Aulas de Basic para linhas Sinclair, Apple e MSX

Carlos Alberto C. Abreu

Este livro versa sobre a Linguagem Basic.

As aulas estão acompanhadas de exercícios, ou sejam programas cujas respostas estão no final do livro, para melhor aprendizado.

O autor procura conduzir o usuário a entender a Linguagem Basic de forma simples e didática.

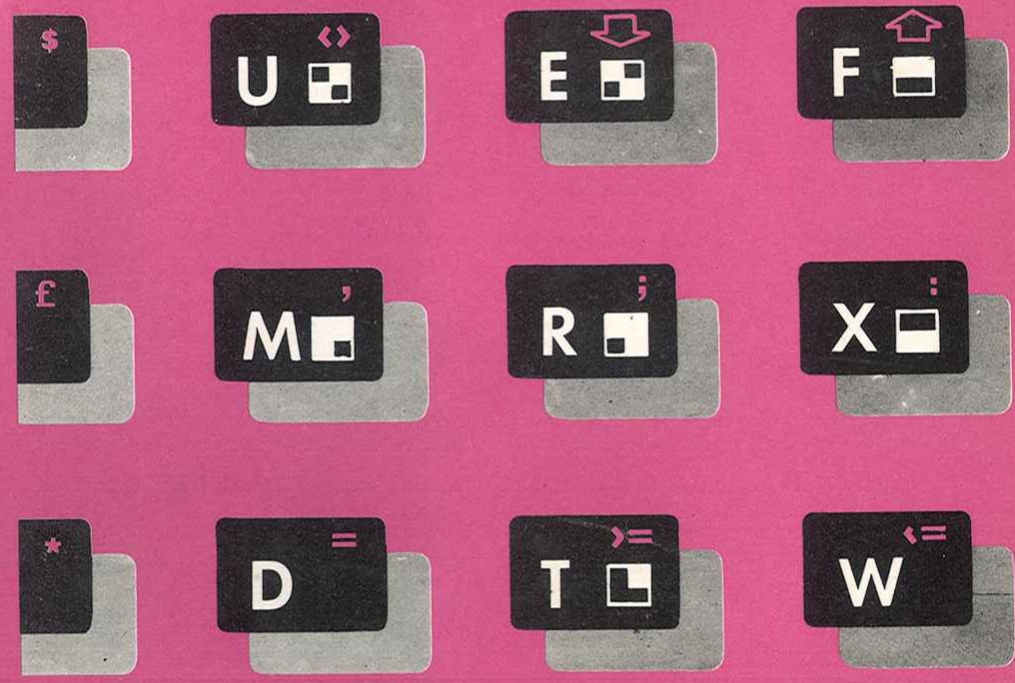
Edições
MICROKIT

ISBN 857065013-2

Aulas de Basic para linhas Sinclair, Apple e MSX ☆ Carlos Alberto C. Abreu

AULAS DE BASIC PARA LINHAS SINCLAIR, APPLE E MSX

Carlos Alberto C. Abreu



VOL. 1

Edições
MICROKIT

Este livro é uma nova versão do **Curso de Basic V. 1**. Devido as necessidades do mercado o autor cobriu as linhas Sinclair Apple e MSX.

Aulas de Basic para linhas Sinclair, Apple e MSX, é um livro didático e vai ajudá-lo a desenvolver-se na área de informática.

Não deixe de ler os outros livros das Edições Micro-Kit.

AULAS DE BASIC PARA LINHAS SINCLAIR APPLE E MSX

CARLOS ALBERTO C. ABREU

MESTRE EM CIÊNCIAS

EM ENGENHARIA DE SISTEMAS

E COMPUTAÇÃO

RUA VISCONDE DE PIRAJÁ, 303 - GRUPO 1005
RIO DE JANEIRO - RJ - BRASIL
CEP: 22410 - TEL. (021) 5214638

Não deixe de ler:

Coleção linha Apple

1- 77 Programas para linha Apple
2- Programas Comerciais da linha Apple - V.1
   (Mala-direta, Contas a pagar e receber,
   Controle de estoque)
3- Usando o Visiplot
4- Programas Comerciais da linha Apple - V.2
   (Arquivos, Cadastro de clientes com
   emissão de fatura/duplicata, Controle de
   vendas)
5- Usando o Assembler 6502
6- Usando o Magic Window
7- Usando as Rotinas Internas do Apple

Coleção linha Sinclair

1- Curso Basic V.1
2- Curso Basic Avançado V.2
3- 47 Programas para ZX SPECTRUM

Coleção linha TRS

1- 77 Programas para linha TRS-80
   CP500/400/300/TRS COLOR
   E COMPATIVEIS

Abreu, Carlos Alberto Castro e, 1945-
Curso de Basic Vol.I
Carlos Alberto C.Abreu . -- 1.ed.--
Rio de Janeiro: Micro Kit Inf., 1986.
90 p: (Linha Sinclair; n.1)

ISBN 85-7065-013-2

1.Programação (Computadores)
I.Título.

CDD-- 001.642

Todos os direitos reservados e protegidos nos termos da lei. Nenhuma parte deste livro podera ser reproduzida ou transmitida sejam quais forem os meios empregados: eletrônicos, mecânicos, fotográficos, gravação ou quaisquer outros.

(C) Copyright & 1985 by
CARLOS ALBERTO DE CASTRO E ABREU

Publicações para Computadores

Linha Sinclair - TK 85, CP 200, TK 90X e outros.

Curso de Basic volume I

Teoria, exemplos e exercícios resolvidos, explanados em 10 aulas, oferecendo uma abordagem simples e direta.

Dicas para adaptar programas de outras máquinas para linha Sinclair.

Poderá ser usado por auto-didatas, ou como livro didático.

Curso de Basic volume II
programação avançada

Tem como objetivo complementar o livro Curso de Basic volume I.

Procura conduzir o usuário, a construir seus próprios programas, e não se limita ser apenas um dicionário de comandos e funções. Todos os programas possuem o respectivo fluxograma e uma explicação comentada da sua montagem. E um livro didático.

Um dos capítulos é dedicado a ensinar como desproteger programas da linha Sinclair.

Linha Apple - todos os computadores compatíveis com Apple e TK 2000.

77 Programas para linha Apple

Através de JOGOS e PROGRAMAS EDUCATIVOS você será induzido a pensar, resolver problemas, e tomar conhecimento de como poderá usar bem o computador, de forma simples e divertindo-se !

Programas Comerciais da linha Apple - V.1
para Pequena-Empresa

Este livro trás a listagem completa dos programas como: Mala-direta, Controle de estoque e Contas a receber e a pagar, indispensáveis à pequena-empresa.

O autor fez uma análise minunciosa desses programas, com o objetivo de dar oportunidade a você de aprender a programar, além de orientar o pequeno empresário como conduzir-se, na área de informática, de forma econômica.

Programas Comerciais da linha Apple - V.2
para Pequena-Empresa

Este livro trás a listagem completa dos programas como: Utilitário de Arquivos, Emissão de Faturas/Duplicatas e Vendas a varejo, indispensáveis à pequena- empresa.

O autor fez uma análise minunciosa desses programas, com o objetivo de dar oportunidade a você de aprender a programar, além de orientar o pequeno empresário como conduzir-se, na área de informática, de forma econômica.

Usando o Visiplot

O autor faz uma análise completa e exemplificada do programa Visiplot e mostra como tirar dados, ou seja interagir, com os programas Visicalc e Supervisicalc.

Usando o Assembler 6502

Tem como objetivos utilizar os recursos do microprocessador 6502.

Sua abordagem é a mais simples possível. Possui exemplos de programação prática e descrição das instruções do 6502.

Este livro pode ser usado por uma pessoa que nunca programou antes o Assembler.

PREFÁCIO

Desde a Segunda Grande Guerra, o computador tem gradualmente se inserido em nossa sociedade, e agora, apresenta um papel preponderante. O impacto de alguns enormes e complexos programas,como aterrissagem lunar, e o subsequente progresso na pesquisa espacial,demonstrou a importancia vital deste equipamento para a moderna tecnologia.

Cada um de nós, tem contactos diários com os seus servicos, através de recebimentos de faturas, uso de cartoes de crédito, e através de atos simples, mas essenciais, como a discagem telefonica de grande distancia (DDD), e nos bancos. Entretanto, nos sabemos que esses contactos constituem só o inícios;a gama de aplicacões continuará se expandindo nos próximos anos.

Estamos passando para uma era de maior uso e controle da informacão. Apesar da sua aplicacão

em muitos aspectos da vida diária, as pessoas continuam a ignorar como funciona,quais as vantagens e e limitacões, e como poderiam, melhorar profissionalmente, aprendendo mais acerca dele. Esta falta de conhecimento, conduz a algumas distorcões.Muitos temem uma mudanca radical da vida humana,subordinando o Homem aos impulsos de uma máquina insensível, enquanto outros acreditam que ele irá erradicar a a maioria das formas de trabalho, deixando o homem livre para o lazer.Ambas as visões são extremas, e a verdadeira promessa do computador, está no meio Ele e um maravilhoso instrumento de trabalho,com com grande potencial.

O objetivo deste curso, é apresentar um guia compreensível,projetado para ensinar ao iniciante a programá-lo,oferecendo uma abordagem simples, direta e tão precisa quanto nos foi possível.

Este método teve como apoio, a experiencia dos cursos para iniciantes oferecidos pela Micro-Kit Informática, para adultos e criancas, e visa, como livro,difundir a linguagem BASIC em todo o território nacional.

Tomou-se como base, a programacão voltada para os micro-computadores da linha Sinclair, Apple e Msx por serem os mais populares e possuirem um crescente número de instalações em nosso País.
A Micro-Kit tem sido a pioneira nesta área mas procura incentivar o espírito de aventura de nossos jovens que tem expandido as nossas fronteiras do Oeste,fornecendo-os subsídios para que sejam reproduzidos os nossos cursos nas regiões mais afastadas dos grandes centros urbanos.

SUMARIO

Primeira aula

1.1 - Introdução a programação.

O programador iniciante deve praticar o que está aprendendo no computador.

Esta experiência no microcomputador e um elemento vital no aprendizado da programação.

O que é um computador ?

- E uma máquina que aceita informação, processando-a de acordo com instruções específicas, e provê o resultado como nova informação.

Embora uma simples máquina de somar tenha esta mesma definição, um moderno computador tem a habilidade adicional para processar grandes quantidades de informações, em alta velocidade; fazer certas espécies de decisões e reter numerosos resultados na memória.

Quando se aprende a usar o computador, vai ficando evidente que essas facilidades somadas, transformam-no em um dispositivo de extraordinaria capacidade.

Antes de aprender a programá-lo, o estudante deve ter uma clara compreensão de dois termos básicos criados pela tecnologia da informática: software e hardware.

As instruções codificadas, programas que dirigem o computador para processar informação, são chamadas software.

O que pretendemos neste curso, e ensinar o caminho para o aluno entender o processo de desenvolvimento do software.

O hardware, consiste numa unidade de processamento central (CPU), memória principal, vários dispositivos de entrada e saída e de armazenagem. Instruções são executadas, uma de cada vez, na CPU, enquanto a memória principal armazena dados, e outras instruções de resultados calculados.

Dispositivos de entrada e saída permitem ao usuário, entrar com informações no computador, e receber o resultado processado. Isto e usualmente feito num teclado, um dispositivo para datilografar a entrada, com a saída sendo registrada em uma impressora ou vídeo. Os mais comuns dispositivos de armazenagem são os discos e as fitas magnéticas, que são usados para armazenagem de programas e dados por um tempo maior.

O que e um programa ?

- É um plano que informa ao computador a sequência na qual entram os dados; como os cálculos são realizados; como fazer as decisões necessárias e como imprimir a saída desejada.

O programa deve ser escrito numa linguagem tal como a BASIC, a qual o computador compreende. Esta linguagem foi originada do Dartmouth College, no início da década de 60.

A vantagem do BASIC é permitir uma grande gama de aplicações e a capacidade de

tomada de decisões, embora use um vocabulário limitado de palavras em inglés.

Estas palavras atuam como COMANDOS, que instruem o computador para fazer certas operações desejadas. Embora um programa muitas vezes possa parecer perfeitamente compreensível para o programador, o computador podera vê-lo como algo estranho, se forem utilizados comandos que não façam parte do seu vocabulário. É importante entender que a máquina compreende somente comandos legítimos em BASIC.

Devido a que vários programas possam ser escritos, produzindo saídas idênticas, o tempo que ele leva para processar cada programa, pode variar. O objetivo de uma boa e eficiente programação, e produzir resultados válidos com o mínimo de tempo de processamento.

1.2 - Numeração de linhas.

Para estabelecer a sequência de instruções em um programa, cada comando e designado por um número de linha. O computador lê as instruções, começando com o número de linha mais baixo, em seguida, o próximo número, e termina com o comando de número mais alto. É possível bater as instruções fora de ordem, pois o computador coloca-as na ordem.

Exemplo: Experimente digitar.

```
5Ø PRINT 3
2Ø PRINT 2
1Ø PRINT 1
```

Agora digite RUN e (ENTER ou NEWLINE) e você vera que o computador irá colocar as linhas em ordem crescente.

1.3 - Operações - Precedências e símbolos.

Se parênteses não são usados, a precedência de operações no computador e:

1) Elevação a potência.

2) Divisão e/ou Multiplicação, encontrada da esquerda para a direita.

3) Adição e/ou Subtração, encontrada da esquerda para a direita.

Operação	Símbolo matemático	Símbolo em BASIC
Adição	+	+
Subtração	-	-
Multiplicação	X ou .	*
Divisão	: ou /	/
Exponenciação	b^n	b ** n

Quando parênteses são usados, o computador calcula a quantidade dentro dos parênteses primeiro. Se mais de um conjunto de parênteses e usado, o conjunto mais interno tem precedência.

Por exemplo:

6/3*2 O resultado e 4, embora a resposta possa ser 6/6=1, o computador começa da esquerda e faz a divisão primeiro.

12+4/Ø O resultado é uma mensagem de erro. O computador não pode dividir por zero.

1.4 - Modo imediato.

Não é necessário escrever um programa de muitas linhas em BASIC. O computador executará um único comando, quando o comando

aparecer sem o número de linha.

Assim, comandos sem número de linhas, são executados imediatamente, enquanto comandos que começam com número de linha, são assumidos pelo computador, como parte de um programa, sendo assim armazenados.

Exemplo:

```
1Ø PRINT "COMPUTADOR"
```

E assumido como parte de um programa.

```
PRINT "COMPUTADOR"
```

Será impresso a palavra COMPUTADOR de imediato.

1.5 - Comandos PRINT e STOP.

Programa 1.1:

Este programa ilustra como o computador pode fazer cálculos e imprimir resultados.

```
1Ø PRINT 31 + 7Ø + 4
2Ø PRINT 2 * 5 - 6 / 3
3Ø PRINT (2 + 3) ** 4
4Ø PRINT (13 - 11) ** 3 / Ø
5Ø STOP
```


Existe uma série de pontos ilustrados por este simples programa:

1) O comando PRINT faz o computador calcular e imprimir o resultado desejado.

2) Cada programa, deve ter um comando STOP, como instrução de número mais alto, para informar ao computador que o programa esta completo. Qualquer comando de número mais alto que o STOP será ignorado pelo computador.

3) O programador digita o comando RUN, para instruir o computador para rodar o programa. Note que este comando não tem número de linha, e não e parte do programa.

1.6 - Comando LET.

O programa seguinte introduz o conceito de designação de resultados de cálculos para uma nova variável.

Êle também indica como um uso expandido do comando PRINT permite ao programador mais flexibilidade na saída.

Suponha que um programa e escrito para computar a área de uma mesa.

Programa 1.2:

```
1Ø LET L=15
2Ø LET C=25
3Ø LET A=L*C
```

```
40 PRINT"AREA DA MESA= ";A

50 STOP
```

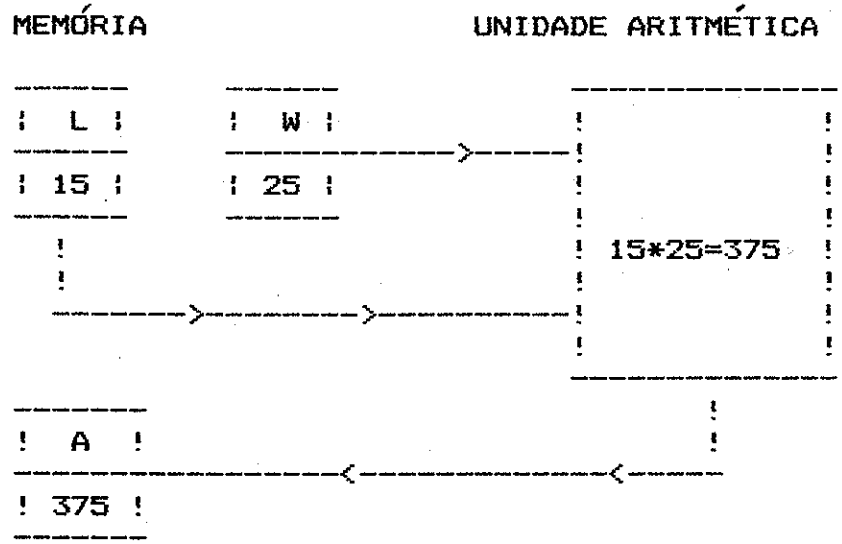
O comando LET, nas linhas 10 e 20 armazena os valores 15 e 25 nas variáveis L e C e na linha 30 ele comanda o computador para fazer um cálculo e designa o resultado para uma nova variável A. Para executar a linha 30, o computador necesssita dos valores de L e C, previamente designados nas linhas 10 e 20.

Na linha 40, o computador e instruído para imprimir as palavras: AREA DA MESA= e depois, o valor da variável A. Palavras ou símbolos entre aspas apos um comando PRINT, são impressos pelo computador.

O ";" instrui o computador para imprimir o valor da variável A, imediatamente apos as palavras entre aspas.

Na figura abaixo veremos como o computador faz o armazenamento das variáveis C e W.

"Gato": a variavel C ou W



A adição das seguintes linhas no programa 1.2, produzirá os resultados na forma de tabela.

```
 5 PRINT "COMPRIMENTO";"  ";"LARGURA";
   "  ";"AREA"

40 PRINT "     ";L;"        ";W;"        ";A
```

1.7 - Comando INPUT.

Em muitos casos, especialmente usando o computador, para fazer cálculos envolvendo dados de um laboratório, e desejável introduzir os dados através do teclado tão logo sejam conhecidos.

Programa 1.3:

Este programa e uma variação do programa 1.2. Agora os valores de L e C podem ser introduzidos durante a execução.

```
10 INPUT L

20 INPUT C

30 LET A=L*C

40 PRINT"AREA DA MESA= ";A

50 PRINT

60 GOTO10

70 STOP
```

Toda vez que o computador encontra o comando INPUT, ele coloca um cursor L no canto esquerdo inferior da tela.

O primeiro número do INPUT, e designado para a primeira variável na linha 10 que e L. O segundo número, e designado para a segunda variável que e C. O programa e então executado, e o valor de A e impresso.

Para parar um programa, em qualquer tempo, durante uma execução, o usuário deverá apertar as teclas SHIFT e STOP e a tecla ENTER ou NEWLINE, assim, poderemos reexecutar o programa ou listá-lo. Podemos melhorar o nosso programa, de maneira que, ao ser executado, ele faça uma pergunta.

Para isto devemos acrescentar a seguinte alteração:

```
5 PRINT"QUAIS SAO AS DIMENSOES?"
```

Devemos observar que na linha 60, aparece o comando GOTO. Este comando tem como finalidade retornar à linha 10 para podermos entrar com novos valores.

1.8 - Função SQR.

Veremos o uso desta função, através do programa 1.4.

Programa 1.4:

Este programa usa o teorema de Pitágoras $C^2 = A^2 + B^2$ para calcular a hipotenusa C de triângulos retângulos, utilizando os catetos A e B. Para o cálculo da variável C, e necessária a introdução da função SQR.

```
10 INPUT A

20 INPUT B

30 LET C=SQR(A**2+B**2)

40 PRINT"HIPOTENUSA= ";C

50 PRINT

60 GOTO 10

70 STOP
```

Na linha 3Ø, a função SQR calcula a raiz quadrada da expressão entre parênteses.

1.9 - Função INT.

O computador tem a habilidade de truncar decimais.

O comando A = INT(X), faz A igual ao maior número inteiro não maior que X.

Observe que esta função, não arredonda um número positivo mas, simplesmente, trunca-o. Para um número negativo, ela toma o próximo inteiro negativo mais baixo.

Por exemplo:

INT(3.764Ø) = 3 INT(5.9) = 5

INT(-1.7) = -2 INT(-3.Ø1) = -4

Arredondar um número para o inteiro mais próximo, e feito, fazendo-se A = INT(X + Ø.5).

Por exemplo:

Se X = 3.4 então 3.4 + Ø.5 = 3.9 e INT(3.9) = 3

Se X = 3.6 então 3.6 + Ø.5 = 4.1 e INT(4.1) = 4

A função INT e também usada para arredondar números para um número específico de casas decimais. INT(1Ø * X + Ø.5) / 1Ø arredondará X de uma casa decimal.

Por exemplo:

Se X = 2.57, então (1Ø * X + Ø.5) = (25.7 + Ø.5) = 26.2, e INT(26.2) = 26, que quando dividido por 1Ø e 2.6, o valor correto arredondado. Uma fórmula geral para arredondar X para N casas decimais e:

```
INT(1Ø ** N * X + Ø.5) / 1Ø ** N
```

Retornando ao programa 1.4, para arredondar a saída para o mais próximo decimo, a linha 4Ø, deve ser alterada para:

```
4Ø PRINT"HIPOTENUSA= ";INT(1Ø * C +
   Ø.5) / 1Ø
```

1.1Ø - Comando REM.

É um comando utilizado no corpo do programa para permitir ao programador, introduzir observações explanatórias.

Exemplo:

```
5 REM PROGRAMA QUE CALCULA A HIPOTENU
  SA DE UM TRIANGULO RETANGULO
```

Exercícios da primeira aula:

1-a) Quais das seguintes variáveis são legitimas em BASIC? (Sim ou Não)

0	1A	A	AB
_____	_____	_____	_____
P3	TO	Ø	H1
_____	_____	_____	_____

1-b) Quais são os tipos de variaveis em BASIC?

_____

2) Escreva um programa para resolver a expressão 12 * X + 7 * Y:

X	3	7	12
Y	2	9	-4

3) Um pedaço de pizza contem cerca de 375 calorias. Uma pessoa gasta 1ØØ calorias ao correr a distância de 1 milha.

Escreva um programa que pergunte a cada pessoa, quantos pedaços de pizza ela comeu e que informe a distância que deve ser corrida para queimar as calorias consumidas.

Exemplo:

QUANTOS PEDAÇOS VOCE COMEU ? 4

VOCE DEVE CORRER 15 MILHAS

4) Usando um comando INPUT, escreva um programa que calcule o volume de uma sala, dada a sua largura, comprimento e altura.

5) Escreva um programa para calcular a media de quatro números.

6) Escreva um programa para calcular os preços unitários dos itens de um supermercado.

7) Escreva um programa para converter graus Fahrenheit em graus Celcius.

A formula para isto e:

C = (5 / 9) * (F - 32)

8) Escreva um programa que calcule a soma de A com B e o produto de A com B.

Segunda aula

2.1 - Decisões e loops.

Os comandos apresentados na aula anterior permitem, ao computador, executar cálculos que podem ser feitos numa simples calculadora. No entanto, a habilidade de um computador de tomar decisões, o faz mais poderoso que máquinas que so fazem cálculos. Uma vantagem importante da tomada de decisões do computador, é permitir a execução de um conjunto de comandos durante um determinado número de vezes. Assim, de forma diferente do homem, os computadores podem repetir por várias vezes qualquer serviço, não importando o quão difícil ele possa ser.

Os comandos introduzidos na primeira aula, GOTO, PRINT, etc., são chamados incondicionais, pois o computador não tem escolha e procura-os na ordem de sua ocorrência no programa.

Nesta aula, comandos condicionais e comandos de tomada de decisões, tais como: IF ... THEN ..., FOR ... TO ... STEP ..., NEXT ... e a técnica de somação, são apresentados.

2.2 - Comando IF ... THEN ....

A forma geral para o comando IF ... THEN... e:

IF (relação) THEN (num. de linha)

A relação compara duas quantidades.

Elas devem ser separadas por um dos símbolos indicados abaixo:

Símbolo	Significado
=	igual
>	maior que
<	menor que
>=	maior que ou igual a
<=	menor que ou igual a
<>	não igual a

Um exemplo do comando IF...THEN...e:

```
20 IF X > 5 THEN GOTO 60
```

Quando a relação na linha 20 (X e maior que 5) for verdadeira, o computador desvia para a linha 60.

Quando a relação e falsa, porque X e menor ou igual a 5, o computador prossegue executando a próxima linha sequencial do programa.

Programa 2.1:

Este programa determina se o valor de entrada de X e solução para a equação:
X ** 2 - 82 * X + 657 = 0.

```
1Ø INPUT X

2Ø IF X ** 2 - 82 * X + 657 = Ø THEN
   GOTO 5Ø

3Ø PRINT X;" NAO E UMA SOLUCAO"

4Ø GOTO 1Ø

5Ø  RINT X;     JMA S   CAO"

6Ø GOTO 1Ø

7Ø STOP
```

Note a necessidade da linha 4Ø. Ela não permite a impressão da linha 5Ø, a menos que X satisfaça a relação no comando condicional da linha 2Ø.

2.3 - Comando FOR..TO..STEP.., NEXT.

Suponha que seja necessário, no programa 2.1, encontrar todas as soluções entre o conjunto de inteiros ímpares positivos de 1 a 99. Com os comandos disponíveis neste ponto, o problema pode ser resolvido, usando-se o comando INPUT, requerendo que 5Ø números sejam datilografados. Os comandos FOR...TO...STEP...e NEXT resolvem o problema de gerar uma grande sequência de números, considerando que cada número na sequência difere, de seu predecessor de uma quantidade constante. A única coisa que pode ser mudada,e estar diferente num FOR, são as variáveis, e as três relações. Se o STEP não for usado no comando, o computador

usara o valor de 1 para o STEP. Revisando o programa 2.1 para usar estes novos comandos, a linha 3Ø e apagada e as linhas 1Ø, 4Ø e 6Ø são substituidas por:

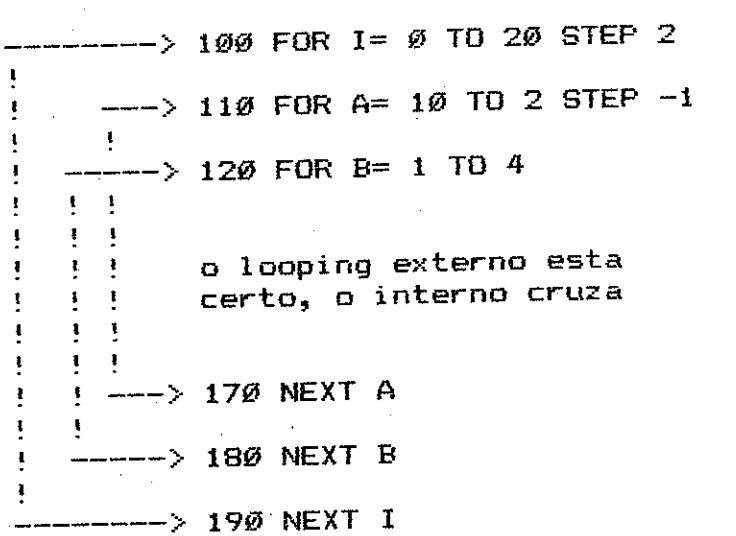
```
1Ø FOR X= 1 TO 99 STEP 2

3Ø

4Ø GOTO 6Ø

6Ø NEXT X
```

Estas linhas, criam um loop, para testar cada número impar de 1 a 99. Quando o programa alcança a linha 6Ø, o computador retorna à linha 1Ø para outro ciclo.

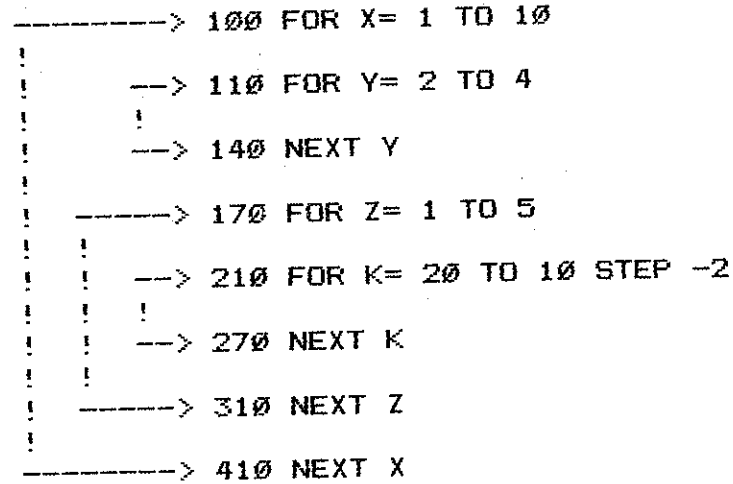
No comando NEXT, a variável deve ser a mesma que a usada no comando FOR que abriu o looping. Muitas vezes, estruturas mais complicadas de looping são necessárias a um programa. As estruturas de looping, não devem se cruzar. O exemplo abaixo, ilustra um segmento de programa com loopings cruzados.

```
  ---> 1ØØ FOR A= 2 TO 2Ø
  !
------> 11Ø FOR B= 4 TO 8
! !
! !       os loopings se cruzam
! !
! !
! ---> 24Ø NEXT A
!
------> 25Ø NEXT B
```

Um outro exemplo com loopings cruzados:



O exemplo seguinte ilustra um complicado looping, onde os loopings estão organizados corretamente:



No exemplo acima, nos temos looping duplos e looping dentro de looping. Lembre-se que qualquer combinação pode ser usada num programa, mas devemos ter o cuidado de não deixar os loopings se cruzarem.

Se isto ocorrer, o computador assinala um erro e para.

2.4 - Somação.

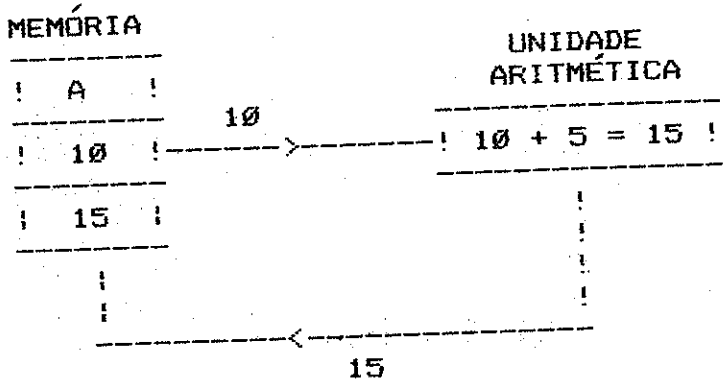
A técnica de somação e um bom exemplo da utilidade das técnicas de decisão e loop. O método começa nomeando uma variável e dando a ela um valor inicial. Cada vez que uma quantidade e somada ao valor anterior da variável, a soma dos dois passa a ser o novo valor da variável.

Por exemplo:

```
30 LET A = A + 5
```

O comando A = A + 5 e ilegal matematicamente. Neste caso, entretanto, o computador interpreta o sinal de igual como: "recoloque" ao invés de "igual a". Usando a analogia de caixa, assuma que o valor 10 foi designado para a caixa A.

Quando o computador processa a linha 30, 5 e somado ao valor na caixa A. O novo valor 15 substitui o previo valor na caixa A.



Programa 2.2:

O seguinte programa usa a técnica de somação para imprimir todos os inteiros pares de 50 ate 60.

```
10 LET A = 50

20 PRINT A;"    ";

30 LET A = A + 2

40 IF A <= 60 THEN GOTO 20

50 STOP
```

A linha 10, inicialmente designa para A o valor 50.

Na linha 30, o valor 2 e somado ao antigo valor de A.

Agora A=52. Na próxima vez que a linha 30 for encontrada, o valor de A passa a ser 54. Na linha 40, o valor de A e comparado com 60. O loop continua ate que o valor de A seja maior que 60.

Os comandos avançados em BASIC, entretanto, permitem o numero de linha no comando IF...THEN..., serem substituidos por um ou mais comandos. Os exemplos seguintes, ilustram como procedimentos que teriam requerido no mínimo dois comandos, podem ser reduzidos para um.

```
IF X = 5 THEN PRINT"CORRETO"

IF X = 12 THEN LET A = 7

IF X = 22 THEN LET N = N + 1
```

O BASIC permite que, uma série de comandos, sejam impressos numa única linha separados por : (dois pontos).

A linha:

```
IF X = 5 THEN PRINT"CORRETO": GOTO 50
```

fara imprimir a palavra CORRETO, e causará um desvio para a linha 50, se X = 5, mas não fara nada, se X não for igual a 5.

2.5 - AND/OR.

Os modificadores de comando AND e OR no comando IF...THEN... podem também serem usados para simplificar comandos nos quais mais de uma condição possa acontecer.

```
IF X > 5 AND Y = 3 THEN GOTO 50
```

Nós só iremos para a linha 50 se ambas as condições forem verdadeiras.

```
IF X > 5 OR Y = 3 THEN GOTO 5Ø
```

Nós iremos para a linha 5Ø se uma ou outra condição for verdadeira. Basta que uma condição seja verdadeira.

Programa 2.3:

Este programa testa valores de X e Y para saber se eles são soluções para as desigualdades 5 * Y + X ** 2 > 1Ø e Y ** 2 - 5 * X < 2Ø.

```
1Ø INPUT X,Y

2Ø IF 5 * Y + X ** 2 > 1Ø AND Y ** 2 -
   5 * X < 2Ø THEN GOTO 5Ø

3Ø PRINT X;" E ";Y;" NAO SAO SOLUCOES"

4Ø GOTO 1Ø

5Ø PRINT X;" E ";Y;" SAO SOLUCOES"

6Ø GOTO 1Ø

7Ø STOP
```

Exercícios da segunda aula:

1 - Escreva um programa que calcule a seguinte expressão:

(3/5) * X + 2 * X + 2/5 > 3ØØ

para os valores de X = (1Ø,12,14,...,4Ø)

2 - Escreva um programa que pergunte a idade de uma pessoa. Se a pessoa tiver 16 anos ou mais, o computador devera imprimir: "VOCE PODE DIRIGIR UM CARRO". Em caso contrário, faça o computador informar quantos anos a pessoa devera esperar para dirigir um carro.

3 - Escreva um programa que gere uma tabela de números e suas respectivas raízes quadradas.

4 - Escreva um programa que conte e imprima números de 1Ø em 1Ø entre Ø e 5ØØ.

5 - Um hotel tem 1Ø andares. Cada andar tem 1Ø aposentos. Faça o computador imprimir uma etiqueta para cada apartamento informando o andar e o número do aposento.

6 - Escreva um programa que imprima uma tabela de conversão de polegadas para centímetros. Existe 2.54 centímetros numa polegada. Comece com Ø e continue ate 1Ø polegadas em STEPS de Ø.5 polegadas.

Terceira aula

3.1 - "STRINGS" (cadeias e formatação).

Nas aulas anteriores, as variáveis têm sido usadas apenas para representar números. Esta aula introduz a variável "STRING", que permite que as variáveis representem palavras, ou qualquer combinação de letras, dígitos e caracteres especiais.

Entre as vantagens das variáveis "STRING", esta a flexibilidade permitida para a formatação. Os comandos de formatação num programa, são aqueles que informam o computador do "layout" desejado para a saida. Comandos PRINT são introduzidos para permitir maior variedade na formatação.

3.2 - Variáveis STRING.

As variáveis numéricas, representam somente números, e são denotadas por várias letras, ou uma letra seguida de um dígito.

O termo STRING, indica uma sequência de caracteres representadas por letras, dígitos e/ou caracteres especiais, que são tratados como uma unidade única.

Por exemplo: ANA, 1984, 609-896, CASA e 5:00 P.M. seriam STRINGS. Qualquer nome legítimo de variável seguido de um sinal de dolar ($), representa uma variável STRING.

Por exemplo: A$, B$, W$. A analogia com caixas, pode ser usada para enfatizar a diferença entre variáveis numéricas e variáveis string.

Enquanto caixas de variáveis numéricas contêm números, caixas de variáveis strings devem conter caracteres.

VARIAVEL NUMERICA	VARIAVEL STRING
E	E$
14.3	ERICKA

Programa 3.1:

O seguinte programa demonstra o uso de uma variável string.

```
10 INPUT A$

20 PRINT A$

30 GOTO 10

40 STOP
```

Onde A$ é : PEDRO, CARLA, 5000

Programa 3.2:

Podemos ter ambas as variáveis, numéricas e strings, no mesmo comando.

```
10 INPUT R$

20 INPUT H$
```

```
30 INPUT X

40 INPUT N$

50 PRINT R$;" ";X;" ";N$," ";H$

60 GOTO 10

70 STOP
```

Onde R$, H#, X e N$ são:

SALA, CASA, 15, CARLA
13, JOAO, PEDRO, 25

3.3 - Comparação de STRINGS.

E possível comparar variáveis string, usando-se um dos seguintes símbolos para determinar se alfabeticamente eles são: iguais (=); menor que (<); maior que (>) ou não iguais (<>).

O formato geral usado e:

IF (string) (simbolo) (string) THEN

Programa 3.3:

O programa abaixo coloca duas strings em ordem alfabética:

```
10 PRINT"DOIS NOMES"

20 INPUT A$
30 INPUT B$

40 IF A$ > B$ THEN GOTO 70
```



```
50 PRINT A$;" VEM ANTES DE ";B$

60 GOTO 10

70 PRINT B$;" VEM ANTES DE ";A$

80 GOTO 10

90 STOP
```

3.4 - SUB-STRINGS.

Dada uma string, uma sub-string consiste de um numero de caracteres consecutivos de string, apanhados em sequência.

A forma geral e:

expressão STRING (pos.1 TO pos.2)

Ela fornece uma sub-string da string A$, comecando com o carater na posição 1 e terminando com o carater na posição 2.

Para verificarmos isto, digitemos as linhas 10 e 20.

```
10 LET A$="123456789"

20 PRINT "CARACTERES 2, 3, E 4 DE A$
   SAO: "; A$(2 TO 4)
```

Observação: não e preciso colocar espaco depois do TO.

Digite RUN e você vera impresso na tela 234.

a) A$(TO pos.2) - Fornecerá a parte

esquerda da string A$, começando do primeiro carater ate a posição 2.

Adicionemos agora a linha 30:

```
30 PRINT"OS PRIMEIROS 4 CARACTERES DE
   A$ SAO: "; A$( TO 4)
```

Será impresso: 1234

b) A$(pos.1 TO) - Fornecerá a parte da direita da string A$, começando na posição 1 ate o final.

Digite a linha 40:

```
40 PRINT"A PARTE DA DIREITA DE A$ A
   PARTIR DO CARATER 5 E ";A$(5 TO)
```

Sera impresso: 56789

c) A$(pos.n) ou A$(pos.n TO pos.n) - Fornecerá o carater na posição n.

Tente a linha 50:

```
50 PRINT"O QUARTO CARATER DE A$ E: ";
   A$(4)
```

d) A$( ) ou A$(TO) - Fornecerá toda a string A$ como uma sub-string.

A linha 60 demonstra isso:

```
60 PRINT"A$ E: ";A$( )
```

Sera impresso: 123456789

e) LEN A$ - Fornece o número de caracteres da string A$. Tente a linha 70, re-

lembrando que LEN e uma função (aperte SHIFT ENTER, e a tecla K):

```
70 PRINT"O NÚMERO DE CARACTERES EM A$
   E: ";LEN A$
```

Será impresso: 9

f) VAL A$ - Determina o valor numérico da string A$.

A linha 80 mostra isto (VAL e uma função e esta na tecla J):

```
80 PRINT"O VALOR DE A$ E: ";VAL A$
```

Nos teremos 123456790. Note que o computador arredondou seu número de nove dígitos para um de oito dígitos.

g) STR$ X - Converte uma expressão numérica X em uma string. Tente as linhas 90 e 100. (STR$ e uma função e esta na tecla Y.)

```
 9Ø LET A=333

 1ØØ PRINT"A STRING DE A E: ";STR$ A
```

Será impresso: 333

h) CODE A$ - Fornece o código do primeiro carater da string A$. Se a string for nula, ou o primeiro carater e um espaço em branco, nos teremos o codigo "Ø".

A linha 11Ø demonstra isto. (CODE e uma função e esta na tecla I).

```
 11Ø PRINT "A$ COMECA COM O CARATER DE
     CODIGO NUMERO: ";CODE A$
```

A linha 11Ø imprimira o número "29" - que é o código do carater "1". Veja no seu manual a lista de códigos.

Tente digitar PRINT CODE "1", você obterá o mesmo resultado.

i) CHR$(cod) - Fornece o carater correspondente a um código. Relembre, que "cod" pode ser uma expressão constante, variavel ou formula, cujo valor deve estar entre Ø e 255.

CHR$ e uma função e esta na tecla U.

Digite a linha 12Ø.

```
 12Ø PRINT "O CARATER PARA O CODIGO 29
     E: ";CHR$(29)
```

Será impresso: 1

3.5 - Pontuação em comandos PRINT.

Nas últimas aulas, vírgulas e ponto e vírgulas, têm sido usados como pontuação em comandos PRINT para instruir o computador sobre como arrumar a saída. Cada linha no video e feita de um predeterminado número de espaços, nos quais caracteres podem ser impressos. Embora os diferentes vídeos possam ter um número variavel de espaços, a linha e usualmente subdividida em duas zonas. Se existem 32 espaços, as duas zonas contêm 16 espaços cada uma.
Isto para os equipamentos da linha Sinclair.Nos equipamentos da linha Apple e MSX,a tela é dividida em 3 partes. Se uma vírgula
é usada para formatação num comando PRINT, a informação que se segue e impressa começando no primeiro espaço da próxima zona disponível.

Quando um ponto e vírgula é utilizado, a informação seguinte é impressa começando no próximo espaço disponível.

Programa 3.4:

O seguinte programa ilustra o uso de pontuação e mostra a localização das duas zonas.

```
 1Ø INPUT A$

 2Ø FOR N = 1 TO 3

 3Ø PRINT A$;

 4Ø NEXT N
```

```
50 PRINT"12"

60 PRINT"A","B","C","D","E","F"
70 STOP
```

Onde A$ e: 1234567890

Programa 3.5:

Este programa calcula a media de três estudantes e mostra a impressão na forma de tabela.

```
10 PRINT "NOME";"          ";"GRAU";
   "       ";"MEDIA"

20 PRINT" "

30 INPUT A$

40 INPUT C

50 INPUT D

60 INPUT E

70 LET A = INT((C + D + E) * 10 / 3 +
   0.5) / 10

80 PRINT A$;"      ";C;" ";D; " ";E;
   "     ";A

90 GOTO 30

100 STOP
```

Onde A$, C, D e E são:

PEDRO, 89, 90, 90

CLARA, 70, 75, 80
ROSA, 80, 83, 85

3.6 - Comando PRINT TAB.

Uma outra maneira de formatar a saída, é usar o comando PRINT TAB. Este comando permite ao programador, começar porções da impressão em locais especificados.

Exatamente como o "TAB" da máquina de escrever, a primeira posição de carater e TAB(0). Uma informação de impressão nas posições 5, 15 e 25 é feita da seguinte maneira:

```
10 PRINT TAB(5);"ISTO";TAB(15);"E";
   TAB(25);"TAB"
```

É importante usar ponto e vírgula (;) após o TAB.

Exercícios da terceira aula:

1 - Como e que uma variável string e identificada em BASIC ?

2 - Escreva um programa para desenhar a letra (L) como na figura abaixo. Somente dois comandos PRINT podem ser usados no texto do programa.

```
XX
XX
XX
XXXXXX
XXXXXX
```

3 - Escreva um programa que produza a saída abaixo usando somente um comando PRINT. Os nomes são espaçados 1Ø vezes, do início de um nome para o início do próximo que e impresso na linha seguinte.

Por exemplo:

```
MARIO
          ANTONIO
CARLOS
          RICARDO
LUCIANO
          BRUNO
```

4 - Escreva um programa que entre com uma cadeia de caracteres e a sua saída seja em diagonal. Cada letra será impressa numa linha embaixo da outra.

5 - Escreva um programa para entrar com uma cadeia de caracteres, e a saída seja impressa na vertical.

6 - Escreva um programa que quando você entrar com uma cadeia de caracteres ele imprima a cadeia, e vai retirando um carater por vez. Por exemplo: se você digitar COMPUTADOR, o computador imprimirá.

```
COMPUTADOR
OMPUTADOR
MPUTADOR
PUTADOR
UTADOR
TADOR
ADOR
```


Quarta aula

4.1 - Funções e subrotinas.

Esta aula esclarece o uso da função SGN, valor absoluto ABS e a função geradora de números aleatorios RND.

Em adição, algumas técnicas de poupar tempo em programação mais sofisticada e apresentada. Isto inclui o comando GOSUB.

4.2 - Funções SGN e ABS.

Em algumas situações que usualmente envolvem equações matemáticas, pode ser necessário saber, se uma variável é positiva ou negativa, ou determinar seu valor absoluto.

A função SGN tem somente três valores possíveis:

SGN(X) = 1 se X > Ø

SGN(X) = -1 se X < Ø

SGN(X) = Ø se X = Ø

Programa 4.1:

```
1Ø FOR X = 2Ø TO -2Ø STEP -5
2Ø LET A = SGN(X)
3Ø PRINT A;" ";
4Ø NEXT X
```

```
5Ø STOP
```

Programa 4.2:

O valor absoluto de um número; !X!, pode ser encontrado usando-se ABS(X).

```
1Ø FOR X = -2Ø TO 2Ø STEP 5
2Ø LET A = ABS(X)
3Ø PRINT A;" ";
4Ø NEXT X
5Ø STOP
```

4.3 - Comando RND e RANDOM.

4.3.1 - Linha Sinclair

Em muitas simulações de computador, tais como, experimentos de laboratório, problemas de ciência geral e jogos, é necessário gerar números aleatórios. O computador faz isto, usando-se a função RND.

O comando RND instruirá o computador para produzir um número aleatório entre Ø e 1.

Programa 4.3:

```
1Ø FOR X = 1 TO 1Ø
2Ø LET R = RND
```



```
3Ø PRINT R,
4Ø NEXT X
5Ø STOP
```

Para obter inteiros de Ø ate 9, a função INT é empregada.

A linha 2Ø é alterada para:

```
2Ø LET R = INT(1Ø * RND)
```

Este comando multiplica o número randômico original por 1Ø, e então, trunca o resultado para o próximo inteiro menor.

A seguinte equação pode ser usada para gerar um conjunto de inteiros randômicos, nos quais N, é o número de inteiros no conjunto, com A sendo o menor inteiro.

```
2Ø LET R = INT(N * RND) + A
```

Para produzir um conjunto de inteiros randômicos de 4Ø ate 5Ø, inclusive, use N = 11 e A = 4Ø. Usando somente o comando RND, o programa produzirá o mesmo conjunto de inteiros randômicos cada vez que ele correr. Para produzir um conjunto diferente a cada corrida, o comando RAND deve ser incluído no programa.

Programa 4.4:

O programa abaixo inclui um comando RAND para produzir números inteiros randômicos de 253 ate 294, inclusive.

```
 5 RAND

10 FOR X = 1 TO 10

20 LET R = INT(42 * RND) + 253

30 PRINT R;" ";

40 NEXT X

50 STOP
```

Programa 4.5:

O programa de jogo abaixo, escolhe aleatoriamente números de 2 dígitos entre 10 e 99 e da ao jogador 3 chances de acertar cada um dos dois dígitos.

```
20 LET Y = INT(90 * RND) + 10

30 FOR A = 1 TO 3

40 PRINT"INFORME O PRIMEIRO DIGITO"

45 INPUT B

50 IF B = INT(Y/10) THEN GOTO 90

60 NEXT A

70 PRINT"VOCE ESTA ENGANADO, O NUMERO
    ERA ";INT(Y/10)

80 GOTO 100
```



```
90 PRINT"VOCE ESTA CORRETO"

100 FOR D = 1 TO 3

110 PRINT"INFORME O SEGUNDO DIGITO"

115 INPUT E

120 IF E = Y - 10 * INT(Y/10) THEN
    GOTO 160

130 NEXT D

140 PRINT "VOCE ESTA ENGANADO, O NUME
    RO ERA ";Y - 10 * INT(Y/10)

150 GOTO 170

160 PRINT"VOCE ESTA CORRETO"

170 STOP
```

4.3.2 - Linha Apple e MSX

Na linha Apple, a função RND deve ser usada com um parâmetro entre parênteses. Se o conteúdo do parâmetro for zero, o número aleatório gerado será sempre o mesmo.
Na linha MSX,para que seja gerado um número realmente aleatório, usar como parâmetro (-TIME).
Exemplo: 10 X = 8
20 PRINT RND (X)
Na linha Apple
20 PRINT RND (-TIME) na linha MSX
A seguir, você pode fazer adaptações nos programas acima,dependendo do tipo do seu equipamento

4.4 - Comando GOSUB e RETURN.

Nesses programas, onde partes do mesmo são repetidas, mas nào produzem os mesmos valores cada vez, pode ser eficiente usar uma subrotina. Uma subrotina inicia com o comando GOSUB "numero de linha", e termina com o comando RETURN.

Por exemplo:

```
 1Ø .......

 2Ø GOSUB 2ØØ

 3Ø .......

2ØØ .......
21Ø .......

23Ø RETURN

999 STOP
```

O esqueleto do programa acima, ilustra o uso de uma subrotina. Na linha 2Ø, o programa desvia para a subrotina que começa na linha 2ØØ, executa a subrotina, e então, na linha 23Ø, retorna para o corpo do programa na linha 3Ø. E´válido usar mais de um comando RETURN, dentro da subrotina.

Deve ser lembrado, que o computador, sempre retorna para o primeiro comando após o comando GOSUB, que chamou a subrotina.

Entretanto, e aconselhável colocar as subrotinas no final do programa.

As subrotinas, podem ser aninhadas de tal manèira, que uma chame a outra.

Programa 4.6:

Este programa é uma versão estendida do programa 4.5, reescrito de tal maneira, que um número de 4 dígitos possa ser adivinhado dígito por dígito, dando ao jogador 3 oportunidades para cada dígito.

```
 2Ø LET R = INT(9ØØØ * RND) + 1ØØØ

 3Ø LET A = INT(R/1ØØØ)

 4Ø LET R = R - 1ØØØ * A

 5Ø LET A$ = " PRIMEIRO "

 6Ø GOSUB 5ØØ

 7Ø LET A = INT(R/1ØØ)

 8Ø LET R = R - 1ØØ * A

 9Ø LET A$ = " SEGUNDO "

1ØØ GOSUB 5ØØ

11Ø LET A = INT(R/1Ø)

12Ø LET R = R - 1Ø * A

13Ø LET A$ = " TERCEIRO "

14Ø GOSUB 5ØØ

15Ø LET A = R

16Ø LET A$ = " QUARTO "

17Ø GOSUB 5ØØ
```

```
180 GOTO 999

500 FOR X = 1 TO 3

510 PRINT"INFORME O ";A$;" DIGITO"

520 INPUT B

530 IF B = A THEN GOTO 570

540 NEXT X

550 PRINT"VOCE ERROU TRES VEZES.  O ";
    A$;" DIGITO ERA ";A

560 RETURN

570 PRINT"VOCE ESTA CORRETO.  O ";A$;
    " DIGITO ERA ";A

580 RETURN

999 STOP
```

Sem uma subrotina, o programador teria que datilografar as linhas 500/580 quatro vezes. Note como as linhas 30, 70, 110 e 150 isolam cada dígito, e como os valores de A, R e A$ são alterados após cada RETURN da subrotina.

Obs. A função RND deste programa está ajustada para a linha Sinclair.
Caso o seu equipamento seja de outra linha, faça os ajustamentos constantes do item 4.3.2

Exercícios da quarta aula:

1 - Para obtermos uma sequência diferente de números aleatórios cada vez que o programa e executado, qual o comando que devemos utilizar ?

2 - Para obtermos a mesma sequência de números aleatórios cada vez que o programa e executado, qual o comando que deve ser utilizado ?

3 - Escreva um programa para imprimir 10 números inteiros aleatórios de 50 até 75, inclusive.

4 - Escreva um programa que simule o lançamento de um dado. O dado deverá ser lançado 10 vezes e em cada vez deverá imprimir o número da face.

Quinta aula

5.1 - Variáveis subscritas.

A função RND desta aula está ajustada para a linha Sinclair.Se o seu caso for outro,faça os ajustamentos constantes do item 4.3.2

O principal objetivo das variáveis subscritas, e armazenar, facilmente, grande quantidade de dados, recuperando-os mais tarde. Previamente, num programa, era possível dar nome a somente 286 variáveis.
O uso de subscritos (indices), e o comando DIM, expande vastamente este limite.

5.2 - Variáveis subscritas simples.

O programa seguinte, gera dez números aleatórios entre 1 e 20.

Programa 5.1:

```
10 FOR X = 1 TO 10

20 LET Y = INT(20 * RND + 1)

30 PRINT Y;" ";

40 NEXT X

50 STOP
```


Neste programa, cada vez que um valor e designado a Y, ele substitui o valor previo de Y. O uso da variável subscrita simples, resolverá o problema de números duplicados.

Uma variável simples subscrita, ou uma variável "string" subscrita, e representada por qualquer variável BASIC legítima, ou nome de um string seguido por um inteiro, que deve aparecer entre parênteses.

Por exemplo:

A(1), C5(3), T$(4).

São todas variaveis subscritas.

A variável subscrita L(1) não é, entretanto, a mesma coisa de L1. Cada vez que o índice da variável subscrita e modificado, uma nova variável é estabelecida, sem que seja perdido o conteudo da variável anterior.

Programa 5.2:

Este programa, é uma revisão do programa 5.1, e utiliza variáveis subscritas, para armazenar os números. Este processo, pode ser difícil sem usar as variáveis subscritas.

```
 5 DIM Y(10)

10 FOR X = 1 TO 10

20 LET Y(X) = INT(20 * RND + 1)
```

```
30 PRINT Y(X);" ";

40 NEXT X

50 STOP
```

Programa 5.3:

Utilizando variáveis subscritas, o programa abaixo calcula a área de um campo medido por três estudantes.

Suas medições, são em metros, e a media de todas as suas áreas calculadas, são em metros quadrados.

```
10 LET B = 0

20 DIM L(3)

30 DIM W(3)

40 FOR X = 1 TO 3

50 PRINT "INFORME O COMPRIMENTO E A
   LARGURA ?"

60 INPUT L(X)

70 INPUT W(X)

80 LET A = L(X) * W(X)

90 PRINT"AREA = ";A;" M**2"

100 NEXT X

110 PRINT TAB(3);"COMP.(L)";TAB(12);
    "LARG.(W)";TAB(20);"AREA(M**2)"
```



```
120 FOR Y = 1 TO 3

130 PRINT TAB(3);L(Y);TAB(12);W(Y);
    TAB(20);L(Y) * W(Y)

140 LET B = B + L(Y) * W(Y)

150 NEXT Y

160 PRINT"AREA MEDIA = ";B/5;" M**2"

170 STOP
```

5.3 - Comando DIM.

Quando o mais alto valor do índice excede de 10 para uma variável subscrita, o computador deve ser informado.

O comando DIM (DIMENSION) é usado para instruir o computador, para abrir bastante caixas em sua memória para comandar a entrada de dados. O programa 5.3 pode armazenar 100 comprimentos e larguras, fazendo modificações apropriadas nas linhas 20, 30 e comandos FOR...TO... É impossível requerer mais espaços na memória, do que o computador pode suprir. Isto resulta em mensagem de erro.

5.4 - Variáveis duplamente subscritas.

O BASIC, pode também usar subscritos duplos para nomear uma variável. Isto é similar a subscritos simples, exceto que números inteiros separados por vírgulas devem aparecer entre parênteses.

Por exemplo:

A(1,5), B(7,3) e C$(4,9)

São variáveis subscritas.

O computador reserva espaço na memória, para variáveis subscritas duplamente por linhas e colunas, ao invés de uma simples coluna, como é o caso das variáveis de subscritos simples. Isto provê uma técnica conveniente para manusear problemas nos quais, os dados, tenham natureza bidimensional. Para entender mais claramente, como variáveis duplamente subscritas operam, a analogia de caixas, e útil novamente.

O primeiro inteiro no subscrito, identifica a linha, e o segundo inteiro, a coluna na qual a variável está localizada.

Por exemplo:

A(2,3)

Está localizada na segunda linha e terceira coluna.

Programa 5.4:

O programa seguinte, pode ser usado por uma companhia de aviação para fazer reserva em um avião. O avião tem 40 linhas de assentos, e 5 colunas por linha. Um nome de pessoa e armazenado na caixa correspondente a cada assento.

```
10 DIM A$(40,5)
```



```
20 FOR X = 1 TO 200
30 PRINT "QUE LINHA E ASSENTO VOCE
   DESEJA ?"
40 INPUT R
50 INPUT S
60 IF A$(R,S) > " " THEN GOTO 100
70 PRINT"QUAL E O SEU NOME ? "
80 INPUT A$(R,S)
90 GOTO 230
100 PRINT "DESCULPE, ESTE ASSENTO JA
    ESTA RESERVADO"
110 PRINT "VOCE DESEJA UMA LISTA DE
    ASSENTOS VAGOS, S ou N ?"
120 INPUT B$
130 IF B$ = "N" THEN GOTO 30
140 PRINT"LINHA,ASSENTO"
150 FOR R = 1 TO 40
160 FOR S = 1 TO 5
170 IF A$(R,S) > " " THEN GOTO 190
180 PRINT TAB(1);R;TAB(5);",";
    TAB(7);S
190 NEXT S
```

```
200 NEXT R

210 PRINT" "

220 GOTO 30

230 NEXT X

240 PRINT"O AVIAO ESTA LOTADO.  OUTRA
    PARTIDA AMANHA."

250 STOP
```

A linha 10, define as dimensões do planejamento de assentos, e a linha 60, determina se um assento ja foi preenchido.

No começo do programa, todas as caixas das variáveis "string" subscritas são automaticamente designadas por um espaço em branco simples (" "). Caixas de variáveis numéricas subscritas, não preenchidas, por outro lado, são automaticamente designadas por 0. Se um passageiro escolhe uma poltrona ja preenchida, a linha 110 pergunta se ele deseja uma lista de assentos vagos.

Se ele digita N, o computador retorna para a linha 30. Por outro lado, as linhas de 140 ate 200 instruem o computador para imprimir uma lista de assentos não preenchidos.

Observe os loops aninhados requeridos, para imprimir os números das linhas e assentos.

5.5 - Limitação de variáveis subscritas.

Strings subscritas e variáveis numéricas, melhoram grandemente a habilidade do programador, para armazenar, e manipular grande quantidade de dados dentro de uma corrida de um certo programa. Entretanto, deve ser lembrado, que se o programa rodar de novo, todos os dados armazenados na memoria do computador, são apagados; todas as caixas passam a ter os valores brancos ou zeros na próxima corrida.

Exercícios da quinta aula:

1 - Qual o objetivo do comando DIM ?

2 - Nós temos um conjunto chamado X. Qual a variável (o nome dela) que o BASIC usa, para localizar o elemento da fileira 3 e coluna 4 ?

3 - Escreva um programa onde você possa entrar com números em uma variável subscrita duplamente, e calcule e imprima, a soma dos elementos em cada linha, e o produto dos elementos de cada coluna.

4 - O seguinte conjunto se chama A:

1	3	5
6	2	4

a - Escreva um comando DIM para A.

b - Qual e o valor de A(2,3) ?

c - Qual o valor de A(A(1,1),A(2,2))?

d - Se X=1 e Y=2, qual o valor de A(X,Y) ?

Sexta aula

Esta aula só está operacional para a linha Sinclair

6.1- Comando PAUSE:

O comando PAUSE pode ser usado para simular o modo SLOW, na execução de um programa.

A forma geral e: PAUSE N

onde: N e o número de quadros que a televisão mostra por segundo.

Algumas televisões apresentam 6Ø quadros por segundo, ou seja PAUSE 6Ø, na qual a execução do programa será interrompida durante 1 segundo. Quando o computador esta em PAUSE, se você digitar qualquer tecla menos espaço, o tempo de interrupção sera menor. Depois que o tempo de PAUSE terminar a tela piscará ou ficará escura.

É aconselhável que após uma linha que contenha o comando PAUSE, usemos uma outra linha com POKE 16437,255 ou poderemos correr o risco de perdermos o nosso programa.

Com o uso do comando PAUSE, é possível programarmos o computador como um relógio, tomando o cuidado em observar, que a precisão não e perfeita.

Exemplo: Programa para desenhar um relógio:

```
1Ø REM DESENHO DO RELOGIO
2Ø FOR N = 1 TO 12
```

```
30 PRINT AT 10-10*COS(N/6*PI),10+
   10*SIN(N/6*PI);N
40 NEXT N

50 REM FUNCIONAMENTO DO RELOGIO

60 FOR T = 0 TO 10000

70 LET A = T/30*PI

80 LET SX = 21+18*SIN A

90 LET SY = 22+18*COS A

100 PLOT SX,SY

110 PAUSE 50

120 POKE 16437,255

130 UNPLOT SX,SY

140 NEXT T
```

Observe como o tempo e controlado pela linha 110.

Você esperaria PAUSE 60 para fazê-lo mudar uma vez por segundo, mas o processamento também toma tempo, e, esse tempo tem que ser considerado. A melhor maneira de fazê-lo e através de tentativa e erro, cronometrando o relógio do computador com um real e ajustando a linha 110. Você não pode fazer isso com precisão; um ajuste de um quadro por segundo e dois por cento ou meia hora em um dia.

6.2 - Função INKEY$.

A função INKEY$, não tem argumento, lê o teclado como se você estivesse pressionando uma tecla; o resultado e o carater que aquela tecla representa; caso contrário, o resultado e uma string vazia.

Os caracteres de controle, não tem o efeito comum, porem, dão resultado como CHR$ 118 para NEWLINE. Eles são impressos como "?". Tente este programa, que funciona como uma máquina de escrever.

```
10 IF INKEY$ = " " THEN GOTO 10

20 PRINT INKEY$;

30 GOTO 10
```

A linha 10 espera você pressionar uma tecla. Ao contrário de INPUT, INKEY$ não espera você. Assim, você não digita NEWLINE. Por outro lado, se você não digitar nada, você tera perdido a sua chance.

A função INKEY$ e muito útil nos programas de jogos com DISPLAY de gráficos.

Exemplo:

Você esta no controle de uma espaçonave, e esta, em combate contra uma nave inimiga. Você deve tentar destruir quantas naves puder. Se for destruído, terá outra chance para provar que é bom.

Você tem três naves, e pode usar as

teclas 6 e 7 para mover-se para cima e para baixo e a tecla 1 para atirar no inimigo.

```
100 LET J=3

110 LET S=-1

120 LET D=INT(RND*28)+3

130 LET S=S+1

140 PAUSE 20

150 LET H=11

160 LET L=INT(RND*20)

170 CLS

180 PRINT AT H,0;"=O"

190 PRINT AT L,D;"+="

200 IF INKEY$ = "1" THEN GOTO 250

210 IF RND < 0.4 AND ABS(H-L) < 2
    THEN GOTO 300

220 LET H=H-(H>0)*(INKEY$="7") +
    (H<20)*(INKEY$="6")

230 LET L=L+INT(RND*3)-1+(L<0)-(L>20)

240 GOTO 170

250 FOR A=2 TO D

260 PRINT AT H,A;"-"
```

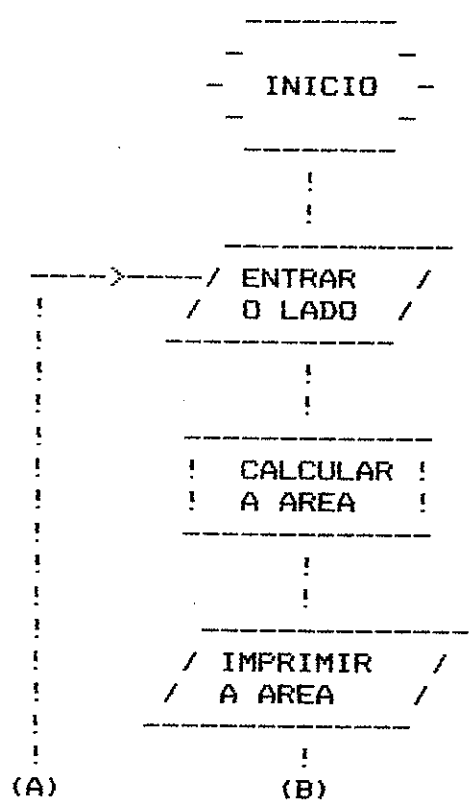


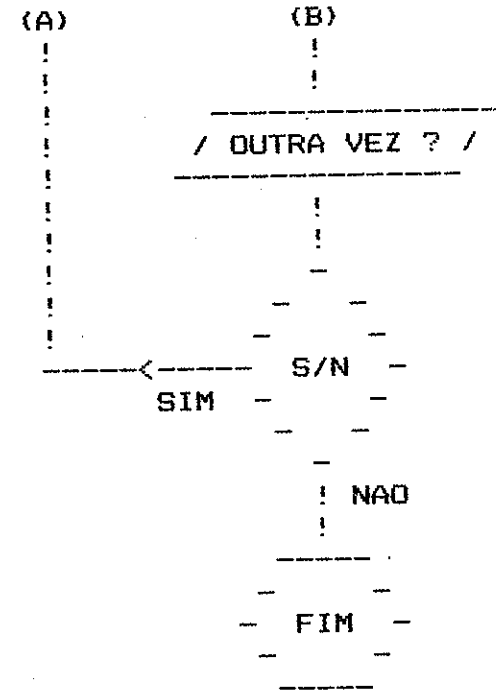
```
270 NEXT A

280 IF H=L THEN GOTO 120
290 GOTO 210

300 FOR A=D TO 1 STEP -1

310 PRINT AT L,A;"-"

320 NEXT A

330 IF H<>L THEN GOTO 220

340 LET J=J-1

350 PRINT"VOCE FOI ATINGIDO"

360 IF J THEN GOTO 120

370 PRINT"FIM DO JOGO"

380 PRINT"PONTOS= ";S

390 PAUSE 50

400 RUN
```

Sétima aula

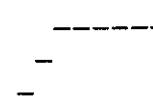
7.1 - Fluxograma e programa aplicativo.

Fluxograma e a representação de um processo por símbolos gráficos. Em programação, "processo" refere-se a "programa ou sequência de instruções". A seguir, apresentamos um fluxograma, para um programa de cálculo da área de um quadrado.

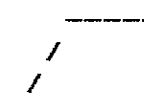




Veja, a seguir o que representam os diferentes símbolos empregados.



Representa o início ou fim do programa. Habitualmente so devem existir dois destes símbolos por programa. Um para o início, e outro para o fim.



Significa uma entrada ou saída de dados, causando uma interrupção no programa, até a ação do operador, como por exemplo: com as instruções INPUT, PRINT.

Significa execução, mas não relacionada com a entrada ou saída de dados ou tomadas de decisão. Algo como operações matemáticas.

É uma tomada de decisão. Oferece duas possibilidades sim ou não.

Sequência ou desvio do programa.

Quando as linguagens de programação eram mais complexas, e os computadores eram ainda de primeira e segunda gerações, os fluxogramas até que podiam ter sido boa coisa.

TOP DOWN e uma expressão do inglês, difícil de oferecer uma boa tradução.

Poderíamos dizer "de cima para baixo", ou "linha reta", ou deixemos mesmo TOP DOWN, que significa exemplificar a execução

de um programa através de suas funções, principalmente pelo esboço das entradas e saídas de dados, em um resumo escrito.

Por exemplo, o programa da área do quadrado:

1. INPUT LADO
2. CALCULAR A AREA
3. PRINT A RESPOSTA
4. CONTINUA ? S/N
5. SIM, THEN RUN
6. NAO, THEN STOP

É um resumo do que se pretende que o programa faça.

Transformando em programa:

```
10 PRINT"LADO ?"

12 INPUT L

14 LET A=L*L

16 PRINT L

18 PRINT"AREA= ";A;" M2"

20 PRINT"OUTRA VEZ ? (S OU N)"

22 INPUT P$

24 CLS
26 IF P$="S" THEN RUN
28 STOP
```

Na linha Apple deve ser usado o comando Home na linha 24.

Programa 7.1 - Lista telefônica.

Este programa arquiva N telefones seguidos de nome e endereço. Procura o telefone pelo nome ou vice-versa, oferece possibilidade de alterar as fichas, etc..
Os comandos CLS devem ser substituídos pelo comando HOME para usuários da linha Apple.
a - Programa.

1) Modulo "MENU".

```
200 PRINT
205 PRINT"DIGITE PARA PROCURAR:"
210 PRINT"1 - PELO NOME"
220 PRINT"2 - PELO TELEFONE"
221 PRINT"DIGITE:"
222 PRINT"3 - PARA ALTERAR/INCLUIR"
224 PRINT"0 - PARA NOVO ARQUIVO"
```

2) Modulo "ESCOLHA"

```
230 INPUT P
240 IF P<0 OR P>3 THEN GOTO 230
242 IF P<>0 AND N=0 THEN GOTO 230
```



```
245 CLS
248 IF P=0 THEN GOTO 22
250 IF P=1 THEN GOTO 400
260 IF P=2 THEN GOTO 300
270 IF P=3 THEN GOTO 510
```

3) Modulo "NOVO ARQUIVO"

```
 22 CLS
 25 PRINT"QUANTOS NOMES ? ";
 30 INPUT N
 40 PRINT N
 45 PRINT
 50 PRINT"MAX. 32 LETRAS POR NOME"
 70 DIM A$(N,32)
 80 DIM A(N)
 90 DIM B$(N,32)
100 FOR X=1 TO N
110 CLS
120 PRINT"QUAL O ";X;" NOME ?"
```

```
130 INPUT A$(X)
140 PRINT"QUAL O TELEFONE ?"

150 INPUT A(X)

154 PRINT"QUAL O ENDERECO ?"

156 INPUT B$(X)

158 IF K=1 THEN GOTO 175

160 CLS

161 PRINT A$(X)

162 PRINT A(X)

163 PRINT B$(X)

164 PAUSE 120

165 CLS

170 NEXT X

175 LET K=0

180 PRINT"NOMES E TELEFONES ARQUIVADO
    S"
```

4) modulo procura "PELO NOME"

```
400 CLS

405 PRINT"DIGITE O NOME"

410 INPUT X$
```



```
412 LET X=LEN X$

420 FOR C=1 TO N

425 LET K$=A$(C,(X+1))

430 IF X$=A$(C,TO X) AND CODE(K$)=0
    THEN GOTO 450

435 NEXT C

440 GOTO 490

452 PRINT A$(C)

455 PRINT A(C)

460 PRINT B$(C)

470 PRINT

480 GOTO 200

490 PRINT"NOME NAO ENCONTRADO"

500 GOTO 200
```

5) modulo procura "PELO TELEFONE"

```
300 CLS

305 PRINT"DIGITE O TELEFONE"

310 INPUT X

320 FOR C= 1 TO N
```

```
330 IF X=A(C) THEN GOTO 350
335 NEXT C
340 GOTO 380
350 PRINT X
352 PRINT B$(C)
355 PRINT A$(C)
360 PRINT
370 GOTO 200
380 PRINT"TELEFONE NAO ENCONTRADO"
386 PRINT
390 GOTO 200
```

6) modulo "ALTERACAO E INCLUSAO"

```
510 CLS
515 PRINT"QUAL A FICHA PARA ALTERAR
    ?"
520 PRINT"DE 1 A ";N
530 INPUT X
540 IF X<1 OR X>N THEN GOTO 530
550 CLS
```



```
560 LET K=1
570 GOTO 110
```

7) Modulo "INICIO"

```
  2 LET N=0
 14 LET K=0
 18 PRINT AT 6,0;"LISTA TELEFONICA"
 19 PRINT,,,,
 20 GOTO 200
```

Oitava aula

Para os equipamentos que não sejam da linha Sinclair,o comando LET pode ser suprimido. A linha 3Ø deve ser escrita GET Z$ nos equipamentos da linha Apple.

8.1 - Programa para desenhar na tela.

```
 1Ø LET X=32
 2Ø LET Y=22
 25 PLOT X,Y
 3Ø LET Z$=INKEY$
 4Ø IF Z$="5" THEN GOTO 15Ø
 5Ø IF Z$="6" THEN GOTO 2ØØ
 6Ø IF Z$="7" THEN GOTO 25Ø
 7Ø IF Z$="8" THEN GOTO 3ØØ
 8Ø IF Z$="1" THEN GOTO 35Ø
 9Ø IF Z$="2" THEN GOTO 4ØØ
1ØØ IF Z$="3" THEN GOTO 5ØØ
11Ø IF Z$="4" THEN GOTO 55Ø
12Ø GOTO 3Ø
15Ø LET X=X-1
16Ø IF X<Ø THEN LET X=Ø
```



```
17Ø GOTO 25
2ØØ LET Y=Y-1
21Ø IF Y<Ø THEN LET Y=Ø
22Ø GOTO 25
25Ø LET Y=Y+1
26Ø IF Y>43 THEN LET Y=43
27Ø GOTO 25
3ØØ LET X=X+1
31Ø IF X>63 THEN LET X=63
32Ø GOTO 25
35Ø LET X=X-1
36Ø LET Y=Y+1
37Ø IF Y>43 THEN LET Y=43
38Ø GOTO 25
4ØØ LET X=X+1
41Ø LET Y=Y+1
42Ø IF Y>43 THEN LET Y=43
43Ø GOTO 25
5ØØ LET X=X+1
51Ø LET Y=Y-1
```

```
520 IF Y<0 THEN LET Y=0

530 GOTO 25
550 LET X=X-1

560 LET Y=Y-1

570 IF Y<0 THEN LET Y=0

580 GOTO 25
```

Neste programa, nas linhas 10 e 20, armazenamos as coordenadas do ponto que será impresso no centro da tela.

A escolha das direções, nas quais poderemos deslocar o ponto que formará o nosso desenho, e obtida apartir da linha 30 ate a linha 120.

Descrição das rotinas:

a) A primeira, é para deslocar o ponto para a esquerda. Para isto, devemos diminuir de uma unidade o valor de X.

b) A segunda, é para deslocar o ponto para baixo. Agora devemos subtrair uma unidade do valor de Y.

c) A terceira, é para deslocar o ponto para cima. Para isto, devemos agora aumentar de uma unidade o valor de Y.

d) A quarta, é para deslocar o ponto para a direita. Para isto, mantemos fixo o valor de Y e variamos o valor de X.

e) A quinta, é para deslocar o ponto

para a esquerda e para cima. Neste caso, iremos diminuir o X e aumentar o Y.

f) A sexta, é para deslocar o ponto para a direita e para cima. Neste caso, iremos aumentar o valor de X e Y.

g) A sétima, é para deslocar o ponto para a direita e para baixo. Neste caso, iremos aumentar o valor de X e diminuir o valor de Y.

h) A oitava, é para deslocar o ponto para a esquerda e para baixo. Neste caso, iremos diminuir o valor de X e diminuir o valor de Y.

Depois que o desenho estiver pronto, para executarmos outro, devemos apertar a tecla BREAK e, em seguida, o comando RUN, para reiniciar. Uma alternativa para evitar isto, séria acrescentarmos uma rotina para apagar a tela.

Nona aula

9.1 - Exercícios.

1) Use o computador e calcule as suas retiradas na biblioteca. Informe o número de livros que você pegou emprestado e quantos dias cada um ficou emprestado. Faça o computador imprimir o total de sua conta se você paga CR$ 1Ø,ØØ por dia e por livro.

2) Considerando que você dorme 8 horas por dia, faça o computador imprimir quantas horas de sua vida você passou dormindo. Entre com os dados do seu nascimento e a data de hoje, na forma numerica, isto e: DDMMAA. (dia, mês e ano)

3) Bilhetes estão sendo vendidos para um show de ópera. Somente vinte estão disponíveis. Use o computador para registrar a venda de bilhetes. Se uma compra particular requerer mais bilhetes do que o número disponível, o computador deve dar um aviso de que existem menos bilhetes que os solicitados.

Se existirem bilhetes para atender o pedido particular, o computador deve aceitar a ordem e incrementar o contador de bilhetes vendidos. Quando todos os bilhetes forem vendidos, o computador devera´ imprimir: FIM DE VENDA.

4) Escreva um programa para imprimir um triângulo com o número N de linhas e N dígitos de altura, usando somente o número 1. Por exemplo: usando N=5, a saída seria como a da figura abaixo.

```
1
11
111
1111
11111
```

ANEXO II - Solução dos exercícios.

Obs. Os exercícios estão resolvidos para a linha Sinclair. Para outros equipamentos, você próprio deve fazer as adaptações a título de exercício.

Aula 1:

1-a) SIM, NAO, SIM, SIM

SIM, NAO, NAO, SIM

1-b) Três -> uma letra, duas letras e uma letra e um numero.

2)
```
10 INPUT X
20 INPUT Y
30 LET A = 12 * X + 7 * Y
40 PRINT A
50 GOTO 10
60 STOP
```

3)
```
10 PRINT"QUANTOS PEDACOS VOCE COMEU ?"
20 INPUT P
```



```
30 LET M = P * 3.75
40 PRINT
50 PRINT"VOCE DEVE CORRER ";M;
" MILHAS"
60 PRINT
70 GOTO 10
80 STOP
```

4)
```
10 PRINT"ENTRE COM A LARGURA"
20 INPUT L
30 PRINT"ENTRE COM O COMPRIMENTO"
40 INPUT C
50 PRINT"ENTRE COM A ALTURA"
60 INPUT A
70 LET V = L * C * A
80 PRINT
90 PRINT"O VOLUME E: ";V
100 PRINT
110 GOTO 10
120 STOP
```

5)
```
10 PRINT"ENTRE COM O PRIMEIRO NUMERO"
```

```
 2Ø INPUT N1
 3Ø PRINT"ENTRE COM O SEGUNDO NUMERO"
 4Ø INPUT N2
 5Ø PRINT"ENTRE COM O TERCEIRO NUMERO"
 6Ø INPUT N3
 7Ø PRINT"ENTRE COM O QUARTO NUMERO"
 8Ø INPUT N4
 9Ø LET M = (N1 + N2 + N3 + N4) / 4
1ØØ PRINT
11Ø PRINT"A MEDIA E: ";M
12Ø PRINT
13Ø GOTO 1Ø
14Ø STOP
```

6)

```
 1Ø PRINT"ENTRE COM O PRECO TOTAL"
 2Ø INPUT P
 3Ø PRINT"ENTRE COM A QUANTIDADE TOTAL"
 4Ø INPUT Q
 5Ø LET U = P / Q
 6Ø PRINT
```



```
 7Ø PRINT"O PRECO UNITARIO E: ";U
 8Ø PRINT
 9Ø GOTO 1Ø
1ØØ STOP
```

7)

```
 1Ø PRINT"ENTRE COM GRAUS F"
 2Ø INPUT F
 3Ø LET C = (5 / 9) * (F - 32)
 4Ø PRINT
 5Ø PRINT F;" GRAUS F SAO ";C;" GRAUS
C"
 6Ø PRINT
 7Ø GOTO 1Ø
 8Ø STOP
```

8)

```
 1Ø PRINT"ENTRE COM A"
 2Ø INPUT A
 3Ø PRINT"ENTRE COM B"
 4Ø INPUT B
 5Ø LET S = A + B
 6Ø LET P = A * B
```

```
70 PRINT
80 PRINT"A SOMA DE A COM B E: ";S
90 PRINT"O PRODUTO DE A COM B E: ";P
100 PRINT
110 GOTO 10
120 STOP
```

Aula 2:

1)
```
10 FOR X=10 TO 40 STEP 2
20 IF (3/5) * X**2 + 2*X + 2/5 > 300
THEN GOTO 40
30 GOTO 50
40 PRINT X;"  ";
50 NEXT X
60 STOP
```

2)
```
10 PRINT"ENTRE COM A SUA IDADE"
20 INPUT I
30 IF I >= 16 THEN GOTO 60
40 PRINT"VOCE DEVE ESPERAR ";16-I;
" ANOS"
```



```
50 GOTO 70
60 PRINT"VOCE PODE DIRIGIR UM CARRO"
70 PRINT
80 GOTO 10
90 STOP
```

3)
```
10 PRINT"N","SQR(N)"
20 PRINT
30 FOR N=1 TO 10
40 PRINT N,SQR(N)
50 NEXT N
60 STOP
```

4)
```
10 LET X=0
20 PRINT X;"  ";
30 LET X = X+10
40 IF X > 500 THEN STOP
50 GOTO 20
60 STOP
```

5)
```
10 FOR A=1 TO 10
```

```
20 PRINT"ANDAR ";A
30 PRINT
40 FOR B= 1 TO 10
50 LET C = A * 100 + B
60 PRINT"APOSENTO NUM. ";C
70 NEXT B
80 NEXT A
90 STOP
```

6)
```
10 PRINT"POLEGADAS","CENTIMETROS"
20 PRINT
30 FOR P=0 TO 10 STEP 0.5
40 LET C = P * 2.54
50 PRINT P,C
60 NEXT P
70 STOP
```

Aula 3:

1) Uma variável STRING e identificada pelo símbolo ($) após uma letra. Exemplo: A$

2)
```
10 FOR X=1 TO 3
20 PRINT TAB(10);"XX"
30 NEXT X
40 FOR X=1 TO 2
50 PRINT TAB(10);"XXXXXX"
60 NEXT X
70 STOP
```

3)
```
10 LET X=0
20 INPUT A$
30 PRINT TAB(X);A$
40 IF X=42 THEN GOTO 10
50 LET X=42
60 GOTO 20
70 STOP
```

4)
```
10 PRINT"ENTRE UMA PALAVRA"
20 INPUT A$
30 FOR X=1 TO LEN A$
40 PRINT TAB(X);A$(X)
50 NEXT X
```

```
60 STOP
```

5)
```
10 PRINT"ENTRE UMA PALAVRA"
20 INPUT A$
30 FOR X=1 TO LEN A$
40 PRINT A$(X)
50 NEXT X
60 STOP
```

6)
```
10 PRINT"ENTRE UMA PALAVRA"
20 INPUT A$
30 FOR X=1 TO LEN A$
40 PRINT A$(X TO LEN A$)
50 NEXT X
60 STOP
```

Aula 4:

1) Devemos usar o comando RAND.

2) Devemos utilizar a função RND.

3)
```
10 FOR X=1 TO 10
20 LET R=INT(26*RND)+50
30 PRINT R;"  ";
40 NEXT X
50 STOP
```

4)
```
10 FOR X=1 TO 10
20 LET R=INT(6*RND)+1
30 PRINT"FACE- ";R
40 NEXT X
50 STOP
```

Aula 5:

1) Reservar espaço na memória para armazenamento das variáveis.

2) X(3,4)

3)
```
10 PRINT"ENTRE COM AS DIMENSOES DE A(I,J)"
20 PRINT"DIGITE O VALOR DE I"
```

```
30 INPUT I
40 PRINT"DIGITE O VALOR DE J"
50 INPUT J
60 DIM A(I,J)
70 FOR X=1 TO I
80 FOR Y=1 TO J
90 PRINT"ENTRE COM O VALOR DE A(";X;
",";Y;")"
100 INPUT A(X,Y)
110 NEXT Y
120 NEXT X
130 LET TL=0
140 LET TC=1
150 FOR X=1 TO I
160 FOR Y=1 TO J
170 LET TL=TL + A(X,Y)
180 NEXT Y
190 PRINT"SOMA DA LINHA ";X;": ";TL
200 LET TL=0
210 NEXT X
```



```
220 FOR Y=1 TO J
230 FOR X=1 TO I
240 LET TC=TC * A(X,Y)
250 NEXT X
260 PRINT"PRODUTO DA COLUNA ";Y;": ";
TC
270 LET TC=1
280 NEXT Y
290 STOP
```

4) a- A(2,3)

b- A(2,3)=4

c- A(1,2)=3

d- A(1,2)=3

Aula 9 :

1)

```
5 LET T=0
10 PRINT"QUANTOS LIVROS EMPRESTADOS"
20 INPUT X
30 FOR I=1 TO X
40 PRINT"QUANTOS DIAS O LIVRO ";I;
```

```
" FICOU EMPRESTADO"
     5Ø INPUT Y
     6Ø PRINT
     7Ø LET T=T+Y*1Ø+1Ø
     8Ø NEXT I
     9Ø PRINT"TOTAL DA CONTA: CR$ ";T
    1ØØ GOTO 1Ø
```

2)

```
    1Ø PRINT"DATA DE NASCIMENTO (DDMMAA)"
    2Ø INPUT N
    3Ø PRINT
    4Ø PRINT"DATA DE HOJE (DDMMAA)"
    5Ø INPUT H
    6Ø LET DN=INT(N/1ØØØØ)
    7Ø LET MN=INT(N/1ØØ)-DN*1ØØ
    8Ø LET AN=N-(DN*1ØØØØ+MN*1ØØ)
    9Ø LET D1=DN+MN*3Ø+AN*36Ø
   1ØØ LET DH=INT(H/1ØØØØ)
   11Ø LET MH=INT(H/1ØØ)-DH*1ØØ
   12Ø LET AH=H-(DH*1ØØØØ+MH*1ØØ)
```



```
   13Ø LET D2=DH+MH*3Ø+AH*36Ø
   14Ø LET X=(D2-D1)*8
   15Ø PRINT"VOCE DORMIU ATE ";H;" ";X;
" HORAS"
   16Ø PRINT
   17Ø PRINT
   18Ø GOTO 1Ø
```

3)

```
    1Ø LET K=2Ø
    2Ø PRINT"QUANTOS BILHETES DESEJA ?"
    3Ø INPUT N
    4Ø IF N>K THEN GOTO 8Ø
    5Ø LET K=K-N
    6Ø IF K=Ø THEN GOTO 12Ø
    7Ø GOTO 1Ø
    8Ø PRINT"SO EXISTEM ";K;" BILHETES
DISPONIVEIS"
    9Ø PRINT
   1ØØ IF K=Ø THEN GOTO 12Ø
   11Ø GOTO 1Ø
   12Ø PRINT"FIM DE VENDA"
   13Ø PRINT
```

```
140 STOP
```

4)

```
10 PRINT"INDIQUE A ALTURA DO TRIANGU
LO"
20 INPUT N
30 FOR I=1 TO N
40 FOR J=1 TO I
50 PRINT"1";" ";
55 NEXT J
60 PRINT
70 NEXT I
80 PRINT
90 GOTO 10
100 STOP
```

Impresso na
ERCA Editora e Gráfica Ltda.
Rua Silva Vale, 870 - Cavalcante
Rio de Janeiro - RJ

OBRAS DO AUTOR:

Linha Sinclair

1 — Curso Basic V. 1
2 — Curso Basic Avançado V. 2
3 — 47 Programas para Spectrum e TK90X

Linha Apple

1 — 77 Programas para linha Apple
2 — Programas Comerciais da linha Apple — V. 1
(Mala direta, Contas a pagar e receber. Controle de estoque)
3 — Programas Comerciais da linha Apple — V. 2
(Arquivos, Cadastro de clientes com emissão de fatura/duplicata, Controle de vendas)
4 — Usando o Visiplot
5 — Usando o Assembler 6502
6 — Usando o Processador de texto Magic Window
7 — Usando as Rotinas internas da Apple

Linha TRS

1 — 77 Programas para a linha TRS

Linha MSX

1 — 77 Programas para a linha MSX