surpreendente.

TAB funciona para o modo de execução imediata, mas você só pode usar VTAB e HTAB em programas. Embora TAB, HTAB e VTAB atuam mais ou menos como coordenadas de PLOT (gráfico), há algumas diferenças. As 40 colunas para instrução TAB são numeradas de 1 a 40, como seriam numa máquina de escrever, ao passo que uma coordenada de uma instrução PLOT pode ir de 0 a 39, o que é mais eficaz para trabalhar com gráficos. Já que os caracteres são mais altos do que os "blocos" com os quais construimos gráficos, na tela há lugares para apenas 24 linhas . Portanto, os limites de VTAB são 1 e 24. O zero ou um número que seja muito grande, ou muito pequeno para TAB, VTAB ou HTAB lhe dará a mensagem:

?VALOR ILEGAL

O maior valor para VTAB é 24, mas o maior valor para TAB ou HTAB é 255. Tanto TAB como HTAB ultrapassam a extensão da tela e pulam até a próxima linha. Para ver esse fato em ação digite:

```
NEW
300 FOR K = 1 TO 255
310 PRINT TAB (K) K
320 NEXT K
```

então tente trocar as linhas 310 e 320 por:

```
310 HTAB K
320 PRINT K
```

e insira:

```
330 NEXT K
```

o que acontece com o programa quando você troca HTAB por VTAB?

CAPÍTULO 4

MUITOS GRÁFICOS

CONVERSANDO COM UM PROGRAMA

Aqui temos um programa que faz um ponto colorido mover-se através da tela, batendo nas extremidades direita e esquerda.

```
NEW
400 REM ESCOLHER A COR DA BOLA
420 BOLA = 9
440 REM ACIONAR MODO GRAFICO
460 GR
480 REM POSICAO INICIAL
500 XANT = 20
520 REM MOVA A BOLA PARA A FRENTE E PARA TRAS
540 XMOV = 1
560 REM NOVA POSICAO DE X
580 XNOV = XANT + XMOV
600 REM BOLA ESTA NA TELA?
620 IF (XNOV >= 0) AND (XNOV < 40) THEN GOTO 720
640 REM ALTERE DIRECAO XMOV
660 XMOV = -1 * XMOV
680 GOTO 580
700 REM PLOT A NOVA BOLA
720 COLOR = BOLA
740 PLOT XNOV, 20
760 REM APAGUE BOLA ANTIGA
780 COLOR = 0
800 PLOT XANT, 20
820 REM GUARDE BOLA NOVA
840 XANT = XNOV
860 REM MOVIMENTE NOVAMENTE
880 GOTO 580
```

Você sempre deve pensar nos nomes a atribuir às variáveis. Pode parecer que XNEW seria um nome mais conveniente para XNOV, até você lembrar que NEW tem um significado especial para o CCE BASIC e que é uma palavra reservada. A razão da variável XMOV ter sido chamada XMOV ficará evidente se alterar seu valor. Experimente, por exemplo, XMOV=2. Se você atribuir um valor muito alto para XMOV, a bola parecerá pular muito na tela, sem deixar nenhuma marca entre as posições.

Este tipo de programa é básico para muitos jogos de vídeo. Vale a pena passar algum tempo estudando-o fazendo alterações só para ver o que pode acontecer. Seria uma boa idéia salvar (SAVE) este programa, assim não teria que digitá-lo novamente, no caso de fazer alguma alteração fatal.

Quando listar este programa, ele não caberá inteiro na tela e as linhas passarão rápido demais para poderem ser facilmente lidas. Um modo de ver todas as linhas do programa é listá-lo em partes. Assim você poderia digitar:

```
LIST 400, 680
```

e apenas seriam exibidas as linhas com números maiores que 400 e menores que 680. E então, você poderia listar o resto do programa. Você também pode usar:

CTRL

S

para interromper a listagem do programa.

Experimente listar o programa e depois teclar rapidamente

CTRL

S

antes que as linhas passem do topo da tela. Para continuar a listagem pressione CTRL S novamente. Para a listagem também pode ser usado CTRL C . Entretanto este comando abortará a listagem de modo que não poderá continuar além do ponto onde parou sem digitar LIST novamente.

Agora você está em posição de entender o programa da bola saltitante, mas talvez seus amigos não entendam. Suponha que quisesse que seus amigos fossem capazes de escolher a cor da bola. Você poderia explicar como mudar a linha 420, mas também teria que explicar as possíveis mensagens de erro e como evitá-las. Isto daria um bocado de explicações. Seria melhor deixá-los interagir com o programa. Para fazer isto, você pode usar uma instrução INPUT. Altere a linha 420 para:

```
420 INPUT BOLA
```

Quando o programa executar esta instrução, aparecerá um ponto de interrogação na tela, seguido do cursor. O computador esperará então até que alguém digite um número seguido do CR. O número digitado se tornará o valor da bola e o programa voltará a ser executado. Pode ser uma boa idéia fazer com que o computador diga aos seus amigos o que é esperado. Você poderia colocar uma instrução PRINT assim:

```
390 REM ACIONE MODO TEXTO
400 TEXT
320 PRINT "PARA ESCOLHER A COR DA BOLA SALTITANTE"
340 PRINT "DIGITE UM NUMERO DE 1 A 15"
360 PRINT "DEPOIS DO PONTO DE INTERROGACAO"
380 PRINT "E ENTAO TECLE CR"
```

Você também pode incorporar uma mensagem na instrução INPUT.

```
420 INPUT "QUAL A COR QUE VOCE GOSTARIA QUE A BOLA
    TIVESSE (1 A 15) ?";BOLA
```

Note que na instrução INPUT, a mensagem tem que estar entre aspas e tem que haver um ponto e vírgula entre a mensagem e a variável. Quando a instrução INPUT contém uma mensagem, não é colocado a seguir um ponto de interrogação. Se quiser que ele apareça, inclua-o na mensagem.

Seus amigos podem usar as teclas de retrocesso [←] e de avanço [→] para corrigir erros de digitação, mas se fizerem um erro e então teclarem [CR], receberão uma mensagem de erro. Se o caráter que se introduziu não for um numérico

```
?REENTRE
?
```

aparecerá na tela. Se o número introduzido for muito grande ou muito pequeno, o programa ou fará a bola se mover para a direita da tela e então parar, ou aparecerá a mensagem:

```
?VALOR ILEGAL EM 720
```

na tela e o programa irá parar. Na maioria das vezes, o usuário não saberá como recolocar o computador em funcionamento e nem terá que saber. Portanto, você deverá fazer com que o programa verifique se todos os números digitados pelos usuários estão corretos.

Isto se faz da seguinte maneira:

```
424 REM BOLA ESTA ENTRE 1 E 15?
428 IF (BOLA > 0) AND (BOLA < 16) THEN GOTO 460
432 PRINT "ESTE VALOR NAO ESTAVA ENTRE 1 E 15. FAVOR
    REDIGITAR"
436 GOTO 420
```

Agora você está começando a perceber porque nós aconselhamos a deixar tanto espaço entre os números de linha?

Uma boa tática de programação é fazer um programa tão à prova de erros quanto possível. Você já chegou ao ponto onde está escrevendo suas próprias mensagens de erro para os outros lerem. Pode estar claro para um programador como você, ler a mensagem ?SINTAX ERRO, mas não é justo forçar um inocente e leigo usuário a lidar com esses mistérios, pois este não saberá como interpretá-los.

Toda vez que você usa uma instrução INPUT, seu programa precisa verificar se aquilo que o usuário digitou, está dentro de certos limites, de modo que o programa não se perca, ou falhe de qualquer outra maneira. Lidar com o usuário destreinado (e você precisa assumir que o usuário não é um programador) é, em si, uma arte. É sempre necessário usar sentenças em linguagem clara e fazer a verificação de tudo que o usuário digita.

A propósito, você pode inserir vários valores com uma única instrução INPUT. A instrução:

```
3000 INPUT X,Y,Z
```

como de costume, mostra um ponto de interrogação e, depois espera que os três números sejam digitados. O primeiro número seria armazenado na variável chamada X; o segundo, na variável Y; e o terceiro, na Z. Os três números precisam estar separados por CR ou vírgulas e o último número precisa ser seguido de CR.

Por via das dúvidas, salve sua melhor versão do programa da bola saltitante. Então, se ainda não o fez, tente somar ao programa um movimento vertical da bola. Use as novas variáveis YANT, YNOV e YMOV. A seguir é fornecida uma solução, mas experimente resolver sozinho antes de olhar.

Quando o programa estiver sendo processado do jeito que você quiser, salve-o no disquete ou na fita. Mais tarde voltaremos a usá-lo.

MOVIMENTO EM QUATRO DIREÇÕES

Aqui está uma maneira de fazer a bola saltar em quatro direções. As instruções precedidas de asterisco são aquelas que foram inseridas ou alteradas a partir do programa que oscilava a bola entre duas paredes.

```
* 260 REM ACIONE MODO TEXTO
* 300 TEXT
* 320 PRINT "PARA ESCOLHER UMA COR PARA BOLA"
* 340 PRINT "DIGITE UM NUMERO DE 1 A 15"
* 360 PRINT "APOS O PONTO DE INTERROGACAO"
* 380 PRINT "ENTAO TECLE CR"
  400 REM ESCOLHA COR DA BOLA
* 420 INPUT "QUAL A COR QUE DESEJA ?";BOLA
* 424 REM A COR DA BOLA ESTA ENTRE 1 E 15 ?
* 428 IF (BOLA>0) AND (BOLA < 16) THEN GOTO 460
* 432 PRINT "NAO ESTAVA ENTRE 1 E 15 !"
* 436 GOTO 420
  440 REM ACIONE MODO GRAFICO
  460 GR
  480 REM POSICAO INICIAL
  500 XANT = 20
* 510 YANT = 38
  520 REM MOVA BOLA PARA FRENTE E PARA TRAS
  540 XMOV = 1
* 545 REM MEXA BOLA PARA CIMA E PARA BAIXO
* 550 YMOV = 1
  560 REM NOVA POSICAO X
  580 XNOV = XANT + XMOV
  600 REM BOLA ESTA NA TELA ?
* 620 IF (XNOV >= 0) AND (XNOV < 40) THEN GOTO 686
  640 REM MOVA BOLA PARA ESQUERDA
  660 XMOV = -1 * XMOV
  680 GOTO 580
* 684 REM NOVA POSICAO Y
* 686 YNOV = YANT + YMOV
* 688 REM BOLA ESTA NA TELA ?
* 690 IF (YNOV >= 0) AND (YNOV < 40) THEN GOTO 720
* 692 REM MOVA BOLA PARA CIMA
* 694 YMOV = -1 * YMOV
* 699 GOTO 686
  700 REM NOVA POSICAO DA BOLA
  720 COLOR = BOLA
* 740 PLOT XNOV,YNOV
  760 REM APAGUE POSICAO ANTERIOR DA BOLA
  780 COLOR = 0
* 800 PLOT XANT,YANT
  820 REM GUARDE POSICAO DA BOLA
  840 XANT = XNOV
* 850 YANT = YNOV
  860 REM VOLTE AO INICIO
  880 GOTO 580
```

Como você verá quando executar este programa, o resultado é um pouco repetitivo.

Você pode alterar o padrão do movimento mudando os valores de partida de XANT e YANT (linhas 500 e 510), mas apresentamos uma alteração que talvez será mais do seu agrado:

```
580 XNOV = XANT + XMOV * PDL (0) / 70
686 YNOV = YANT + YMOV * PDL (1) / 70
```

Para ver o que acontece, mexa nos controladores de jogo. Mais uma sugestão. Por quê não inserir mais um INPUT, atribuindo um valor a uma variável chamada FUNDO, no início do programa logo depois do GR? Encha a tela uma vez com a cor de fundo. Então para apagar a posição anterior da bola use:

```
780 COLOR = FUNDO
```

ou ainda

```
780 COLOR = FUNDO + 3
```

SALVE (SAVE) sua versão favorita do programa.

CRIANDO SONS

Sons como: cliques, tiques, toques e vários zumbidos são facilmente criados. Você pode produzir sons em seu computador batendo nele, arranhando-o com seus dedos, deixando-o cair, e etc, mas os sons abrangidos por este manual são produzidos programando-o para tal. Portanto, vá para um lugar sosseyado e tente estudar este capítulo.

Para construir um programa gerador de sons você precisará da fórmula mágica:

```
150 SOM = PEEK (-16336)
```

Não existe uma explicação simples para essa forma. O número -16336 está relacionado com o endereço de memória do alto-falante do computador e é estabelecido pela parte eletrônica do mesmo. Você terá que obter esse número quando precisar dele.

PEEK obtém um código numérico armazenado em certo local de memória. Na maioria das locações (NT. qualquer lugar onde podem ser armazenados dados), PEEK apenas fornece um valor numérico, mas em algumas, como por exemplo -16336, pode originar um clique. Toda vez que o computador executar esta instrução, o computador produzirá um "clique". Execute o programa e escute atentamente seu computador. Agora acrescente esta linha:

```
160 GOTO 150
```

e execute seu programa. Nenhum problema de audição agora! Para fazer com que seu EXATO emita bips durante um período limitado de tempo, acrescente instruções como:

```
140 FOR BIP = 1 TO 100
160 NEXT BIP
```

Um tom é gerado por uma rápida sequência de cliques. Qualquer programa que use PEEK (-16336) repetidamente, gerará algum tipo de barulho. Já que é tão complicado digitar -16336, inserimos outra instrução que nos permitirá substituir o símbolo por outro mais fácil de digitar:

```
100 S = -16336
```

Para produzir um clique bonito e ressonante, altere a linha 150 para:

```
150 SOM = PEEK (S) - PEEK (S) + PEEK (S) - PEEK (S) +
    PEEK (S) - PEEK (S)
```

Diferentes números de PEEKs n instrucão, oroduzirão cliques diferentes. Experimente executar lgumas variacões. Para obter sons mais semelhantes a zumbidos, coloque uma de suas variáveis em um loop. Em geral, quanto mais rápido for o loop, mais alto será o tom.

Agora, para usar estes sons, carregue novamente o segundo programa da bola saltitante chamado. Tente acrescentar um som de "pulo" a cada vez que a bola bate na parede. Uma solução possível, é fornecida mais adiante, mas primeiro tente descobrir por si mesmo. (dica: ocorre um salto a cada vez que XMOV ou YMOV muda de valor).

SOM PARA BOLA SALTITANTE

Eis aqui um meio de tornar audíveis os saltos da bola. Acrescente as seguintes linhas ao programa.

```
240 REM AJUSTE S PARA O ENDERECO DO ALTO-FALANTE
260 S = -16336
665 REM SOM DE SALTO
665 FOR B = 1 TO 5
670 SALTA = PEEK(S)-PEEK(S)+PEEK(S)-PEEK(S)
675 NEXT B
695 REM SOM DE SALTO
696 FOR B = 1 TO 5
697 SALTA = PEEK(S) - PEEK(S) + PEEK(S) - PEEK(S)
698 NEXT B
```

Agora experimente seus próprios sons. Por quê não fazer cada parede ter um som diferente?

NOTAS MAIS ALTAS

Para se conseguir tons ainda mais altos, pode introduzir-se mais uma característica do CCE BASIC. É possível colocar mais de uma instrução na mesma linha. Digite este programa de uma única linha:

```
NEW
50 S = PEEK (-16336): GOTO 50
```

Em qualquer programa onde queiramos ter mais de uma instrução por linha, pode-se usar os dois pontos (:). Entretanto, apenas a primeira instrução da linha tem um número de linha, de modo que você pode desviar apenas para a primeira instrução com um GOTO. Agora acrescente:

```
40 FOR PAUSA = 1 TO 2500: NEXT PAUSA
```

As vantagens de instruções múltiplas com um número de linha comum são:

1- as instruções são executadas mais rapidamente. (Isso é uma vantagem apenas se você precisar de maior rapidez);

2- uma parte maior do seu programa vai caber na tela;

3- seu trabalho de digitação é menor;

4- você pode agrupar instruções que realizam uma função coletivamente, tais como a linha 40;

5- requer menos memória. Essa é uma vantagem apenas se você estiver processando com falta de memória e o computador lhe enviar uma mensagem: ?FALTA MEMORIA ou ?PROGRAMA GRANDE enquanto você estiver inserindo um programa.

Há também algumas desvantagens:

1- é mais difícil interpretar o programa;

2- é mais difícil modificar ou corrigir o programa;

3- você pode apenas desviar o programa para o início de cada linha;

4- é muito desencorajador digitar uma longa instrucão múltipla. Apenas para ter como resultado (quando rodar o programa) a mensagem ?SINTAX ERRO e fazendo com que seja necessário redigitar toda a instrucão.

NUMEROS ALEATORIOS

Digite esse programa:

```
NEW
100 PRINT RND (1)
110 GOTO 100
```

Na linha 100 RND é abreviação de Random (aleatório). A função RND fornece números ao acaso. Interrompa o programa. Os números gerados por esse programa são frações decimais aleatórias entre 0 e 1. Altere a linha 100 para:

```
100 PRINT RND (6)
```

Execute o programa novamente e pare-o. Interessante!! Os números aleatórios continuam entre 0 e 1. Para não se esquecer, anote o último número que foi gerado. Agora, altere novamente a linha 100:

```
100 PRINT RND (0)
```

e execute-o. Pare o programa e compare suas anotacões com o que apareceu agora na tela. RND (0) fornece o último número aleatório que foi gerado.

Frações decimais, aleatórias, entre 0 e 1 podem ser um pouco estranhas. Frequentemente números inteiros (como 3, 6 e 10) são mais fáceis de serem usados. Para conseguir números inteiros aleatórios de 0 a 9, temos que acrescentar algumas linhas no programa. Digite:

```
NEW
90 REM ATRIBUI NUMEROS RND A X
100 X = RND (1)
110 REM MULTIPLICA X POR 10
120 X = X * 10
130 REM ELIMINA A PARTE FRACIONARIA
140 X = INT (X)
150 PRINT X
160 GOTO 100
```

A linha 140 introduz a função INT. A instrução INT(X) fornece o maior número inteiro menor ou igual a X. Por exemplo, se o valor for de X é 3,6754, então o valor INT (X) será igual a 3. Os parênteses depois do INT podem conter qualquer expressão aritmética ou uma variável numérica.

Agora execute o programa. Funcionou como você esperava? Para alterar o programa de modo que gere números de um a dez ao invés de zero a nove, simplesmente some um ao valor de X, acrescentando esta linha ao programa:

```
145 X = X + 1
```

A princípio o programa pode parecer um pouco complicado. Para ver o que acontece passo a passo, você pode acrescentar uma série de instruções PRINT.

Modifique seu programa para ficar igual a este:

```
90 REM ATRIBUE NUMERO ALEATORIO A X
100 X = RND(1) : PRINT "X=RND(X) ", X
105 PRINT
110 REM MULTIPLICA X POR 10
120 X = X * 10 : PRINT "X = X * 10  , X
125 PRINT
130 REM ELIMINA A FRACAO
140 X=INT(X):PRINT "X = INT(X) ", X
150 REM SOMA 1 AO VALOR DE X
160 X = X + 1 : PRINT "X = X + 1"
170 PRINT X
175 PRINT
180 FOR PAUSA = 1 TO 2000 : NEXT PAUSA
190 GOTO 100
```

Processe esse programa e veja o que acontece.

Você poderá condensar esse programa para uma única linha.

```
100 PRINT INT (10 * RND (1)) + 1 : GOTO 100
```

Você sabe como funciona a linha 100 ?

SIMULANDO UM PAR DE DADOS

Você pode usar o que aprendeu a respeito de números aleatórios, para fazer um programa que simule um par de dados.

```
NEW
100 PRINT "DADO BRANCO"
110 PRINT INT (6 * RND(1)) + 1
120 PRINT "DADO VERMELHO"
130 PRINT INT (6 * RND(1)) + 1
```

Este programa gera números inteiros aleatórios que vão de 1 a 6 para cada dado. Para jogar os dados rode o programa novamente. Você é capaz de fazer um programa de jogo que use esses "dados"? Experimente.

Tente escrever um programa de uma única linha que gere números ao acaso de 1 a 50. De 0 a 25. Invente seus próprios números. Lembre-se de que se não quiser gerar zeros, deverá acrescentar (1) ao número aleatório.

Veja um meio colorido de usar números inteiros ao acaso.

```
NEW
200 GR
210 REM ESCOLHER COR ALEATORIA
220 COLOR = INT(16 * RND (1))
230 REM ESCOLHER UM PONTO ALEATORIO
240 X = INT ( 40 * RND(1))
250 Y = INT ( 40 * RND(1))
260 REM DESENHE O PONTO ALEATORIO
270 PLOT X, Y
280 REM FACA NOVAMENTE
290 GOTO 220
```

Experimente usar RND em outros programas. Você é capaz de fazer um programa que trace linhas coloridas aleatórias na tela?

SUB-ROTINAS

Imagine que exista um jogo para o qual você precise de uma peça que se pareça com um cavalo azul, de patas cor de laranja e cara branca. Veja o programa que desenha tal figura:

```
NEW
1000 REM PROGRAMA PARA DESENHAR CAVALO AZUL COM CARA
     BRANCA E PATAS COR DE LARANJA.
1010 GR
1020 COLOR = 7 : REM AZUL CLARO
1030 PLOT 15,15
1040 HLIN 15,17 AT 16
1050 COLOR = 9 : REM LARANJA
1060 PLOT 15,17
1070 PLOT 17,17
1090 COLOR = 15 : REM BRANCO
2000 PLOT 14,15
```

Não há nada de errado com esse programa. Ele desenha um cavalo azul com patas cor de laranja e cara branca. Agora, suponha que você precise desenhar um outro cavalo em outro lugar da tela. Poderia redigitar este programa com novos valores de X e Y, mas seria uma amolação. Deve haver um meio de usar o mesmo programa para colocar uma figura em qualquer lugar da tela, sem ter que redigitar o programa a cada vez.

A chave para conseguir isso, começa com a observação de que você pode mover um ponto que está nas coordenadas (A,B) para a direita, aumentando o valor da primeira coordenada, A, neste caso.

Por exemplo: o ponto (4,17) se movimentará 10 colunas para a direita se você acrescentar 10 a primeira coordenada, fazendo o ponto (14,17).

Da mesma maneira, um ponto se move para a esquerda se você diminuir a primeira coordenada (ou acrescentar um valor negativo). Uma experiência simples irá lhe mostrar que somando ou subtraindo um valor à segunda coordenada, o ponto subirá ou descerá respectivamente.

Tendo esses fatos em mente, você pode redigitar o programa para colocar o cavalo em quase qualquer ponto da tela (X,Y). Por quê quase qualquer ponto? Porque se você escolher 1 ponto central na extremidade da tela, o cavalo sairá fora da mesma e isso

poderá lhe dar uma mensagem ?VALOR ILEGAL EM 1030 (ou em outro número de linha). Veja agora um programa melhorado:

```
NEW
1000 REM COLOQUE UM CAVALO EM QUALQUER LUGAR DA TELA
1010 COLOR = 7 : REM AZUL CLARO
1020 PLOT X, Y - 1
1030 HLIN X, X + 2 AT Y
1040 COLOR = 9 : REM LARANJA
1050 PLOT X, Y + 1
1060 PLOT X + 2, Y + 1
1070 COLOR = 15 : REM BRANCO
1080 PLOT X - 1, Y - 1
```

Note que o GR não foi incluido. Queremos usar essa parte do programa para colocar vários cavalos na tela. Um GR aqui, limparia a tela antes que cada cavalo novo fosse desenhado. Este programa não pode ser processado na forma em que está. Primeiro você precisa ajustar o modo gráfico e escolher X e Y. Uma primeira tentativa para usar o programa cavalo poderia ser:

```
20 GR
30 REM COORDENADAS DO PRIMEIRO CAVALO
40 X = 12
50 Y = 5
```

Se você executar essa programa, conseguirá um cavalo no local desejado, mas o programa vai acabar aí. Queremos colocar dois cavalos na tela. O que aconteceria se pudesse escrever:

```
60 DESENHE O CAVALO DEPOIS VA PARA LINHA 70
70 REM SEGUNDO CAVALO
80 X = 33
90 Y = 2
100 DESENHE O CAVALO E DEPOIS PARE
```

Não seria fácil e bonito? Você sabe que o computador não pode ler instruções estranhas nas linhas 60 e 100. Entretanto, pode ler:

```
GOSUB 1000
```

Em CCE BASIC, um programa como o que começa na linha 1000 é chamado Sub-rotina. GOSUB 1000 diz ao computador para ir à Sub-rotina que começa na linha 1000 e executar as instruções. Também diz ao computador para quando tiver acabado a sub-rotina, para voltar à linha que segue a instrução GOSUB.

Ele sabe que a sub-rotina terminou, quando encontra uma instrução RETURN. Para transformar a parte que desenha cavalos do seu programa em uma sub-rotina, acrescente:

```
1090 RETURN
```

Agora você pode completar aquele programa.

```
20 GR
30 REM PRIMEIRO CAVALO
40 X = 12
50 Y = 35
60 GOSUB 1000
70 REM SEGUNDO CAVALO
80 X = 33
90 Y = 2
100 GOSUB 1000
```

Ao executar o programa, você receberá uma mensagem de erro:

```
?"RETURN" SEM "GOSUB" ERRO EM 1090
```

Fora isso, o programa parece que funcionou perfeitamente. Na realidade, acrescentou uma nova instrução do CCE BASIC: um comando para desenhar um cavalo. Agora você pode usar a instrução:

```
GOSUB 1000
```

para desenhar um cavalo em qualquer local da tela (X, Y) que você tenha escolhido.

VERIFICAÇÕES

A parte do programa, da linha 1000 à linha 1090 é chamada de sub-rotina ou subprograma. A parte do programa que vai da linha 20 à linha 100 é chamada de rotina principal.

Para poder verificar o fluxo do programa; ou seja, a sequência da execução, você pode usar uma característica especial chamada TRACE. Essa característica especial pode mostrar-lhe porque o computador exibiu uma mensagem de erro, quando foi executado o programa de desenho do cavalo. Acrescente esta linha ao programa:

```
TRACE
```

e, por um momento, cancele a linha 20.

Coloque o computador em modo texto e execute o programa. Os números que você vê na tela são os números de linha de cada instrução que está sendo executada. Você pode ver que o programa começa na linha 10 e continua através da rotina principal até a chamada da sub-rotina. Então executa-a voltando em seguida ao programa principal; executa novamente a sub-rotina e, não encontrando números de linha menores, vai à linha 1000 e mais uma vez executa a sub-rotina.

Aí é que acontece o problema. Entendeu agora a mensagem de erro? Para remediar o problema, acrescente esta nova linha ao programa:

```
110 END
```

Quando o programa chegar a linha 110 ele fará exatamente o que diz a linha (fim). Execute o programa mais uma vez. Não há mais mensagem de erro. Como você viu, TRACE é muito útil quando você tiver problemas com um programa. Se quiser usar a instrução TRACE em apenas uma parte do programa, poderá usar a instrução NOTRACE. Acrescente esta linha:

```
63 NOTRACE
```

e o programa será verificado apenas até executar a linha 63.

TRACE também pode ser usado no modo imediato. Simplesmente tecle:

```
TRACE

RUN
```

e o computador informará a sequência de execução do seu programa.

NOTA: Uma vez que tenha colocado um comando TRACE, seja no modo imediato ou como instrução de seu programa, seu programa será verificado a cada vez que você processá-lo, até o momento que encontre o comando NOTRACE em qualquer um dos dois modos de comando.

TORNANDO PROGRAMAS MAIS FÁCEIS

As sub-rotinas devem ser usadas entre outros motivos para que possíveis erros não surjam quando o programa estiver sendo executado. Um problema com nossa sub-rotina de desenhar cavalos é que o cavalo saia fora dos limites da tela.

Isso pode ser evitado por um conjunto de instruções tais como:

```
1012 IF X < 1 THEN X = 1
1014 IF X > 37 THEN X = 37
1016 IF Y < 1 THEN Y = 1
1018 IF Y > 38 THEN Y = 38
```

Por quê o valor máximo de Y deverá ser 38, enquanto o de X deve ser limitado a 37 ?

Se houver uma tentativa de localizar um cavalo fora da tela, o cavalo será deslocado para o ponto mais próximo dentro da mesma. Há outras estratégias possíveis, tais como emitindo uma mensagem de erro e interrompendo o programa. Entretanto, nossa escolha tem a vantagem de não parar o programa e você pode ver que algo está acontecendo.

Algumas vezes você pode querer alterar valores em uma sub-rotina de um programa, a cada vez que ela for executada. Por exemplo, um segundo jogador pode querer colocar uma peça,e esta seria um cavalo de cor diferente. Uma maneira de fazê-lo, seria digitar toda a sub-rotina novamente, com cores diferentes. Entretanto, vamos acostumar-nos a usar variáveis ao invés de números. A linha 1010 poderia ser feita assim:

```
1010 COLOR = CORPO
```

Analogamente, você poderia escrever:

```
1040 COLOR = PATAS
1070 COLOR = CARA
```

O programa principal ficaria assim:

```
20 GR
30 REM COR DO CAVALO DO PRIMEIRO JOGADOR
40 CORPO = 7 : REM AZUL CLARO
50 PATAS = 9 : REM LARANJA
60 CARA = 15 : REM BRANCO
70 REM POSICAO DO PRIMEIRO CAVALO
80 X = 15
90 Y = 30
100 GOSUB 1000
```

e assim por diante (tenha certeza de colocar uma instrução END antes de executar o programa). A cada vez que quiser desenhar um cavalo, você terá que digitar várias instruções. Por isto, vamos tentar melhorar isto.

Para facilitar ainda mais a programação, um truque sutil é ter uma sub-rotina que atribua cores aos cavalos de cada jogador e fazer com que cada uma dessas sub-rotinas chame-se por sua vez sub-rotina de colorir cavalos.

```
2000 REM:  DESENHA CAVALO  AZUL COM  PATAS LARANJA  E
           CARA BRANCA.
2010 CORPO = 7: REM AZUL CLARO
2020 PATA = 9: REM LARANJA
2030 CARA = 15: REM BRANCA
2040 GOSUB 1000
2050 RETURN
```

Agora, tudo o que precisa para colocar um cavalo azul com cara branca e patas cor de laranja na posição (10,11) é:

```
30 REM CAVALO DO PRIMEIRO JOGADOR
40 X = 10
50 Y = 11
60 GOSUB 2000
```

Para colocar um cavalo laranja em (19,2), tudo o que precisa é:

```
70 REM CAVALO DO SEGUNDO JOGADOR
80 X = 19
90 Y = 2
100 GOSUB 2500
```

Ambas sub-rotinas de colorir cavalos da página anterior, começando com as linhas 2000 e 2500, chamam outra sub-rotina que começa na linha 1000. Neste estágio as coisas começam a ficar bem eficientes.

Uma vez que você tenha feito uma boa sub-rotina, que verifique erros e que use variáveis compatíveis à rotina de chamada (que pode ser a rotina principal ou outra sub-rotina), poderá ligá-la a outras sub-rotinas. Isto torna as rotinas principais muito mais fáceis de serem feitas. Usando as três sub-rotinas é muito fácil fazer bonitos desenhos de cavalos.

Mas antes, experimente esta rotina prática:

```
3000 REM ESCOLHA ALEATORIAMENTE X,Y
3010 X = INT (RND (1) * 37)+1
3020 Y = INT (RND(1) * 38)+1
3030 RETURN
```

Agora a rotina principal fica assim:

```
10 REM ACIONAR O MODO GRAFICO
20 GR
30 REM ESCOLHA UM PONTO ALEATORIO
40 GOSUB 3000
50 REM COLOQUE ALI UM CAVALO AZUL
60 GOSUB 2000
70 REM ESCOLHA OUTRO PONTO ALEATORIO
80 GOSUB 3000
90 REM COLOQUE ALI UM CAVALO LARANJA
100 GOSUB 2500
110 REM REPITA TUDO NOVAMENTE
120 GOTO 30
```

Assim é que deve se parecer uma rotina principal, se você é um bom programador, aparecem principalmente REMs e GOSUBs. O trabalho deve ser feito em sub-rotinas relativamente curtas, fáceis de serem digitadas e completas. Sinta-se a vontade para usar TRACE para ver como se comporta este programa exemplo.

GRÁFICOS DE ALTA RESOLUÇÃO

Os gráficos que vínhamos usando até agora são conhecidos por gráficos de Baixa Resolução. Neste capítulo você aprenderá a usar outro tipo de gráficos chamados "Gráficos de Alta Resolução". Este novo tipo de gráfico permite que você desenhe com muito mais detalhe do que seria possível com um quadriculado de 40 colunas por 40 linhas. A nova tela de gráficos de Alta Resolução é de 280 por 160 divisões ou pontos. As coordenadas horizontais começam com 0, a esquerda da tela, e terminam com 279 a direita. Do mesmo modo as coordenadas verticais vão de 0 no topo da tela a 159 na base da mesma.

Não é difícil entender gráficos de Alta Resolução. Normalmente os comandos de gráficos de Alta Resolução são os mesmos usados com gráficos de Baixa Resolução correspondentes, exceto pela adição de um H (High-resolution). Será útil que você, neste segmento, tenha um bom conhecimento de gráficos de Baixa Resolução.

Digite:

```
HGR
```

para entrar no modo gráfico de Alta Resolução. Esse comando faz com que a tela fique preta, deixando na base quatro linhas para o

texto. Esta área de texto pode ser comecada pelo HGR com o uso de coordenadas verticais (192 é o máximo), mas estes pontos não são mostrados na tela. Se o cursor não estiver na área do texto, pressione algumas vezes a tecla CR até que ele apareça na base da tela.

Os gráficos de Alta Resolução são realmente maravilhosos, más para usá-los terá que fazer alguns sacrifícios. Neste novo modo, há menos cores disponíveis. As cores de gráficos de Alta Resolução vão de Ø (preto) a 7 (branco).

As cores são:

Ø Preto 1
1 Verde
2 Violeta
3 Branco 1
4 Preto 2
5 Laranja
6 Azul
7 Branco 2

Essas cores poderão variar entre os aparelhos de TV e, de acordo com suas posições na tela. Por exemplo, a cor número 3 será azul se a coordenada horizontal for impar, verde se a coordenada horizontal for par e branco somente se estes dois pontos forem colocados um ao lado do outro. Isso se deve ao método de funcionamento dos aparelhos de TV domésticos.

NOTA: Traçar linhas em gráficos de Alta Resolução é mais fácil que em Gráficos de Baixa Resolução. Você simplesmente tem que traçar de um ponto da tela a outro ponto. Para fazer isto, acompanhando a borda superior da tela, digite:

HPLOT Ø, Ø TO 279,Ø

Se quiser traçar uma linha no canto direito do ponto 279,Ø até o 279,159, canto inferior da tela, tudo que tem a fazer é digitar

HPLOT TO 279,159

e aparece uma linha ao longo do canto direito da tela. Ao usar esta última instrução, a nova linha tem seu ponto de partida e cor determinados a partir do último ponto traçado (mesmo se tiver emitido um novo comando HCOLOR depois de traçá-lo). Você pode encadear estes comandos e traçar várias linhas em uma única instrução.

Limpe a tela com HGR e experimente isto:

```
HPLOT Ø, Ø TO 279,Ø TO 279,159 TO Ø,159 TO Ø,Ø
```

Deve aparecer uma linha contornando a tela. Se não houver esta linha, verifique primeiro se a digitação estava correta. Se a linha ainda não apareceu ou não é continua, mude o HCOLOR e tente novamente. Algumas partes da tela só aparecam, em algumas TVs, em determinadas cores.

Não só é fácil traçar linhas verticais e horizontais, com linhas diagonais em gráficos de Alta Resolução. Para se tracar uma diagonal do canto superior esquerdo da tela ao canto inferior direito, digite apenas:

```
HPLOT Ø,Ø TO 279,159
```

Caso você tenha "joystick", pratique desenhando linnas em HGR em cores e comprimentos diferentes. Aqui está um programa que transformará seu computador em uma tela de desenhos.

```
NEW
200 HGR
210 HCOLOR = 3
220 X = PDL (Ø)
230 Y = PDL (1)
240 IF Y > 159 THEN Y = 159
250 HPLOT X, Y
260 GOTO 220
```

Foi incluida a linna 240 porque a funcão PDL pode gerar valores até 255 e a coordenada Y ficaria fora dos limites de tela. Execute este programa.

Ele funciona, mas poderia ser melhorado se fosse mais fácil tracar uma linha ininterrupta. Os espaços entre os pontos traçados, nem sempre são desejados. Você pode melhorar esse programa digitando as seguintes linnas:

```
220 GOSUB 1000
230 HPLOT X, Y
240 GOSUB 1000
250 HPLOT TO X, Y
260 GOTO 240
1000 X = PDL (Ø) / .913
1010 Y = PDL (1) / 1.6
1020 RETURN
```

Liste o programa e verifique cuidadosamente se você digitou corretamente todas as instruções. Aqui você tem a sequência em que este programa é executado.

O EXATO é colocado no modo HGR e a cor das linhas é definida, então o programa executa a sub-rotina começando na linha 1000. A sub-rotina determina o valor das coordenadas X e Y. Os controladores de jogo atingem um valor máximo de 255. Devido ao fato dos gráficos de Alta Resolução utilizarem coordenadas horizontais de 0 a 279, os valores gerados pelo PDL(0) são divididos por 0.913 para ampliar seu alcance à toda largura da tela. Da mesma forma, os valores fornecidos por PDL (1) são divididos por 1.6 para diminuir seu alcance aos limites das coordenadas de HGR: 0 a 159. Assim como nos gráficos de Baixa Resolução, as coordenadas realmente usadas são os valores inteiros menores ou iguais aos valores gerados. A linha 1020 retorna a execução à rotina principal e, então a linha 230 desenha o ponto (X, Y). Em seguida o programa volta à sub-rotina, atribue novos valores a X e Y e retorna à linha 250 da rotina principal onde uma linha muito curta é tracada do antigo ponto X, Y ao atual ponto (X, Y). O GOTO da linha 260 repete todo o programa, exceto as instruções HGR e HCOLOR, obtendo novos valores para X e Y a dependendo da posição dos controladores de jogo e depois tracando a nova posição X, Y e assim por diante. Para parar o programa use CTRL C.

Há uma razão para tracar linhas ao invés de tracar cada ponto separadamente. Leva um certo tempo para tracar um ponto e quando o comoutador faz isto, um de cada vez, nem sempre é capaz de manter o mesmo ritmo dos controladores de jogo. Essa é a razão para os espacos entre os pontos quando você movia rapidamente os controladores de jogo no primeiro programa de desenho. Isto é remediado, tracando-se uma pequena linha até cada uma das novas posições especificadas pelos controlador de jogo: tracando-se uma linha entre um ponto e outro, todos os pontos intermediários são automaticamente tracados e muito mais rapidamente do que se fossem tracados um a um.

Agora salve (SAVE) o programa e então execute-o.

Aqui está um programa que faz belos desenhos na tela.

```
NEW
90 HOME
100 VTAB 24: REM  MOVA CURSOR PARA  LINHA DA BASE  DA
    TELA
120 HGR: REM ACIONA MODO ALTA RESOLUCAO
140 A = RND (1) * 279: REM FIXAR UM "A" PARA CENTRO
160 B = RND (1) * 159: REM FIXAR UM "B" PARA CENTRO
180 N = INT (RND (1) * 4) +2: REM FIXAR O TAMANHO
    DE PASSO
```

```
100 HTAB 15: PRINT "PASSO="; N;
120 FOR X  = 0 TO  278 STEP N:  REM INCREMENTANDO  OS
    VALORES DE A COM O PASSO N
240 FOR S = 0 TO 1 : REM 2 LINHAS, DE X E X + 1
260 HCOLOR  =  7  *  S  :  REM  PRIMEIRA  LINHA PRETA,
    PROXIMA BRANCA
280 REM DESENHE  UMA LINHA  ATRAVES DO "CENTRO" ATE O
    OUTRO LADO
300 HPLOT X + S,0 TO A,B TO 279 - X - S,159
320 NEXT S,X
340 FOR Y = 0 TO 158 STEP N: REM INCREMENTANDO OS
    VALORES DE B COM PASSO N VALORES B
360 FOR S = 0 TO 1 : REM 2 LINHAS, DE B E B + 1
380 HCOLOR  =  7  *  S  : REM  PRIMEIRA  LINHA  PRETA,
    PROXIMA BRANCA
400 REM: DESENHE UMA LINHA ATRAVES DO "CENTRO" ATE  O
    LADO OPOSTO
420 HPLOT 279, Y + S TO A, B TO 0,159 - Y - S
440 NEXT S,Y
460 FOR PAUSA = 1 TO 1500: NEXT PAUSA: REM ATRASO
480 GOTO 120
```

Este é um programa bastante longo, digite-o cuidadosamente e liste-o em partes (por exemplo LIST 0,320) para verificar se tudo está correto. Execute-o apenas quando tiver certeza de que está tudo certo.

Como dá para ver neste programa, nas linhas 320 e 440 uma instrução NEXT pode servir para mais de uma instrução FOR. Tenha cuidado em colocar as próximas variáveis NEXT na ordem correta, de modo a evitar loops cruzados.

Para interromper este programa use CTRL C e depois digite TEXT.

Você consegue imaginar alguns meios de alterar o programa? Após ter armazenado esta versão no seu cassete ou disquete, experimente mudar aleatóriamente o valor de HCOLOR. Talvez desenhar linhas laranja e azuis, ou apenas linhas azuis.

Há muito mais coisas a respeito de gráficos de Alta Resolução do que é apresentado aqui. Quando estiver seguro quanto a utilização dos comandos do HGR apresentados aqui, procure maiores informações nos capítulos 8 e 9 do Manual de Referência de Programação em CCE BASIC.

CAPITULO 5

CADEIAS E MATRIZES

TRABALHANDO COM CADEIAS

(CADEIA = É uma sequência linear ou cadeia de caracteres.)

Você gostaria de ver seu nome escrito de trás para frente? Até agora trabalhamos com números e gráficos, mas os computadores também são capazes de manipular letras e símbolos. Seu computador pode lidar com um único caráter ou com uma cadeia (string) inteira de caracteres de uma só vez. Isso parece bastante natural, já que nós, humanos, geralmente lidamos com punhados de caracteres. As variáveis alfanuméricas (string), tal como as variáveis numéricas, têm nomes. Os nomes das cadeias obedecem as mesmas regras das variáveis numéricas, exceto que, acabam com o símbolo do dólar ($). Aqui mostraremos alguns exemplos:

```
NOME$
A$
FRASE$
```

A variável A é diferente da variável A$ e ambas podem ser usadas no mesmo programa.

Se você deseja que a cadeia chamada NOME$ (pronunciada "NOME- DOLAR") contenha as letras "JOSE DA SILVA" você pode digitar:

```
NOME$ = "JOSE DA SILVA"
```

note que os caracteres que você coloca em uma cadeia tem que estar entre aspas. A instrução:

```
PRINT NOME$
```

exibirá o conteúdo da variável NOME$, nesse caso, um nome comum no Brasil. Assim, se você tem uma cadeia de caracteres que usa frequentemente, você poderá armazená-la em uma variável com um nome pequeno.

Há várias outras instruções em CCE BASIC que manipulam cadeia. Suponha que você queira saber o comprimento de uma cadeia (quantos caracteres contêm). Você pode digitar:

```
PRINT LEN ("JOSE DA SILVA")
```

ou pode digitar a instrução equivalente:

```
PRINT LEN (NOME$)
```

e o EXATO irá exibir o tamanho (comprimento) da cadeia. Nesse caso 13. Note que os espaços são contados como caracteres.

O número de caracteres em uma cadeia pode variar de Ø a 255. Se você tentar usar mais de 255 caracteres obterá uma mensagem: ?SINTAX ERRO ou ?"STRING" LONGA. Uma cadeia com Ø caracteres é chamada de CADEIA NULA.

Recorra ao Manual de Referência para programação do CCE BASIC para maiores informações sobre CADEIA NULA. Em alguns casos, você poderá desejar exibir apenas uma parte de NOME$. Para isso, você pode utilizar umas funções muito úteis:

LEFT$, RIGHT$ e MID$

Se, por exemplo, você quiser exibir as primeiras 5 letras de NOME$, você digitará:

PRINT LEFT$ (NOME$,5)

e aparecerá JOSE na tela. Se você digitar:

PRINT RIGHT$ (NOME$,5)

o resultado será SILVA.

No início da execução de um programa, as variáveis numéricas equivalem a zero e as cadeias são nulas. Portanto, você deve atribuir valores para as cadeias do seu programa. Aqui mostraremos o uso das funções LEN e LEFT$.

```
NEW
9Ø NOME$ = "JOSE DA SIL "
1ØØ FOR N = 1 TO LEN (NOME$)
11Ø PRINT LEFT$ (NOME$, N)
12Ø NEXT N
```

execute este programa. O comando RIGHT$ é semelhante ao comando LEFT$, com a exceção de que usa os caracteres mais a direita da CADEIA. Agora digite outro programa substituindo LEFT$ por RIGHT$. O que acontece quando você roda esse programa ?

Se você quiser usar os caracteres do meio da cadeia, ao invés do começo ou do fim, é preciso usar a função MID$. Digite:

PRINT MID$ (NOME$, 9)

Seu computador responde com:

SILVA

Já que o "S" é o 9º caráter na cadeia. Agora digite o seguinte programa:

```
NEW
190 NOME$ = "JOSE DA SILVA"
200 FOR N = 1 TO LEN(NOME$)
210 PRINT MID$ (NOME$,N)
220 NEXT N
```

Você conseguiu o que queria quando o programa foi processado? Suponhamos que você queira exibir apenas o "E DA S" da cadeia chamada NOME$. Para isso, é necessário acrescentar mais um argumento à função MID$.

```
PRINT MID$ (NOME$, 4, 6)
```

O primeiro número (4) especifica o caráter, o qual o computador deverá começar a exibir. O segundo número (6) diz quantos caracteres devem ser exibidos. Assim, o EXATO interpreta a instrução como: ache o 4º caráter de NOME$ e exiba seis caracteres, a partir dele. Altere a linha 210 do programa anterior para:

```
210 PRINT MID$ (NOME$,4,6) e processe-o.
```

Não vá adiante nesse manual, até ter testado exaustivamente as funções LEFT$, RIGHT$ e MID$. Considere o seguinte programa:

```
NEW
200 A$ = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
210 PRINT
220 PRINT "DIGITE UM NUMERO DE 1 A 26"; LEN (A$);" "
230 PRINT "E EU LHE DIREI QUE LETRA DO ALFABETO ESTA
     NAQUELA POSICAO "
240 INPUT P
250 IF P > LEN(A$) OR P < 1 THEN GOTO 210
260 PRINT
270 PRINT MID$(A$,P,1);" E O NUMERO DA LETRA "; P;
    " NO ALFABETO"
280 PRINT: PRINT
290 PRINT "DIGITE UMA LETRA E EU LHE DIREI"
300 INPUT "ONDE ELA ESTA NO ALFABETO"; X$
310 FOR N = 1 TO LEN (A$)
320 IF MID$(A$,N,1) = X$ THEN GOTO 370
330 NEXT N
340 PRINT
350 PRINT "ISTO NAO E UMA LETRA DO ALFABETO": PRINT
360 GOTO 290
370 PRINT
380 PRINT X$ " E A LETRA NUMERO"; N ". DO ALFABETO"
390 PRINT
400 GOTO 210
```

Este programa ilustra algumas práticas comuns de programação. Note como ele encontra uma posição de um caráter na cadeia. Este método de usar um loop para esquadrinhar uma cadeia, uma posição por vez, é muito comum. Note também a função dos espaços em branco nas ordens PRINT. O que aconteceria com a saída do programa sem esses espaços em branco? Finalmente observe que os limites do programa são sempre estabelecidos por LEN (A$) e não pelo número real de caracteres no alfabeto. Isso permite que o programa funcione mesmo se você especificar um "alfabeto" diferente na linha 200. Tente e verá. Você pode substituir uma cadeia por outra com uma instrução de troca assim:

```
X$ = A$
```

Esta instrução copia o conteúdo de A$ em X$. Entretanto, você não pode usar notação parcial da cadeia do lado esquerdo de uma instrução de substituição. Por exemplo, a instrução:

```
MID$ (X$, 3, 3) = "XYZ"
```

é ilegal, mas a instrução

```
X$ = MID$ (A$, 24, 3)
```

está certa. Apenas uma variável pode ser deixada do lado esquerdo de uma instrução de substituição.

Você ainda quer ver seu nome escrito de trás para frente? O próximo programa fará exatamente isso:

```
NEW
100 REM PROGRAMA PARA ESCREVER NOME AO CONTRARIO
110 INPUT "ESCREVA SEU NOME E EU O MOSTRAREI AO
    CONTRARIO"; N$
120 REM - INVERTA A ORDEM DAS LETRAS
130 FOR T = LEN(N$) TO 1 STEP - 1
140 R$ = R$ + (MID$(N$,T,1))
150 NEXT T
160 PRINT : PRINT "O SEU NOME AO CONTRARIO E ";R$
170 PRINT : PRINT
180 GOTO 110
```

Rode este programa, testando vários nomes diferentes. Após este programa ter sido executado algumas vezes, você notará que há alguma coisa errada. A chave do problema está na linha 140. Se, por exemplo, seu nome é EDGARD. Quando solicitado você o digitará e assim ajustaria N$ para EDGARD e R$ para DRAGDE. Talvez sua amiga CERES esteja aí e também queira ter seu nome escrito de trás para frente. Na próxima vez que o programa fizesse uma solicitação ela digitaria seu nome ajustando N$ para CERES. A linha 140 então ajustaria R$ ao R$ antigo, mais N$ escrito ao contrário, em outras palavras: DRAGDESERE.

É necessário, portanto, um comando que reajuste as cadeias deixando-as nulas de modo que R$ possa ser novamente preenchido com caracteres após cada GOTO na linha 11Ø.

Felizmente, tal comando existe em CCE BASIC. É o comando CLEAR. Ele associa a todas as variáveis de todos os tamanhos, forma e cor o valor zero. Some essa linha a seu programa,

```
175 CLEAR
```

rode novamente o programa.

O comando CLEAR também pode ser usado em execução imediata. Digite:

```
N = 254
PRINT N
```

Agora digite:

```
CLEAR
```

e então

```
PRINT N
```

O EXATO forneceu Ø como sendo o valor de N ?

O QUE É CONCATENAÇÃO ?

É possível acrescentar uma outra cadeia ao final de uma cadeia já existente, usando o sinal de mais (+). Esse processo é conhecido por concatenação. Tente no seu EXATO :

```
C$ = "BOM DIA"
D$ = C$ + " " + "EDUARDO"
PRINT D$
```

Seu EXATO responderá com:

```
BOM DIA EDUARDO
```

A concatenação é especialmente útil se você quiser desmanchar uma cadeia e depois juntá-la novamente com pequenas modificações. Por exemplo, se você queria criar uma nova cadeia semelhante ao D$ com a diferença de que os espaços entre as palavras seriam substituídas por traços, você poderia digitar:

```
E$ = RIGHT$ (D$, 7) + " - " + LEF$ (D$, 3)  + " - " +
     MID$ (D$, 5, 3).

PRINT E$
```

e na tela aparecerá :

```
EDUARDO-BOM-DIA
```

Eis um programa que usa concatenação.

```
NEW
100 INPUT "ESCREVA A PRIMEIRA METADE DE UMA FRASE";
    METADE$
110 INPUT "AGORA A OUTRA METADE DA FRASE";
    OUTRAMETADE$
120 TUDO$ = METADE$ + OUTRAMETADE$
130 PRINT
140 PRINT TUDO$
150 PRINT:PRINT:PRINT:GOTO 100
```

Assim é que se faz concatenação.

MAIS FUNCÕES COM CADEIA

Podem-se formar cadeia com quase qualquer tipos de caráter incluindo números. Entretanto, em uma instrução PRINT, os caracteres entre as aspas de uma cadeia, não podem ser matematicamente interpretados, mesmo que sejam números. Para ver o que acontece, digite:

```
C$ = "123"
PRINT C$ + 7
```

Seu EXATO exibirá:

```
?TIPO INCOMPAT ERRO
```

e não conseguirá lidar com essa instrução. Precisamos da ajuda do comando VAL (da palavra VAlor) que é uma função que resolverá este problema. A função VAL retorna o valor do conteúdo de uma cadeia ao contrário de seu conteúdo real. Digite:

```
PRINT C$
```

e depois digite:

```
PRINT VAL (C$)
```

Aparentemente, ambos comandos fornecem o mesmo resultado: entretanto, as aparências podem enganar. Você já sabe que se digitar:

```
PRINT C$ + 5
```

seu EXATO responderá com

```
?TIPO INCOMPAT ERRO
```

Experimente digitar:

```
PRINT VAL (C$) + 5
```

e na tela aparecerá o número 128

Note que o nome da variável sequencial é o argumento da função VAL e, tem necessariamente que estar entre parênteses.

E se você quiser colocar resultado de C$ menos 21, em uma variável numérica, Simples, é só digitar:

```
Q = VAL (C$) - 21
```

Agora digite:

```
PRINT Q
```

e veja o que obteve. O conteúdo de Q é o que você esperava ? Você pode usar VAL para somar o valor numérico de duas cadeias diferentes. Para testar isso, crie uma nova cadeia.

```
K$ = "12"
```

e então digite:

```
P = VAL (C$) + VAL (K$)

PRINT P
```

Tente com cadeias diferentes, incluindo cadeias que comecem ou acabem com letras.

Às vezes, é necessário transformar uma variável numérica em uma cadeia. A função STR$, que trabalha de modo contrário à função VAL, pode ser usada para se conseguir essa alteração. Suponha que você queira transformar a variável numérica P numa cadeia.

Então digite:

```
P$ = STR$ (P)
PRINT P$
```

Você verá como funciona STR$. Eis um programa que usa STR$ e VAL.

```
300 INPUT "DIGITE UM NUMERO DE 1 A 999999999"; N$
310 N = VAL (N$)
320 IF N < 1 OR N > 999999999 THEN GOTO 300
330 N$ = STR$ (N)
340 FOR T = LEN (N$) TO 1 STEP - 1
350 P$ = P$ + MID$ (N$, T, 1)
360 NEXT T
370 PRINT : PRINT "ORIGINAL ",N$
380 PRINT : PRINT "REVERSO ", P$
390 P = VAL (P$)
400 PRINT : PRINT "ORIGINAL + REVERSO = "; N + P
410 CLEAR
420 PRINT : PRINT : GOTO 310
```

Entendeu como funciona o programa ? Porque há virgulas nas linhas 370 e 380 ? Tente apagar a linha 330 para ver o que acontece. As primeiras quatro linhas deste programa demonstram os primeiros passos para fazer uma rotina de entrada realmente à prova de erros.

Veja quais as entradas que ainda p[illegible]m o programa e então invente as etapas para evitar que [illegible]as interrompam o processamento.

INTRODUZINDO MATRIZES (ARRAYS)

NOTA: MATRIZ é o mesmo que arranjo sistemático de elementos em formato de tabela.

Nesse capítulo sobre matrizes, usaremos exemplos matemáticos, mas eles são exemplos de matemática recreativa e não necessitam nada além de aritmética elementar.

Os arranjos permitem-nos selecionar qualquer elemento numa tabela de números. O poder de programação que eles nos dão, compensam mais do que o esforço com a experimentação e estudos que são necessários para nos familiarizarmos com eles.

Uma matriz é uma tabela de números. O nome desta segue as mesmas regras que qualquer nome legal de variável, por exemplo A. O nome da matriz A é distinto, e diferente da variável numérica A.

Para criar uma matriz, primeiro você tem que definir para o computador o número máximo de elementos que você quer que ela acomode.

Para isso, use uma instrução DIM (significa DIMensionar). Os elementos numa matriz são numerados a partir de Ø, assim para DIMensionar uma matriz chamada A que terá um máximo de 16 elementos, digite:

```
DIM A(15)
```

A instrução DIM acima nos deu 16 novas variáveis. Elas se comportam exatamente como as variáveis que você já estudou. Elas são:

```
A (Ø)
A (1)
A (2)
e assim por diante até
A (15)
```

Embora você possa achar estranha sua digitação, elas podem ser usadas exatamente da mesma forma que outras variáveis. A instrução:

```
A (9) = 45 + A (12)
```

é perfeitamente correta. O número entre parênteses é chamado de índice ou subscrito; a notação A(12) é lida "A DOZE". O índice pode ser uma expressão matemática ou pode ser representado por uma variável. A variável A pode também ser chamada de variável indexada.

Digite o seguinte programa. Ele ilustra o uso de variáveis no índice e exibe os conteúdos de cada elemento da matriz.

```
100 REM DIMENSAO DA MATRIZ CHAMADA DIAS PARA CONTER
    7 NUMEROS
110 DIM DIAS (6)
120 REM PREENCHA A MATRIZ
130 FOR NUM = Ø TO 6
140 DIAS (NUM) = NUM + 1
150 NEXT NUM
160 REM EXIBE OS ELEMENTOS DA MATRIZ
170 FOR I = Ø TO 6
180 PRINT "DIAS (" ; I ; ") = " ; DIAS (I)
190 NEXT I
```

Se uma matriz é usada em um programa antes de ser DIMensionada, o EXATO reserva um espaço de 11 elementos (subscritos de Ø a 1Ø). Entretanto, é uma boa prática de programação fazer o DIMensionamento em todas as matrizes.

Suponha que você quer escrever um programa que gera números embaralhados de 1 a 8. Para conseguir isso, você precisa manipular as tabelas de dados. Isso é justamente o tipo de coisa para que as matrizes são excelentes. O programa abaixo demonstra isso.

```
NEW
200 REM DIMENSIONE A MATRIZ
210 DIM COPO(8)
220 REM PREENCHA A MATRIZ
230 FOR I = 1 TO 8
240 COPO(I) = I
250 NEXT I
260 REM MISTURE A MATRIZ E ESCOLHA CADA ELEMENTO
270 FOR VINHO = 1 TO 8
280 REM ESCOLHA ALGUM OUTRO ELEMENTO
290 LEITE = INT( RND(1) * 8) + 1
300 REM LEITE E DIFERENTE DE VINHO?
310 REM SE NAO E TENTE NOVAMENTE"
320 IF LEITE = VINHO THEN GOTO 280
330 REM FACA INTERCAMBIO COPO (VINHO) E COPO (LEITE)
340 EXTRA = COPO (VINHO):COPO (VINHO) = COPO (LEITE):
    COPO (LEITE) = EXTRA
350 NEXT VINHO
360 REM EXIBA CONTEUDOS DA MATRIZ
370 FOR C = 1 TO 8
380 PRINT COPO(C)
390 NEXT C
```

Entendeu como funciona esse programa ? Primeiro ele preenche uma matriz com números e então mistura os elementos da mesma.

Note que você não tem que começar a preencher a matriz a partir do zero. Eis uma descrição do que algumas linhas mais enganosas do programa fazem. Das linhas 23Ø a 25Ø preencha a matriz e atribua a cada elemento da matriz, um número correspondente a seu número de matriz (COPO(1) = 1, etc). A linha 27Ø coloca na nova variável VINHO números de 1 a 8. A linha 28Ø coloca na variável LEITE números inteiros aleatórios de 1 a 8. Então, a linha 3ØØ assegura-se de que em nenhuma hora o valor de VINHO é igual ao valor de LEITE. Os conteúdos das variáveis COPO(VINHO) e COPO(LEITE) são trocados na linha 31Ø. Finalmente, a matriz é exibida das linhas 33Ø a 35Ø.

As trocas que acontecem na linha 310 podem ser imaginadas assim: vamos dizer que temos 2 copos, 1 copo de vinho e o outro de leite. Desculpe, houve um engano. O leite está no copo de vinho e o vinho no copo de leite. Ainda bem que dispomos de um copo extra! Podemos colocar o leite no copo extra e então por o vinho no copo de vinho e finalmente colocar o leite no copo de leite. Agora ambas as bebidas foram colocadas em seus copos apropriados.

MENSAGENS DE ERRO EM MATRIZES

Há algumas mensagens de erro que poderão ocorrer enquanto estiver fazendo programa com o uso de matrizes.

?REDIMENSIONAR

Esta mensagem acontece quando uma matriz é dimensionada mais de uma vez no mesmo programa. Frequentemente ocorre este erro quando uma dimensão foi criada, e posteriormente foi acrescida ao programa, uma nova instrução DIM, com o mesmo nome.

?INDICE ILEGAL

Se é feita uma tentativa para usar um elemento da matriz que está fora da dimensão da matriz, ocorrerá esta mensagem de erro.

Por exemplo, se A foi dimensionado em 25 para a instrução DIM A(25), referindo-se ao elemento A(52) ou qualquer outro elemento cujo subscrito é menor que zero ou maior que 25, dará um

?INDICE ILEGAL

?VALOR ILEGAL

Você receberá essa mensagem se tentar usar um número negativo como subscrito de matriz.

Estas são algumas das maneiras como você pode usar matrizes. As matrizes usadas aqui são todas dimensionadas. Você também pode usar matrizes que contenham duas ou mais dimensões. Para maiores informações sobre matrizes, recorra ao Manual de Referência de Programação do CCE BASIC.

CONCLUSÃO

Esse manual apresentou o essencial do CCE BASIC. Se você repassar as lições, refazendo seus próprios programas com as instruções que apresentamos aqui, você solidificará consideravelmente os seus conhecimentos. O EXATO tem muitas outras habilidades e, ao dominar as apresentadas neste manual, você terá conhecimentos para explorá-las.

APÊNDICES

APÊNDICE A

RESUMO DOS COMANDOS

NOTA: Este apêndice contém tanto os comandos do CDOS como os do CCE BASIC.

A seguir há um resumo de comandos que podem ser usados na linguagem de programação CCE BASIC. Para maiores informações a respeito destes comandos, leia o Manual de Referência para programação em CCE BASIC.

TECLAS DE MOVIMENTOS HORIZONTAIS

As teclas marcadas com flechas apontando para a direita e para a esquerda, são usadas para editar programas CCE BASIC. A flecha que aponta para a direita, move o cursor para a direita. Ao fazê-lo, cada caráter sobre o qual ela passa, é interpretado como se você tivesse digitando. A tecla com a flecha apontando para a esquerda, move o cursor para a esquerda. Ao movê-lo, o caráter sobre o qual passa, é anulado da linha do programa que você estiver digitando.

CALL - 151

Faz com que apareça um asterisco de solicitação de entrada, indicando que o EXATO está respondendo a sua linguagem nativa, chamada linguagem de máquina. Se você não for um programador, experimente. Provavelmente não precisará de se comando.

CATALOG

Mostra na tela uma lista de todos os arquivos do disquete na unidade de disco especificado ou não. À esquerda do nome do arquivo são indicados o tipo de arquivo e o número de setores ocupados por ele. Os tipos de arquivo são:

I - programa em CCE INTEGER;
A - programa em CCE BASIC;
T - arquivo Texto: foi criado por um comando WRITE;
B - arquivo Binário: é a cópia de parte da memória do EXATO em determinado momento.

O comando CATALOG é um comando CDOS (Sistema de Operação de Discos CCE) e não um comando CCE BASIC.

CLEAR

Zera todas as variáveis numéricas e faz com que as cadeias fiquem nulas.

COLOR = 12

Determina a cor usada para o traçado em modo gráfico de Baixa Resolução (GR). Nesse exemplo, a cor é ajustada para o verde. O comando GR, ajusta a cor a zero.

Os nomes das cores e os números que lhes são associados são:

0 PRETO	8 MARROM
1 MAGENTA	9 LARANJA
2 AZUL-ESCURO	10 CINZA
3 PURPURA	11 ROSA
4 VERDE-ESCURO	12 VERDE CLARO
5 CINZA	13 AMARELO
6 AZUL-MEDIO	14 AGUA MARINHA
7 AZUL-CLARO	15 BRANCO

CONT

Se a execução do programa foi interrompida por STOP, END ou CTRL C, o comando CONT faz com que a execução continue na próxima instrução (como GOSUB) e não no próximo número de linha.

CONT não poderá ser usado, caso você tenha:

a - modificado, acrescentado ou cancelado uma linha do programa;

b - obtido uma mensagem de erro que parou a execução do programa.

CTRL

C

Pode ser usado para interromper a listagem ou o processamento de um programa. Também pode ser usado para interromper um INPUT (entrada), se este foi o primeiro caráter a ser inserido. O INPUT (entrada) não é interrompido até ser pressionada a tecla CR.

CTRL

X

Diz ao computador para ignorar a linha que está sendo digitada, sem cancelar qualquer linha anterior com o mesmo número de linha. No final da linha a ser ignorada, aparece uma barra transversal (\).

DEL 23, 56

Apaga as linhas do programa especificadas nos parâmetros da instrução. No exemplo serão canceladas as linhas de 23 a 56. Para cancelar (DELete) uma única linha, digamos a linha 350, use a forma DEL350,350 ou, simplesmente digite o número de linha e então tecle CR.

DIM NOME$ (50)

Quando é executada uma instrução DIM, ela deixa um espaço reservado para a matriz especificada com subscritos variando de 0 até o subscrito fornecido. No exemplo NOME$(50) serão concedidas além dela mesma, mais 51 cadeias de qualquer comprimento. Se um elemento de matriz for usado num programa antes de ter sido DIMensionado, para cada dimensão no elemento subscrito, serão concedidos um máximo de 10 subscritos.

END

Faz com que pare a execução de um programa e retorna o controle ao usuário. Não é exibida nenhuma mensagem.

ESC I, ESC J, ESC K ou ESC M

A tecla ESCape pode ser usada juntamente com as teclas de letras I, J, K ou M para mover o cursor sem afetar os caracteres que são sobrepostos pelo cursor. Para mover o cursor primeiro tecle ESC para entrar o modo de edição. Então digite a tecla adequada uma vez para cada movimento na direção desejada, não sendo necessário pressionar ESC cada vez. A tecla REPT pode ser usada para apressar os movimentos, apertando-se uma das teclas de movimento (J,K,I,M) ao mesmo tempo que a tecla REPT.

COMANDO	MOVE O CURSOR UM ESPAÇO
	para cima para esquerda para direita para baixo

FLASH

Ajusta o modo do vídeo para ficar piscando, assim o caráter exibido pelo computador é alternadamente mostrado na tela em branco sobre fundo preto e, então revertido a preto sobre fundo branco. Use o comando NORMAL para voltar ao display com letras brancas sobre fundo preto.

```
FOR W = 1 TO 20 ..... NEXT W
FOR Q = 2 TO -3 STEP - 2 ..... NEXT Q
FOR Z = 3 TO 4 STEP 3 ..... NEXT Z
```

Permite que você faça um "loop" para realizar quaisquer instruções entre o comando FOR (no alto do "loop") e o comando NEXT (na base do "loop") um número especificado de vezes. No primeiro exemplo, a variável W conta quantas vezes vai executar as instruções; as instruções dentro do "loop" serão executadas para W igual a 1,2,3...20 então o "loop" acaba (com W=21) e a instrução após NEXT W é executada. O segundo exemplo ilustra como indicar que, o tamanho do STEP (passo) deverá ser diferente de 1. A verificação ocorre no final do "loop", para que no terceiro exemplo as instruções dentro do loop sejam imediatamente executadas.

GOSUB 250

Faz com que o programa pule para a linha indicada (no exemplo 250). E quando é executada uma instrução RETURN o programa retorna para a instrução imediatamente seguinte ao GOSUB mais recentemente executado.

GOTO 250

Faz com que o programa desvie para a linha indicada (250 no exemplo).

GR

Aciona o modo gráfico de Baixa Resolução (40 por 40) deixando 4 linhas para o texto na base da tela. É limpa, ficando escura; o cursor é movido para a janela de texto e a cor é ajustada a 0 (preto).

HCOLOR = 4

Ajusta com a cor especificada dos gráficos de Alta Resolução (HGR). Os nomes das cores e seus valores associados são:

0 PRETO	4 PRETO
1 VERDE	5 LARANJA
2 VIOLETA	6 AZUL
3 BRANCO	7 BRANCO

HGR

Ajusta o modo gráfico Alta Resolução (280 por 160) deixando na base da tela 4 linhas reservadas para o texto. A tela é limpa ficado negra e é acionada a página 1 da memória. Nem o HCOLOR nem a área de texto são afetados quando se executa HGR. O cursor não se move para a janela de texto.

HGR2

Ajusta o modo gráfico de Alta Resolução de tela inteira (280 por 192). A tela é limpa e é acionada a página 2 da memória. A memória de texto não é afetada.

HLIN 10, 20 AT 30

Usada para desenhar linhas horizontais em modo gráfico de Baixa Resolução usando a cor mais recentemente especificada por COLOR. A origem (X = 0 e Y =0) encontra-se no ponto superior mais à esquerda da tela. No exemplo, a linha é desenhada de X = 10 a X = 20 em Y = 30. Outra maneira de dizer isto é: a linha é desenhada do ponto 10 ao ponto 20 da linha 30.

HOME

Movimenta o cursor para a posição superior esquerda da tela e limpa todo o texto da mesma.

```
HPLOT 10, 20
HPLOT 30, 40 TO 50, 60
HPLOT TO 70, 80
```

Traça pontos e linhas em modo gráfico de Alta Resolução, usando a cor mais recentemente especificada em HCOLOR. A origem é o ponto superior esquerdo X = 0, Y = 0. O primeiro exemplo define um ponto em X = 10, Y = 20. O segundo exemplo traça uma linha do ponto X = 30, Y = 40 para o ponto em X = 50, Y = 60. O terceiro exemplo traça uma linha do último ponto definido ao ponto em X = 70, Y = 80: usando a cor do último ponto definido, não necessariamente do último HCOLOR.

```
HTAB 23
```

Move o cursor para a esquerda ou direita da coluna em que está (1 a 40). No exemplo, o cursor será posicionado na coluna 23.

```
IF IDADE < 18 THEN A = 0 : B = 1 : C = 2
IF AN$ = "SIM" THEN GOTO 100
IF N < MAX THEN GOTO 25.
```

Se a expressão que se segue ao IF é avaliada como verdadeira (isto é diferente de zero); então serão executadas as instruções após o THEN na mesma linha. Caso contrário, qualquer instrução após o THEN é ignorada e a execução passa para a instrução contida no próximo número de linha do programa. As expressões CADEIAs são avaliadas de acordo com a ordem alfabética.

```
INPUT A
INPUT "MODO DE DIGITAR : IDADE,NOME ";B,C$
```

No primeiro exemplo, INPUT exibe um ponto de interrogação e espera que o usuário digite um número, que será atribuído à variável numérica A. No segundo exemplo, INPUT exibe a mensagem opcional, entre aspas, exatamente como mostrado e, espera o usuário digitar um número (o que será atribuido à variável B), depois uma virgula, e então a cadeia de entrada (que será atribuido à variável cadeia C$). Entradas múltiplas em INPUT, podem ser separadas por virgula ou CR.

INT(NUM)

Fornece o maior número inteiro menor ou igual ao número existente. No exemplo se NUM é 2,369, então o 2 será fornecido se NUM é -45,123345, então será fornecido -46.

INVERSE

Ajustar o modo inverso de vídeo, de modo que os displays sejam feitos com letras pretas sobre o fundo branco. Use o comando NORMAL para conseguir letras brancas em fundo preto.

LEFT$ ("CCE BASIC", 3)

Fornece o número especificado de caracteres situados mais à esquerda da cadeia. No exemplo, será obtido: CCE (os 3 caracteres mais a esquerda).

LEN ("UMA FRUTA POR DIA")
LEN (A$)

Fornece o número de caracteres de uma cadeia, entre 0 e 255. No primeiro exemplo, a resposta será 17.

A = 23,567
A$ = "DELICIOSA"

A variável à esquerda do igual (=) receberá o valor da variável à direita, sejam numéricas ou cadeias.

LIST
LIST 200, 3000
LIST 200 - 3000

O primeiro exemplo faz com que todo o programa seja mostrado na tela; o 2º e 3º exemplos fazem com que sejam, da mesma maneira, apresentadas as linhas de 200 a 3000. Para listar desde o começo do programa, até a linha 200, use LIST - 200; para listar da linha 200 até o fim do programa, use LIST 200 -. A listagem é interrompida permanentemente por CTRL C e o comando CONT não pode ser usado. Para parar temporariamente o programa em algum ponto da listagem, use CTRL S. Use CTRL S novamente para continuar a listagem.

LOAD

Carrega um programa em fita cassete para a memória do computador. Não é dado sinal de solicitação de entrada: o usuário precisa rebobinar a fita e pressionar "play", no gravador, antes de (carregar) LOAD. Toca um BIP quando a informação é encontrada na fita. Um segundo BIP toca quando o programa estiver carregado e retornará o caráter de solicitação de entrada. O processo LOAD só pode ser interrompido por um [RESET].

LOAD CIDADE MARAVILHOSA

Tenta localizar um arquivo de programa com o nome de CIDADE MARAVILHOSA no disquete na unidade de disco especificada. Se o programa for encontrado, será carregado na memória do EXATO. LOAD apaga qualquer programa do computador antes de colocar um novo programa na memória. Esse comando, quando seguido do nome do arquivo, é um comando CDOS.

```
MID$ ("UMA PERA POR DIA",4)
MID$ ("UMA PERA POR DIA",4,9)
```

Fornece a cadeia especificada. No primeiro exemplo, serão fornecidos do 4º ao último caráter da cadeia: PERA POR DIA. No 2º exemplo, 9 caracteres, começando com o 4º caráter da cadeia serão fornecidos: PERA POR.

NEW

Cancela o programa e todas variáveis, que estiverem na memória.

NEXT

Veja capítulo 3 referente a FOR...TO...STEP

NORMAL

Ajusta o modo de vídeo para as habituais letras brancas sobre fundo preto tanto para entrada como para saída.

NOTRACE

Desliga o modo TRACE. Veja TRACE.

POL (1)

Gera o valor atual, um número de Ø a 255, indicado no controlador de jogo. São válidos números de Ø a 3 de controlador de jogos, o número entre parênteses.

PLOT 10,20

Coloca um ponto no local desejado, no modo gráfico de Baixa Resolução (GR). No exemplo o ponto estará em X = 10, Y = 20. A cor do ponto é determinada pelo último valor de COLOR que, usado, caso não tenha sido previamente especificado, será Ø (preto).

PRINT
PRINT A$; "X="; X

O primeiro exemplo faz com que uma linha seja pulada na tela. Itens de uma lista a ser exibida devem ser separados por vírgula (,) se cada um deve ser apresentado em um campo tabulado. Os itens deverão ser separados por ponto e vírgula, caso devam ser exibidos um ao lado do outro sem espaço entre eles. Se A$ contém "CORES" e X é 3; o segundo exemplo exibirá: cores X=3.

REM (isto é uma observação)

Permite que seja inserido um texto em um programa, em forma de comentário. Este texto será ignorado pelo EXATO.

REPT

Se você pressiona a tecla de repetição, chamada REPT, enquanto digita qualquer outra tecla, o mesmo será repetido enquanto você segurá-la.

RETURN

Retorna para a instrução imediatamente seguinte ao último GOSUB executado.

RIGHT$ ("O EXATO", 5)
RIGHT$ (G$,2)

Fornece o número especificado de caracteres situados à direita da cadeia. No primeiro exemplo, será fornecido : EXATO (os 5 caracteres mais a direita).

RND (5)

Fornece um número real aleatório maior ou igual a zero e menor do que um. RND (Ø) Fornece um número aleatório mais recentemente gerado. Cada argumento negativo gera um número aleatório próprio e que é o mesmo sempre que RND for usado com aquele argumento. RNDs subsequentes, com argumentos positivos seguirão sempre uma determinada e repetida sequência. Cada vez que RND é usado com qualquer argumento positivo é gerado um novo número aleatório de Ø a 1, a não ser que seja parte de uma sequência de números aleatórios iniciada por um argumento negativo.

RUN 5ØØ

Limpa todas as variáveis, indicadores e "pilhas". E começa a execução no número de linha indicado (no exemplo, 5ØØ). Se não é especificado um número de linha, a execução começa na linha de número mais baixo do programa.

RUN CIDADE MARAVILHOSA

Carrega o arquivo chamado "CIDADE MARAVILHOSA" e depois executa o programa carregado. Quando seguido do nome de um arquivo, o RUN é um comando CDOS, e não um comando CCE BASIC.

SAVE

Armazena em fita cassete o programa que atualmente está na memória. Não é dado o sinal de solicitação de entrada ou qualquer outro sinal. O usuário precisa apertar as teclas "RECORD" e "PLAY" do gravador, antes de ser executado o comando SAVE. Este comando não verifica se os botões do gravador estão pressionados, mas emite um BIP no início e no fim da gravação.

SAVE PAGAMENTO

Armazena o arquivo atualmente na memória. Se no disquete não for encontrado o arquivo chamado PAGAMENTO, é criado um novo arquivo neste disquete e o programa atualmente na memória é armazenado com o nome de arquivo a ele atribuido. Se o disquete contém um arquivo com o nome especificado do arquivo, na mesma linguagem, o conteúdo original do arquivo é perdido e o programa atual é armazenado. Não é dada qualquer advertência. SAVE seguido do nome do arquivo é um comando CDOS.

STR (12.45)

Gera uma cadeia que representa o valor do argumento (valor entre parênteses). No exemplo é fornecida a sequência "12.45".

TAB (23)

Precisa ser usada numa instrução PRINT; o argumento tem que estar entre 0 e 255 e entre parênteses. Se o argumento for maior do que o valor da posição atual do cursor, então TAB movimentará o cursor para a posição de exibição especificada, contando a partir da margem esquerda da atual linha do cursor. Se o argumento for menor que o valor da posição atual do cursor, então o cursor não se moverá. TAB (0) coloca o cursor na posição 256.

TEXT

Coloca a tela no modo usual de texto, com 40 caracteres por linha e 24 linhas.

TRACE

Faz com que o número de linha de cada instrução apareça na tela quando a mesma é executada. TRACE não é desativado por RUN, CLEAR, NEW, DEL OU RESET. NOTRACE desliga TRACE.

VAL ("-3.7E3ATO")

Tenta interpretar uma sequência, até o primeiro caráter não numérico, como um número inteiro ou real, e fornece o valor de tal número. Se não existir nenhum número antes do primeiro caráter não numérico, o valor fornecido será um 0. No exemplo, é devolvido, -3700.0.

VLIN 10,20 AT 30

Traça uma linha vertical na cor indicada pela última instrução COLOR, no modo gráfico de Baixa Resolução. A linha é tracada na linha indicada pelo terceiro argumento. No exemplo, a linha é tracada de Y=10 a Y=20 em X=30.

VTAB (5)

Move o cursor na tela para a linha especificada no argumento. A linha superior é a linha 1, a inferior é a 24. VTAB moverá o cursor para cima ou para baixo, mas não para a direita ou esquerda.

APÊNDICE B

PALAVRAS RESERVADAS EM CCE BASIC

A lista abaixo contém TODAS as palavras reservadas em CCE BASIC. Embora a maioria destas palavras não apareçam em outro lugar deste manual, a lista é útil como um guia para denominar variáveis. Para descobrir como usar um ou mais desses comandos, veja o Manual de Referência de Programação do CCE BASIC.

&	GET	NEW	SAVE
	GOSUB	NEXT	SCALE=
ABS	GOTO	NORMAL	SCRN(
AND	GR	NOT	SGN
ASC		NOTRACE	SHLOAD
AT	HCOLOR=		SIN
ATN	HGR	ON	SPC(
	HGR2	ONERR	SPEED=
CALL	HIMEM:	OR	SQR
CHR$	HLIN		STEP
CLEAR	HOME	PDL	STOP
COLOR=	HPLOT	PEEK	STORE
CONT	HTAB	PLOT	STR$
COS		POKE	
	IF	POP	TAB(
DATA	IN#	POS	TAN
DEF	INPUT	PRINT	TEXT
DEL	INT	PR#	THEN
DIM	INVERSE		TO
DRAW		READ	TRACE
	LEFT$	RECALL	
END	LEN	REM	UR
EXP	LET	RESTORE	
	LIST	RESUME	VAL
FLASH	LOAD	RETURN	VLIN
FN	LOG	RIGHT$	VTAB
FOR	LOMEM:	RND	
		ROT=	WAIT
FRE	MID$	RUN	
			XPLOT
			XDRAW

O CCE BASIC utiliza estas palavras. Cada uma gasta apenas um byte de armazenagem de programa. Todos os outros caracteres de palavras não reservadas ocupam um byte cada um no armazenamento de programa na memória.

O símbolo "%" serve para uso interno do computador. Não é um comando CCE BASIC. Esse símbolo, quando executado como instrução, causa um salto incondicional para a locação $DF5.

XPLOT é uma palavra reservada que não corresponde a um comando CCE BASIC de uso comum.

Algumas palavras reservadas são reconhecidas pelo CCE BASIC apenas em determinados contextos:

COLOR, HCOLOR, SCALE, SPEED e ROT

São analisados como palavras reservadas apenas se o próximo caráter não espaçado é um sinal de igual (=). Isto é de pouca valia no caso de COLOR e HCOLOR, já que a palavra reservada é incluida ou de qualquer modo previne seu uso em nomes de variáveis.

SCRN, SPC e TAB

São analisados como palavras reservadas apenas se o próximo caráter não espaçado é um parêntese esquerdo (().

HIMEM: precisa ser seguido dos dois pontos (:) para ser analisado como palavra reservada.

LOMEM: também requer os dois (:) para ser analisado como palavra reservada.

ATN: é analisado como palavra reservada, apenas se não houver espaço entre o T e o N. Se houver o espaço, será analisada a palavra reservada AT, ao invés de ATN.

TO é analisada como palavra reservada a menos que precedida por A e haja um espaço entre o T e o O. Caso isto ocorra, será analisada a palavra reservada AT, ao invés de TO.

Às vezes são usados parentèses para se contornar palavras reservadas. Por exemplo:

```
quando digitamos:    100 FOR A = LOFT OR CAT  TO 15
quando listamos :    100 FOR A = LOF TO RC AT TO 15
entretanto,

quando digitamos:    100 FOR A = (LOFT) OR (CAT) TO 15
quando listamos :    100 FOR A = (LOFT) OR (C AT) TO 15
```

APÊNDICE C

CARACTERÍSTICAS DE EDIÇÃO

SETA PARA ESQUERDA E DIREITA

A tecla com a seta que aponta para a esquerda, [<-] também conhecida por tecla de retrocesso, move o cursor para trás (esquerda) um espaço, anulando o caráter sobre o qual passou. Se você não apertou [CR] ao final da última linha que digitou, a tecla de retrocesso afeta apenas os caracteres desta linha.

Digitando a tecla com a flecha apontando para a direita, [->] também chamada de tecla de avanço, faz com que o cursor avance para a frente (direita), rebatendo o caráter sobre o qual passar. Se você rebater uma linha com a tecla de correção, então aperte [CR] . O computador comporta-se como se você tivesse redigitado a linha.

Você pode fazer com que o cursor mova-se mais rapidamente apertando a tecla [REPT] , enquanto aperta uma das teclas com flecha.

MOVIMENTOS SIMPLES DE CURSOR

As teclas [ESC] , [I] , [J] , [K] e [M] são usadas para movimentar o cursor sem afetar qualquer um dos caracteres da tela. Imagine flechas desenhadas nas teclas das letras conforme indicado.

Apertando [ESC] , você entra no modo de edição. Uma vez estando no modo de edição, apertando uma das teclas de letras acima mencionadas, fará com que o cursor se mova um caráter na direção da seta correspondente. Você pode usar essas setas para mover o cursor para qualquer lugar da tela.

Para movimentos mais rápidos do cursor, aperte uma dessas teclas e então pressione a tecla [REPT] . O cursor se movimentará enquanto ambas teclas estiverem pressionadas. Se o cursor alcançar o topo da tela, parará. Se o cursor atingir a base da tela, também parará e as linhas começarão a subir.

Se alcançar a extremidade direita, o cursor desaparecerá e reaparecerá na borda esquerda da próxima linha. Caso atinja a borda esquerda, reaparecerá uma linha acima da qual se encontrava, mas na extremidade direita. Para retornar ao modo normal, aperte uma vez a barra de espaço.

APAGANDO LINHAS DO PROGRAMA

Um método fácil de cancelar uma linha de um orograma é simplesmente digitar o número de linha que se deseja apagar e então pressionar [CR]. Se você tiver mais de uma linha para cancelar, você poderá usar o comando DELete (cancele). Por exemplo, para cancelar as linhas entre 100 e 200, você digitaria:

```
DEL 100, 200
```

Todas as linhas do programa, entre 100 e 200 inclusive, seriam então canceladas.

Digitando [CTRL] [X] cancelará a linha que você está digitando no momento. Isso é útil quando você perceber que cometeu um erro antes de ter pressionado [CR].

Se você estiver em modo de edição, poderá limpar a tela, digitando simplesmente [ESC] [SHIFT] [F].

O computador voltará ao modo normal quando teclar [CR].

O comando HOME também limpará a tela.

Digite simplesmente:

```
HOME
```

e o cursor se posicionará no canto superior esquerdo da tela.

Também é possível limpar apenas porções da tela. Para se limpar de um ponto da tela até o fim da mesma, entre em modo de edição pressionando [ESC]. Use então as características de movimentos simples de cursor para mover o cursor até o primeiro caráter que deseja limpar. Aperte a tecla [F] e desaparecerão todos os caracteres deste ponto até o final da tela. Para limpar os caracteres até o final de uma linha, você precisará primeiro estar em modo de edição, então mova o cursor até o primeiro caráter a ser apagado e pressione [E]. Em ambos casos, o EXATO voltará ao modo normal depois de ter executado do comando.

CARACTERíSTICAS DE EDIÇÃO

- Entrar no modo edição	Tecle [ESC]
- Sair do modo de edição	Tecle barra de espaço
- Mover o Cursor	Tecle [ESC] e depois [I], [J], [K] ou [M]
- Apagar um caráter	Tecle [<-]
- Avançar um caráter	Tecle [->]
- Limpe do cursor até o final da linha	Tecle [ESC] e então [E]
- Limpar do cursor até o final da tela	Tecle [ESC] então [F]
- Limpar a tela toda	Tecle [ESC] e então [SHIFT] [F]
- Parar a listagem	Tecle [CTRL] e [S]
- Recomecar a listagem	Tecle [CTRL] e [S]

APÊNDICE D

MENSAGENS DE ERRO

Todas as mensagens de erro que podem ser geradas em CCE BASIC são enumeradas aqui, juntamente com suas descrições.

Veja o Manual de Referência para Programação do CCE BASIC para maiores informações sobre mensagens de erro não apresentadas neste manual.

Depois de acontecer um erro, o CCE BASIC retorna ao nível de comando, conforme indicado pelo caráter de solicitação de entrada e pelo cursor.

As variáveis e o texto do programa permanecem intactos, mas o programa não pode continuar e todos GOSUBs e FORs (contadores de "loops") são anulados.

Quando ocorre um erro na instrução de execução imediata, não é exibido qualquer número de linha.

Formatos de Mensagens de Erro:

Instruções de execução imediata: ? XX ERRO

Instruções de execução adiada: ? XX ERRO EM YY

Em ambos exemplos acima "XX" é a mensagem de erro específico, "YY" é o número de linha da instrução de execução adiada onde ocorreu o erro. Os erros em instruções de execução adiada não são detectados até que a instrução tenha sido executada.

A seguir, vêm os possíveis erros de código e seus significados:

? "CONT" ILEGAL

Tentativa de continuar um programa: quando não havia nenhum programa, depois que ocorreu um erro, ou quando uma linha foi apagada ou acrescentada ao programa.

? DIVISAO POR ZERO

Dividir por zero. é um erro.

? FORMULA COMPLEXA

Foram executadas mais de duas instruções na forma IF "XX" THEN em uma mesma instrução.

? DIRETO ILEGAL

Como comando de execução imediata, não podem ser usados: INPUT, DEF FN, GET ou DATA.

? VALOR ILEGAL

O parâmetro passado a uma função matemática ou cadeia, estava fora dos limites. Os erros de valor ilegal podem ocorrer devido a:

a - parâmetro negativo (ex: A(-1) = 0);

b - usando LOG com argumento negativo ou igual a zero;

c - usando SQR com argumento negativo ou igual zero;

d - A ^ B sendo A negativo e B um número não inteiro;

e - uso de MID$, LEFT$, RIGHT$, WAIT, PEEK, POKE, TAB, SPC, ON...GOTO ou qualquer uma das funções gráficas com um argumento impróprio.

?"NEXT" SEM "FOR"

Numa instrução NEXT à variável não correspondeu a variável da instrução FOR que ainda estava em efeito; ou um NEXT sem nome que não correspondeu a qualquer FOR que ainda estava válido.

?FIM DE DATA

Foi executada uma instrução READ, mas todas as instruções DATA do programa já foram lidas. O programa tentou ler dados demais, ou dados insuficientes foram incluídos no programa.

?FALTA MEMORIA

Qualquer uma das afirmações seguintes podem causar erros: programa grande demais; com variáveis demais, loops FOR aninhados com mais de 10 níveis; GOSUBs aninhados com mais de 24 níveis; uma expressão complicada demais; parênteses aninhados com mais de 36 níveis; tentativa de ajustar LOMEM alto demais; tentativa de ajustar LOMEM mais baixo que o valor atual; tentativa de ajustar HIMEM baixo demais.

?S/ESPACO

Podem causar erros: se o resultado de um cálculo for grande demais para ser representado no formato de número do CCE BASIC e se ocorrer um erro devido o número ser pequeno demais (fluxo insuficiente); como resultado é exibido um zero e a execução prossegue sem exibir qualquer mensagem de erro.

?REDIMENSIONAR

Após ter sido dimensionada uma cadeia, foi encontrada outra instrução de dimensionamento para a mesma cadeia. Esse erro acontece frequentemente se foi dada a dimensão 10 a uma cadeia, por exemplo uma instrução como A(I)=3 e mais tarde no programa por um DIM A (100).

Essa mensagem de erro pode mostrar-se útil se você quiser descobrir em que linha do programa foi dimensionada determinada cadeia. Apenas insira a instrução de dimensão para aquela cadeia na primeira linha, processe o programa e o CCE BASIC lhe dirá onde está a instrução original de dimensionamento.

?"RETURN" SEM "GOSUB"

Uma instrução RETURN é encontrada sem o correspondente GOSUB ter sido executado.

?"STRING" LONGA

Foi feita uma tentativa de criar uma cadeia com mais de 255 caracteres de comprimento.

?INDICE ILEGAL

Foi feita uma tentativa de consultar um elemento de uma cadeia que é maior que o seu dimensionamento. Este erro pode ocorrer se for usado um valor errado de dimensão em uma referência, por exemplo: LET A(11)=Z quando A foi dimensionado com DIM A (2).

?SINTAX

Falta de parênteses numa expressão, caráter ilegal numa linha, pontuação errada, etc.

?TIPO INCOMPAT

O lado esquerdo de uma instrução de comando foi uma variável numérica e a parte da direita era cadeia, ou vice-versa; ou uma função que esperava um argumento da cadeia, recebeu um numérico ou vice-versa.

?LINHA INDEFINIDA

Foi feita uma tentativa de GOTO, GOSUB ou THEN a um número de linha inexistente.

?FUNCAO INDEF.

Refere-se a uma função criada pelo usuário e que ainda não foi definida.

Sugestões para o aperfeiçoamento do manual, bem como indicações de possíveis falhas encontradas no decorrer deste são bem vindas.

Favor enviá-las à:

CCE - Indústria e Comércio de Componentes Eletrônicos S/A
DIVISÃO DE INFORMATICA
Av. Otaviano Alves de Lima, 2724
CEP 02501 - São Paulo - SP.

NOME : _____

PROFISSAO : _____

CARGO : _____

TELEFONE : _____

MANUAL : _____

SUGESTÃO : _____

_____

_____

_____

_____

_____

_____

_____

_____