CURSO DE INFORMÁTICA

1

CURSO DE INFORMÁTICA

BASIC

EDROS
EDITORA ROSA E SILVA LTDA

Curso de Informática

BASIC

AGRADECIMENTO

Nossos agradecimentos ao Editor, Sr. F. da Silva Neto, por sua preciosa colaboração, o que fez tornar realidade a edição desta nossa obra.

O Autor

Wenceslau Carlos Galvão Filho

Curso de Informática

BASIC

FUNDAMENTAL
NÍVEL I e II

VOLUME 1

EDROS — Editora Rosa e Silva Ltda.
São Paulo

Editor — F. da Silva Neto.
Colaborador especial:
Abel De Mariano Camargo.

INTRODUÇÃO

As regras estabelecidas pelo Computador e que regerão o nosso Progresso e Bem Estar nos dias que se seguirão daqui para a frente, deverão ser compreendidas e estudadas por nós.

Vejamos todas as perspectivas que teremos para melhorar e muito nossa vida profissional com o uso do Computador.

Hoje lemos e ouvimos falar que na área da informática está tudo em franco desenvolvimento; o comércio, a indústria, a agricultura, etc., sendo a área de informática a única que desconhece a crise.

Pois bem, já estamos entrando em uma nova era, a era da computação e informática, não podemos desconhecer seus princípios, suas regras e sua aplicação nas várias áreas de nosso conhecimento.

O COMPUTADOR NÃO FAZ NADA DO QUE QUEREMOS, A NÃO SER QUE QUEIRAMOS O QUE ELE FAZ.

ABERTURA

Você poderá fazer um curso de computação sem precisar do computador.

Através deste curso você terá todos os conhecimentos necessários para resolver os problemas em um computador.

Assim, toda a teoria matemática, desde o início até o final está dirigida às operações matemáticas, a programação e finalmente à análise matemática e também os termos específicos usados pelo computador, em inglês.

Deste modo você poderá escolher quaisquer das profissões na área da informática, tais como:

— Operador de máquina
— Digitador
— Programador
— Analista

Portanto, vamos começar a nos inteirar sobre o assunto.

Os exercícios têm por objetivo a transferência de todo o conhecimento teórico para a prática ou seja para o uso do computador na solução dos problemas

através da programação dos dados, em linguagem de montagem e assim a obtenção dos resultados pelo computador.

Portanto a primeira linguagem a ser estudada é a:

BASIC

Uma vez adquiridos todos esses conhecimentos, você estará apto a ingressar na era da **Informática**, investido de todos conhecimentos fundamentais e com os quais poderá deduzir todas as outras linguagens, e também poderá fazer uso de qualquer tipo de computador.

A NOVA INGUAGEM; A DA INFORMÁTICA

Em primeiro lugar precisamos justificar esse título.

O que vem a ser esta nova linguagem?

Então tudo o que estudamos terá que ser modificado?

Através deste curso iremos demonstrar o que mudou e o que continua.

Será fácil de entender, pois essa mudança está se processando com a introdução do computador em nosso convívio e em todas as áreas de nossas atividades, quer sejam comerciais, didáticas, industriais, etc.

É por essa razão que se torna imperativo essa nova linguagem, que nada mais é senão a linguagem de montagem da programação dos nossos problemas, para serem resolvidos em fração de segundos pela máquina; máquina essa que é: o Computador.

Para essa mudança de hábitos e conhecimentos temos a necessidade de começar um novo estudo, desde o seu princípio.

Assim teremos que estudar os novos significados das palavras, das construções de frases, etc.

Estas serão novamente abordadas para serem comparadas aos novos significados.

É com este critério que iremos estudar e entender esta nova linguagem.

Vejamos um exemplo para facilitar a compreensão.

Temos a palavra rotina, que é usada para indicar a repetição sistemática de um trabalho ou fato que nós realizamos.

Pois bem, essa mesma palavra na informática significa um trabalho ou evento escrito e gravado em linguagem de máquina que nós não precisamos mais nos preocupar com a sua programação, pois a hora em que nós precisarmos fazer uso desse trabalho é só colocá-lo no computador e pronto, estaremos fazendo seu uso para a solução do problema.

É importante saber que isto não ocorre somente com o nosso Português mas também com todos os outros idiomas.

DEFINIÇÃO DE LINGUAGEM NA INFORMÁTICA

A linguagem é um conjunto de representações simbólicas, de convenções e regras usadas para a comunicação e processamento das informações.

Este conjunto é definido para cada tipo de linguagem.

Temos assim as linguagens, que recebem os nomes segundo as suas aplicações, tais como:

— Linguagem de Montagem
— Linguagem Simbólica
— Linguagem de Máquina, etc.

LINGUAGEM DE PROGRAMAÇÃO

Esta é a linguagem usada para montar um programa ou problema a ser resolvido pelo computador.

Ela faz o uso de todos os caracteres alfanuméricos, especiais e da lógica.

As duas principais linguagens são

— Linguagem de Montagem.
— Linguagem de Máquina.

LINGUAGEM DE MONTAGEM
(Assembly language)

É a linguagem que interpreta um programa em linguagem simbólica para a linguagem de máquina.

LINGUAGEM DE MÁQUINA

Chama-se linguagem de máquina a linguagem usada diretamente pelo computador.

Chama-se também de linguagem orientada para a máquina.

Esta linguagem é variável segundo a estrutura e projeto lógico da construção do computador e ela pode ser deduzida dos conhecimentos de comandos de instruções das operações de instruções indicadas em seu teclado e pelos manuais dos computadores.

PALAVRAS — (WORDS)

São definidas como palavras na Informática um conjunto de caracteres ou série de bits considerados como uma unidade.

Este conjunto pode conter um número de caracteres não superior a 30.

Esses caracteres são: as letras do alfabeto inglês, os dígitos de 0 a 9 e o travessão (—) e também os dígitos do sistema Hexadecimal.

As leis e regras que regem a organização dos caracteres na formação das palavras são definidas pelas linguagens de programação tais como, BASIC, COBOL, etc.

Usa-se também como sinônimo de palavra, **grupo.**

Veremos que existe também um conjunto de **palavras proibidas** em cada linguagem ,assim como outro conjunto de **palavras reservadas.**

VOCABULÁRIO — (VOCABULARY)

É o conjunto de palavras definidas especificamente para o uso no computador, formadas por um conjunto de códigos de operação ou instruções, usado para escrever um programa.

FRASES E SENTENÇAS

As frases e sentenças de nossa linguagem são substituídas por expressões matemáticas e strings.

TEXTO

Texto é aqui substituído por programa.

PROGRAMA

Programa é o conjunto lógico de instruções e comandos em seqüência ou aleatório que um computador pode executar para a solução de um problema específico.

SINTAXE

A sintaxe de construção de um programa segue as regras e leis definidas pela lógica da linguagem

usada na montagem de um programa ou da programação, e quando estas regras ou leis não são seguidas o computador acusa o **ERROR** e onde ele está acontecendo.

ERROS — (ERROR)

Palavra de caracter geral usado para indicar que o significado ou valor do dado não está correto ou não segue a lógica ou condição de verdadeiro.

ACENTUAÇÃO E PONTUAÇÃO

A acentuação não existe nas palavras de máquina.

PONTUAÇÃO

As regras de pontuação seguem um determinado critério estabelecido pela construção do computador as quais veremos no decorrer do curso de programação e computação.

Atenção: O uso do computador como simples máquina de operações de gravar tanto no Vídeo como nas memórias externas, é neste caso uma linguagem comum.

Na linguagem usada pelo processador central todas as regras e leis são absolutamente rígidas.

Vejamos um exemplo:

Através do comando de **Edição** podemos escrever qualquer programa e em qualquer linguagem executar a sua gravação em qualquer memória externa.

Todas estas operações são realizadas nos periféricos ou software.

Agora se nós quisermos realizar o processamento desses dados no processador interno então vere-

mos que todas as mensagens ou dados terão que seguir as leis, regras e a lógica da programação sem o que não haverá nem um resultado.

Concluimos assim que uma mensagem que não segue as leis da linguagem de máquina não pertence a programação mas simplesmente à literatura.

LINGUAGEM DE PROGRAMAÇÃO

Vamos demonstrar através deste curso a aplicação do conhecimento matemático e da lógica na programação de dados de um problema a ser resolvido pelo computador.

Dependendo do tipo de problema teremos o uso da linguagem de montagem.

Através das definições de comandos e instruções e suas aplicações no computador iremos compreender mais facilmente esse assunto.

Veremos também que a aprendizagem do uso do computador não exige apenas o manuseio da máquina, mas também das linguagens.

TIPOS DE LINGUAGENS

Temos dois tipos de linguagens.

A de nível I e II, que é a **low-level language.**

Esta é a linguagem usada nas máquinas de calcular, onde os dados são introduzidos um a um obedecendo a uma seqüência lógica de operações.

Temos a outra que é a de nível superior **high-level language,** traduzida por linguagem de nível superior ou alto nível, que é caracterizada por sua independência em relação a linguagem de máquina.

BASIC é acrônimo de **B**egineer's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode.

Código simbólico de instruções para principiantes de todos os propósitos ou fins.

Importante:

O nosso computador só entende o inglês e em linguagem de máquina; por essa razão teremos que estudar os comandos e instruções usadas na programação em Inglês.

INTRODUÇÃO À PROGRAMAÇÃO EM BASIC

DEFINIÇÃO DE PROGRAMA

Programa é um conjunto lógico de instruções que um computador pode executar numa seqüência definida para resolver um problema específico.

DEFINIÇÃO DE PROGRAMAÇÃO

Programação é a escrita simbólica redigida em linguagem própria para expressar programas de computação.

Com estas definições podemos iniciar o nosso estudo com a primeira linguagem de programação que é o BASIC ,conforme já dissemos.

Chama-se programa BASIC aquele escrito em código de instruções BASIC, pronto para ser executado pelo Computador.

COMANDOS DE LINGUAGEM BASIC

Comandos de Controle.

Comando é a instrução em linguagem de máquina.

Temos dois tipos de comandos. O primeiro é o realizado pelo operador da máquina através do programa que para isso precisa conhecer perfeitamente a função do comando e da instrução, pois qualquer erro terá como resultado o: **Nada.**

O segundo tipo é o comando automático que será realizado pelo computador segundo a lógica para a qual ele foi construído.

Aqui as operações são feitas automaticamente pelas indicações corretas do primeiro comando ou seja as do operador.

O BASIC

A linguagem BASIC pode ser encontrda no SYSTEM, isto é, o computador aceita o BASIC diretamente e, quando isto não ocorre, ele poderá ser pedido através de comandos e instruções especiais. (Obs.: Este fato vem indicado nos manuais do computador).

EXEMPLO DE COMO PREPARAR UM DETERMINADO COMPUTADOR PARA O BASIC

Quando você liga o computador aparece no Vídeo a pergunta:

BASIC (S ouN)?

Que significa a escolha entre o BASIC e a linguagem SYSTEM.

Digitando o **S** você escolhe o BASIC e o computador estará operando em BASIC.

Após a resposta digitada S aparece outra pergunta:

C a s s ?

Esta se refere a velocidade de transferência dos dados do computador para as memórias externas (disco, disquete, fita, etc.) e vice-versa.

Essas velocidades são normalmente oferecidas em duas opções: alta A e baixa B.

Para a baixa pressiona-se a tecla B.

Para a velocidade alta pressiona-se A ou a tecla ENTER. (Estas convenções são indicadas pelo manual do Computador em uso).

Após esta escolha aparece a última pergunta:

Mem. Usada?

Esta se refere ao limite máximo de endereçamento da memória RAM que o Interpretador BASIC poderá usar.

A resposta será ENTER que será entendida pelo computador o uso de toda sua capacidade de armazenamento RAM. Temos então o computador pronto para receber a programação cuja mensagem emitida na tela é:

— I n I I (a marca do computador)
READY — (pronto).
20 PRINT A + B * 5 − C/2 + 7
30 LET P = A + B * 5 − C/2 + 7
40 END
RUN.

OBSERVAÇÕES

Daqui para frente vamos encontrar todas as palavras escritas pelo computador sem a acentuação e sinais das palavras, pois o computador não reconhece a fonética nem a regra de pronúncia, isto é, a semântica.

DEFINIÇÃO DOS CARACTERES PERMITIDOS EM BASIC

Temos todas as letras do alfabeto incluindo K, Y e W, e os dígitos de 0 a 9, além destes temos:

ARROBA @
APOSTROFO
ASTERISCO * (estrela)
BARRA / (sinal de divisão)
BARRA INVERTIDA \
CIFRÃO $
DOIS PONTOS :
"E" COMERCIAL &
ENTRE ASPAS " "
ENTRE PARENTESES ( )
ENTRE COLCHETES [ ]
ESPAÇO - ∅
PONTO .
PONTO DE EXCLAMAÇÃO !
PONTO DE INTERROGAÇÃO ?
PONTO E VÍRGULA ;
RETICÊNCIAS ...
SÍMBOLO DE NUMERAL §
SÍMBOLO DE PORCENTAGEM %

SINAIS MATEMÁTICOS

SUBLINHADO ——
MAIS +
MENOS −
MULTIPLICAÇÃO *
DIVISÃO /
POTENCIAÇÃO — [ou * *

SÍMBOLOS DE RELAÇÃO

IGUAL OU ATRIBUIÇÃO =
MAIOR QUE >
MENOR QUE <
MAIOR OU MENOR QUE < >
MAIOR OU IGUAL A > =
MENOR OU IGUAL A = <

DESTES CARACTERES SÃO CONSIDERADOS ESPECIAIS OS SEGUINTES:

vírgula , ponto e vírgula ;
ponto . dois pontos :
ponto de interrogação ?
ponto de exclamação !
espaço em branco ∅
aspas " "
abre parênteses (
fecha parênteses)
abre colchetes [
fecha colchetes]

Lembre-se que na gravação dos caracteres pelo computador as "palavras" não possuem os nossos conhecidos acentos.

CAPACIDADE DE LINHA DE MEMÓRIA

N CARACTERES — (N variável para cada tipo de computador).

Quando uma linha tiver 2 ou mais instruções estas devem ser separadas por dois pontos (:).

INSTRUÇÕES OU COMANDOS

Uma instrução ou comando determina quais as operações que devem ser executadas.

Ex.: STOP para a execução liberando o comando.

EXPRESSÕES QUE REPRESENTAM OS DADOS

O conjunto de informações fornecidas ao computador são feitas por expressões cuja classificação é feita pela sua natureza através das representações dos dados.

REPRESENTAÇÃO DOS DADOS

Os dados de uma expressão são representados por constantes e variáveis.

CONSTANTES:

Todas as informações que não estão sujeitas a modificações durante o processamento são tratadas como constantes.

Ex.: 5 10

5 + 10

"A raiz quadrada de 4E"; 2

As constantes de um programa podem ser numéricas ou Strings.

CONSTANTES NUMÉRICAS

São consideradas constantes numéricas, todos os números inteiros e fracionários com até 7 algarismos.

Os números fracionários com até 6 algarismos são tratados como constantes.

Ex.: 578

3,7502

− 35,201

CONSTANTES STRINGS

Todas as informações ou dados alfanuméricos são considerados constantes strings e vem obrigatoriamente entre aspas.

Ex.: "Tratado"

"JOÃO MORA ALI"

"RUA 7 DE SETEMBRO"

VARIÁVEIS

São tratados como variáveis todos os dados e informações que podem sofrer alterações dos valores durante o processomento.

Temos dois tipos de variáveis: numéricas e strings.

VARIÁVEIS NUMÉRICAS

São tratados como variáveis numéricas todos os valores representados por números fracionários dizimas, ou números irracionais.

Com estes conjuntos de números teremos que considerar um fator: a precisão.

Com respeito a precisão o processador considera duas condições de precisão:
simples precisão e dupla precisão.

SIMPLES PRECISÃO

Quando o número contiver até 7 algarismos decimais ele será tratado como número de simples precisão.

Ex.: 5,7897
3.00002
3891234

DUPLA PRECISÃO

São considerados números de dupla precisão todos os números formados de 7 a 17 algarismos.

Ex.: 7 8 9 1 0 5 8 7 8
3 2 9 1 0 1 0 0 0 3 4
– 3 4 5 6 7 8 0 1 9

VARIÁVEIS STRINGS

Todos os dados de uma expressão que podem sofrer variações tanto em valor numérico como valor de significado recebem o nome de variáveis.

Ex.: Uma instrução representada pela expressão:

A $ = "BOLA"

Agora se representarmos no mesmo programa o nome da variável A$ com outro conteúdo, teremos:

A$ = "VERDE"

então em lugar de

A$ = "BOLA"

teremos:

A$ = "VERDE"

NOME DE VARIÁVEIS

Em BASIC as variáveis são representadas por nomes que começam por uma letra do alfabeto de A a Z, ou ainda por uma letra seguida de um número ou ainda outra letra.

Ex.: A A2 A3
B B1 AX

(Obs.: existe versões que não aceitam 2 letras)

EXPRESSÃO

Chama-se expressão o conjunto de dados de um programa para ser processado pelo computador.

Temos 4 tipos de expressões: Numéricas, Strings, Relacionais e Funções.

EXPRESSÕES NUMÉRICAS

São expressões aritméticas formadas por números e sinais de operações binárias.

Ex.: (3 + 5) (10 + 2/)
4 − 8 − 9 + 10 2 − 5 — 7

EXPRESSÕES STRINGS

São expressões alfanuméricas compostas de letras, numerais e sinais gráficos que vêm obrigatoriamente entre aspas.

Ex.: "Pedro mora"
"2 mais 2 igual a", 4

Operadores e operações de cálculo e classificação de dados

OPERADORES

Os operadores são definidos por símbolos ou palavras-chave que indicam qual a operação a ser realizada entre dois valores especificados ou um único valor.

OPERAÇÕES BINÁRIA E UNÁRIA

Quando o operador age entre 2 operandos, se diz operação binária e quando age somente em um operando se diz unária ou operação monádica.

Ex.: operação binária.

a + b a + c
a − b

operação unária
operador operando

− ou + a
− 5 + 5

o operador age nos operandos + 5 ou − 5

Temos quatro categorias para os operadores, que são: Numéricos, Strings, Relacionados e Lógicos.

OPERADORES NUMÉRICOS

São operadores usados exclusivamente com expressões numéricas.

Esses operadores já conhecidos nossos são:

+ adição — subtração

[ou ** ou potenciação.

* multiplicação / divisão

OPERADOR STRING

O único operador string disponível nesta linguagem é + (mais), o qual permite unir duas ou mais Strings.

Esta operação indica a união de strings em série e recebe o nome de concatenação.

Ex.: PRINT "PEDRO" + "MORA" + "LA" — resultado: PEDRO MORA LA ?

EXPRESSÕES RELACIONAIS

São expressões que estabelecem condições de relações lógicas entre duas ou mais expressões.

Estas relações podem ser numéricas ou strings.

RELAÇÕES NUMÉRICAS

Os operadores que estabelecem estas relações são:

> maior que

< menor que

= igual

= > ou > = maior ou igual a

= < ou < = menor ou igual a

< > ou > < maior ou menor que → diferente

RELAÇÕES STRINGS

Os sinais relacionais entre as expressões strings indica a seqüência no código ASCII das expressões.

Assim teremos os dados classificados por prioridade, isto é, o que tiver o menor número no código ASCII terá prioridade sobre o seguinte;

Assim —
menor prioridade > maior prioridade
maior prioridade < menor prioridade
String A < > ou > < String B
não tem relação de prioridade, são de níveis diferentes.

prioridade menor ou igual >= ou = > prioridade maior ou igual
< = ou = < conclua Você esta relação.

Exemplo de relação:
String.
"B" > "C" N.° de B no Código ASCII é 66
N.° de C no Código ASCII é 67

OPERADORES LÓGICOS

São operadores que fazem comparações lógicas entre duas ou mais expressões relacionais, formando expressões lógicas.

Estes operadores seguem a lógica da álgebra de Boole e são os operadores OR, AND e NOT, e seus derivados NOR, NAND.

HIERARQUIA DOS OPERADORES

A hierarquia das operações seguem as leis da hierarquia das operações matemáticas já nossa conhecida.

Assim quando as operações são do mesmo nível não haverá indicação especial, serão feitas na ordem indicada da esquerda para a direita.

Quando as expressões contiverem níveis diferentes, então as operações serão indicadas por

FUNÇÕES

As expressões que representam funções são subrotinas automáticas e são tratadas como qualquer outra informação e cuja execução é conseqüência de uma instrução comum.

Não há necessidade de se escrever uma rotina em qualquer linguagem, basta uma palavra chave que identificará cada função contida no computador.

Ex.:

S Q R (B + 10)

Esta função calcula a raiz quadrada da expressão (B + 10) entre parênteses.

CARACTERES PROIBIDOS

São proibidos na linguagem BASIC como **variáveis todas as palavras de comandos ou instruções** e ainda as palavras que as contenham na sua sintaxe.

Ex.: FOR comando FORTE
END comando ENDEREÇO
AND comando MANDAR

As palavras de comandos ou instruções de uma linguagem de montagem são definidas como palavras **reservadas** à essa determinada linguagem.

Concluimos assim que uma seqüência de caracteres igual a um comando não podem ser usadas como variável.

RELAÇÃO

*	ELSE	LLIST	RESET
ABS	END	LPRINT	RESTORE
AND	ENTER	LOAD	RESUME
ASC	EOF	LOF	RETURN
ATN	ERL	LOG	RIGHT$
AUTO	ERR	LOC	RND
CDBL	ERROR	MEM	RSET

CHR$	EXP	MERGE	RUN
CINT	FIELD	MID$	SAVE
CLEAR	FIX	MKD$	SET
CLOCK	FN	MKI$	SGN
CLOSE	FOR	MIKS$	SIN
CLS	FORMAT	NAME	SQR
CMD	FRE	NEW	STEP
CONT	FREE	NEXT	STOP
COS	GET	NOT	STRING$
CR	GOSUB	ON	STR$
CSNG	GOTO	OPEN	SYSTEM
CVD	IF	OR	TAB
CVI	INKEY$	OUT	TAN
CVS	INP	PEEK	THEN
DATA	INPUT	POINT	TIME
DEFDEL	INSTR	POKE	TO
DEFFN	INT	PO S	TROFF
DEFINT	KILL	POSN	TRON
DEFSNG	LEFT$	PRINT	USING
DEFUSR	LET	PUT	USR
DEFSTR	LSET	RANDOM	VAL
DELETE	LEN	READ	VARPTR
DIM	LINE	REM	VERIFY
EDIT	LIST	RENAME	PAL

COMANDOS E INSTRUÇÕES BASIC

INSTRUÇÃO CLS

Quando existir um programa na tela e vamos usá-la para outro programa, digitamos CLS e este comando limpa a tela.

COMANDO CLEAR

Este comando zera todas as variáveis numéricas e anula as variáveis strings.

COMANDO RESET

Este comando devolve o controle à condição inicial do programa. Desativa uma posição especificada pelas coordenadas (x, y) na tela, colocando em zero.

COMANDO SET

Ativa a posição especificada na telas pelas coordenadas (x, y).

CAPACIDADE DA TELA NO VÍDEO

A tela está dividida em 16 linhas.

Os caracteres podem ser de duas larguras: normal e dupla.

Cada linha pode conter 64 caracteres, largura normal, ou 32 caracteres de largura dupla cuja indicação é: CPI caracteres por polegadas.

A mudança d alargura normal para a dupla é feita pelo comando SHIFT □
e para o retorno da largura normal pelo SHIFT □ em determinado computador.

MEMÓRIAS EXTERNAS OU ARMAZENAMENTOS DE DADOS

Os gravadores externos são acoplados ao Hardware para o uso de discos, disquetes ,fitas, etc., possibilitando o uso de programas já gravados em linguagem de máquina ou não, evitando assim a repetição de digitação de programas já executados.

SISTEMAS OPERACIONAIS DE COMPUTADOR

DOS — Disk Operating System — sistema operacional de discos ou fitas.

Uma vez ligado o computador aparece uma mensagem na tela.

A gravação de um programa pode ser feita de duas maneiras:

PRIMEIRA

O programa está em linguagem simbólica.

Neste caso ele é introduzido no computador através da digitação, que por ele é aceito tal qual a digitação feita com erros e tudo.

É um caso de Edição de linhas.

Através dos comandos ele é listado na tela, gravado na fita ou disco, ou impresso pela impressora.

Este programa não é feito para ser processado pelo computador. Os comandos usados neste processo serão definidos no decorrer do Curso.

SEGUNDA

Temos um programa escrito em linguagem de programação para ser processado pelo computador; então ele terá que estar dentro das regras e leis que regem o seu processamento segundo a construção do computador.

O USO DA LINGUAGEM BASIC POR UM DETERMINADO COMPUTADOR

Todo computador é acompanhado por um manual prático da linguagem BASIC e contém desde a instalação, operações e seu funcionamento, até a finalização do programa.

MEMÓRIAS INTERNAS — RAM E ROM

RAM — Random Access Memory — memória de acesso aleatório ou randônico.

ROM — Read Only Memory — memória apenas de leitura.

CAPACIDADES DE MEMÓRIAS

Em determinado computador a memória ROM armazena 16 KB (kilobytes).

1 KB vale 1024 bytes que são 16 * 1024 caracteres ou bytes de programação permanente.

A memória RAM armazena seus programas, resultados e os dados referentes a eles. A sua capacidade de armazenamento da RAM é de 48 KB, portanto 3 vezes maior que a ROM.

Os dados e informações armazenados na memória RAM permanecem somente quando o computador está em uso e assim quando se desliga ela perde todas as informações.

USANDO A LINGUAGEM BASIC NO COMPUTADOR

Quando se liga o computador e se chama a linguagem BASIC no interpretador de linguagem, teremos no Vídeo, pressionando-se a tecla RESET, a mensagem BASIC (S ou N)?

Você escolhe o BASIC digitando S já visto.

ENTRADA DE UM PROGRAMA EM LINGUAGEM BASIC NO COMPUTADOR

Após as operações de inicialização introduzimos o programa através do teclado por digitação das teclas de comando e dos caracteres.

Veremos que o BASIC não aceita espaços em branco dentro de um linha, isto é, os caracteres digitados serão sempre consecutivos. E estando no estado de execução, se o primeiro caráter da linha for um número o interpretador tratará como linha de programa e esta vai para a memória de programas; caso contrário considera-se como linha imediata e a executa.

Ex.: **linha de programa.**

```
20 PRINT A, B, C
   (vai para a memória de programa).
30 PRINT 2, 3
40 END
   RUN
   A B C
   2, 3
```

Ex.: **linha imediata.**

```
PRINT "10"
ENTER
10
```

Obs.: Quando você usa um mesmo número de outra linha esta é eliminada e a nova linha é guardada na memória.

Esta propriedade é usada para a correção de uma linha ou do texto.

Ex.: Se você cometeu um erro na linha 110 então a correção será feita do seguinte modo:

110 PRINT A + B + C e o certo é A + B
basta digitar:

110 PRINT A + B e a correção será feita.

A primeira linha 110 é substituída pela segunda linha 110.

CAPACIDADE DE LINHA DE MEMÓRIA

N Caracteres — N número variável para cada tipo de computador.

Quando uma linha tiver 2 ou mais instruções estas devem ser separadas por dois pontos (:).

Instruções ou Comandos

Uma instrução ou comando determina quais as operações que devem ser executadas.

Ex.: STOP para a execução, imprime a mensagem com o número da linha onde foi interrompido o programa.

A retomada da execução pode ser feita com a instrução CONT.

O USO DO TECLADO (KEYBOARD) NUM DETERMINADO COMPUTADOR

Caracteres maiúsculos e minúsculos.

Caracteres superiores e inferiores.

Para se introduzir o carácter inferior da tecla basta pressioná-la.

Para se introduzir o carácter superior da mesma tecla pressiona-se juntamente com a tecla o comando SHIFT.

CARÁCTER MAIÚSCULO

O computador normalmente começa com os tipos maiúsculos.

Quando se deseja caracteres maiúsculos e minúsculos deve-se pressionar SHIFTO.

Neste estado os símbolos serão minúsculos e para os símbolos maiúsculos usaremos novamente SHIFT.

E finalmente para se voltar ao normal ou inicial, pressiona-se SHIFTO novamente.

Ex.: NEW ENTER
READY
30 PRINT "NÃO VA"
40 PRINT "SHIFTO o que tem aí?!" SHIFTO
50 PRINT "SHIFT NADA"
60 PRINT "SHIFTO TUDO" SHIFTO
70 PRINT "TUDO BEM"

O resultado será
NÃO VA
O que tem aí?
NADA
tudo
TUDO BEM

INTRODUÇÃO PARA O PROCESSAMENTO DO PROGRAMA PELO COMPUTADOR

Uma vez corrigido e gravado o programa em linguagem de máquina enviamos esse programa para a memória principal através do comando LOAD.

Uma vez o programa carregado em linguagem de máquina na memória principal, executamos com o comando RUN que indica uma corrida de máquina para o processamento do programa e temos assim a solução do programa ou problema.

Durante essa operação teremos a indicação dos erros cometidos no programa ou na sua introdução na máquina.

Ex.:

Aparece no vídeo a mensagem ERROR com o número de linha onde se processou o erro.

Assim:

```
20 INPUT A, B, C
30 LETX = A + B/C
40 END
   RUN
```

Aparece no vídeo:

20 INPUT A, B, C ERROR, pois a variável C não pode ser nula ou zero.

GRAVAÇÃO DE UM PROGRAMA EM DISCO OU FITA

A gravação de um programa que está na memória pode ser feita pelo comando SAVE ou CSAVE. "nome do programa ENTER".

O programa a ser gravado deverá ser definido com um nome identificado por uma letra qualquer.

CARREGAR UM PROGRAMA EM LINGUAGEM DE MÁQUINA

Usa-se o comando LOAD ou CLOAD e a seguir ENTER, o computador ligará o gravador de fita ou disco e carregará o primeiro programa que encontrar.

Quando usarmos o nome do programa que queremos, então o computador irá procurá-lo e indicará quando encontrá-lo e o gravará.

OPERAÇÕES FUNDAMENTAIS

A **introdução** do programa e linhas imediatas, comando direto.

Execução do programa mediante o comando.

Edição — programa de correção dos dados.

Sistema — usado para carregar programa armazenado na memória externa.

Obs.: Você deve conhecer perfeitamente as teclas do seu terminal e também as teclas que o seu sistema usa.

Ex.: As teclas de **Retorno de Linha** são CR, RETURN ou ENTER.

INTRODUÇÃO PARA OPERAÇÃO DE MÁQUINA

Comando NEW — comando de controle — limpa a memória. — Apertar a tecla RETURN, ENTER ou CR.

READY aparece no terminal .

Com o uso da tecla RETURN, ENTER ou CR colocamos o comando na memória interna .

Se o primeiro carácter da linha for um número, será interpretado como uma linha de programa e irá para a memória de programas.

Caso contrário será executada.

Obs.: Neste caso se o número deste comando for usado por outro comando, apagará o comando anterior.

COMANDO INPUT

Sintaxe

N.° comando mensagem

20 INPUT A, B, C

a lista de variáveis devem ser separadas por vírgulas.

Ex.: A = 3 B = 5 C = 10.

20 INPUT A, B, C — este comando imprime o sinal ?. Solicitando os valores de A, B, C, que serão fornecidos através do teclado.

Este comando envia os dados para a memória principal, os quais serão processados segundo o comando seguinte.

INPUT

Sintaxe

30 INPUT (comentário entre aspas).

Permite a entrada de dados durante a execução de um programa.

Ex.:

```
20 INPUT "Qual é o número"; y
30 PRINT "O cubo é" y * 3 ou y [ 3
40 END
   RUN
```

Qual é o número? 3 (este número é introduzido pelo operador).

Resolução — O cubo é 27.

COMANDO PRINT

Sintaxe

20 PRINT lista de dados a serem impressos no vídeo ou impressora.

Esta instrução permite a impressão de um conjunto de itens na tela ou impressora.

Este comando sem comentário ou itens, é usado para pular linha.

```
Ex.: 10 INPUT A, B, C
     30 PRINT A, B, C
     40 PRINT
     50 END
```

COMANDOS INPUT, PRINT

Sintaxe

20 INPUT (variável ou comentário)

30 PRINT (variável ou comentário)

Estes comandos permitem ler e escrever os dados

A = 5 B = 3 C = 2

Ex.: 20 INPUT A, B, C

30 PRINT A, B, C

40 END

RUN

Resultado:

? comando INPUT solicitando os valores A = 5 B = 3 C = 2.

5, 3, 2 valores digitados

5 3 3 valores impressos.

Sintaxe do comando PRINT A, B, C (variáveis)

A vírgula, na instrução PRINT faz com que cada dado seja impresso na zona seguinte de impressão.

Ex.: Seja o comando

20 PRINT A, B, C, A=10 B=5 C=20.

RUN

10 5 20

O ponto e vírgula faz com que cada dado seja impresso um após outro, sem espaços.

Sintaxe PRINT A; B; C

Ex.: 20 PRINT A; B; C

COMANDO RUN

```
10 5 20
20 5 20
```

Comando PRINT com variáveis não numéricas ou strings.

Sintaxe:

```
10 PRINT (variável entre aspas)
   A$ variável não numérica
```

Ex.:
```
A$ "ENDEREÇO".
B$ "RUA 10 N.° 5"
20 PRINT "ENDEREÇO"
30 INPUT A$
40 PRINT "RUA 10 Nº. 5"
   END
   RUN
```

O uso do ponto e vírgula faz com que a mensagem seja gravada uma seguida da outra.

Ex.:
```
20 PRINT "JOÃO"; "PEDRO"
30 END
   JOÃOPEDRO
```

O uso do ponto e vírgula colocado no final da 1.ª linha seguido da 2.ª linha com outra mensagem e outro comando faz com que a 2.ª mensagem seja impressa na mesma linha da primeira, não imprimindo a primeira mensagem.

```
Ex.: 20 PRINT "JOÃO";
     30 PRINT "PEDRO"
     40 END
        RUN
        JOÃO PEDRO
```

Sem o ponto e vírgula teríamos 2 linhas de impressão.

COMANDO PRINT

Teremos as mesmas regras para os comandos: usando (?),(;) e < PRINT.

A vírgula e ponto e vírgula no comando PRINT controla a impressão da mensagem na tecla ou impressora.

Sendo uma linha de impressão dividida normalmente em 4 zonas, teremos os seguintes resultados da sintaxe do comando PRINT.

O uso da vírgula separando as mensagens ou variáveis faz com que estas sejam impressas na zona seguinte.

Ex.: 20 PRINT "JOÃO", "PEDRO"
30 END
JOÃO PEDRO

Comando PRINT sem n. de comentário e com n

Ex.: 5 PRINT "linha de programa"
PRINT "linha imediata"
? substitui a instrução PRINT.
PRINT usado sem número ou comentário é usado para espaçar linhas.

O COMANDO SHIFT

<SHIFT> copia o conteúdo da tela na impressora.
<SHIFT> @ comando de pausa ou parada, para continuar o programa pressiona qualquer tecla menos <BREAK> e <RESET> CLEAR, RESET

COMANDO LET

Sintaxe:

20 LET variável = expressão aritmética.

Este comando realiza as operações aritméticas de uma expressão.

Ex.: P = 5 + 2 * 7/2

```
NEW
READY
20 INPUT 5, 2, 7, 2
30 LET P = 5 + 2 * 7/2 (digitado)
40 END
```

RUN realiza as operações no computador.

No computador:

20 ? (solicita o valor 5, 2, 7, 2
30 realiza as operações indicadas
40 termina o programa
RUN processa os dados.

CÁLCULO DE UMA EXPRESSÃO ARITMÉTICA

Seja:

$$X = A \cdot B$$

$$Y = A + B \cdot 3 - 5 \cdot A$$

$$Z = \frac{B + C}{C} \Big/ B * E$$

Temos o seguinte comando:

```
NEW
30 INPUT A, B, C, E
40 LET X = A * B
50 LET Y = A + B * 3 - 5 * A
60 LET Z = ((B+C)/C)/B * E
70 X, Y, Z
80 END
   RUN
```

Execução de um programa pela digitação do comando RUN.

OPERAÇÕES ALTERNATIVAS

Após o comando END digitar RUN.

Ex.: Executar o programa a partir do 1.° comando até encontrar o comando STOP ou END.

Ex.: O programa que se encontra na memória interna é: A=50 B=20 C=2

```
100 INPUT A, B
110 LET P=(A-B) / (2 * C)
120 PRINT P
130 END
  RUN
```

teremos:

? o computador solicita os valores A, B, C.
50, 20, 2 — valores digitados
7, 5 — valor calculado por LET
7, 5 — valor impresso por PRINT

O comando RUN usado para iniciar a operação a partir de uma determinada linha do programa.

Sintaxe:

```
20 RUN 100
```

A execução do programa será feita a partir da instrução 100.

INSTRUÇÃO END

Sintaxe: END

Finalizar a execução de um programa e voltar ao nível de comando.

Ex.:
```
100 INPUT A, B
110 PRINT A, B
120 END
```

Após este comando RUN é o próximo para a execução do programa pelo processador central.

COMANDO STOP

Sintaxe: STOP

Interrompe a execução do programa em execução e volta ao nível de comando direto, mostrando a mensagem "BREAK n.º" seguida do número da linha interrompida.

```
100 INPUT A, B, C.
110 LET D = A+B+C.
120 STOP
130 PRINT A, B, C, D
140 END
```

Só continuará após CONT.

COMANDO LIST

Sintaxe: LIST número linha — número linha.

Permite fazer uma lista total ou parcial do programa contido na memória para vídeo.

LIST lista o programa inteiro
LIST 100 lista somente a linha 100
LIST 100— lista o programa a partir de 100
LIST 50-80 lista o trecho do programa entre a linha 50 a 80, portanto, 50, 60, 70 e 80.
LIST — 100 lista até a linha 100
LIST . lista a linha atual.

COMANDO NEW

Sintaxe NEW

Apaga o programa da memória principal preparando-a para um novo programa.

COMANDO SAVE

Sintaxe SAVE

Grava o programa da memória principal em uma memória externa, fita, disco, etc.

COMANDO APPEND

Idêntico ao LOAD porém permite que o programa seja acrescentado a outro já existente na memória principal.

COMANDO AUTO

Este comando permite a função automática de numeração de linhas de programa.

Sintaxe:

AUTO pressiona-se ENTER

Teremos a numeração automática a partir de 10; 10 20 30 etc.

O uso do número da linha inicial mais o incremento indicará a função.

Sintaxe:

Auto 5,5 teremos 5, 10, 15

COMANDO DELETE

Este comando tem como função apagar as linhas de um programa na memória que é especificada pelo número de linha.

Ex.:

DELE 20, 30 — apaga as linhas.

DELETE 100 — apaga as linhas a partir de 100, até o final do programa.

DELETE — 100 — apaga as linhas desde o início até a linha 100.

COMANDO BREAK

Este comando interrompe a execução de qualquer operação do computador, quer seja programa, gravação, impressão, etc.

Ele devolve o controle da máquina para o usuário.

EDIÇÃO

COMANDO EDIT

O computador opera em modo de edição

Este comando é usado para corrigir ou alterar o texto de um programa que está sendo executado.

Sintaxe:

EDIT — número da linha a ser modificada.

O computador entra a nível de edição para a linha definida pelo número.

SUBCOMANDOS EDIT

1 — para movimentar o curso no Texto .
2 — para inserir dados no Texto.
3 — para cancelar o Texto.
4 — para substituir o Texto.
5 — para pesquisar o Texto.
6 — para encerrar e reiniciar a Edição.

MOVIMENTAÇÃO DO CURSOR

Sintaxe: — Número Inteiro e Barra de espaço

Com o uso da Barra move-se o cursor n espaços para a direita, movendo-se carácter a carácter.

Para Retroceder o cursor n espaços.

Sintaxe:

Número Inteiro e <— (BACKSPACE)

Move o cursor para a esquerda n espaços sem apagar os caracteres.

COMANDO EDIT

INSERÇÃO DE TEXTO

Sintaxe:

SHIFT *

Neste estado o texto a ser incluído será feito a partir do cursor .

Sintaxe:

X

Move o cursor para o fim da linha e entra em subnível de inserção.

COMANDO EDIT

CANCELAMENTO DE TEXTO

Sintaxe: Número Inteiro n.

Com o uso da barra de espaço.

Cancela n caracteres a partir do cursor (carácter a carácter).

Os caracteres cancelados aparecem no vídeo entre pontos de exclamação (!)

Sintaxe: H

Cancela **todos os caracteres** a partir do cursor ao fim da linha e entra no subnível de inserção, para introdução do novo texto.

Sintaxe: Número inteiro K Carácter/Dígito

Cancela todos os caracteres entre a posição do cursor até a enésima ocorrência do Carácter/Dígito.

COMANDO EDIT

PESQUISA DE TEXTO

Sintaxe: n.º inteiro S Carácter/Dígito.

Procura a enésima ocorrência do carácter na linha a partir da posição do cursor.

COMANDO EDIT

SUBSTITUIÇÃO DE TEXTO

Sintaxe: Número inteiro C texto.

Substitui n caracteres a partir da posição do cursor pelo texto, se não tiver o número n então somente um carácter será substituído.

COMANDO EDIT

ENCERRAMENTO E REINÍCIO DA EDIÇÃO

ENTER: Mostra o restante de linha e volta ao nível de comando direto.

E: O mesmo que ENTER porém não mostra o restante da linha.

Q: Volta ao nível de comando sem nenhuma modificação.

L: Mostra toda linha e posiciona o cursor no início.

A: Cancela todas as modificações e posiciona o cursor no início da linha.

COMANDO DE INSTRUÇÕES

END
FOR / NEXT
GOSUB / RETURN
GOTO
IF / THEN / ELSE
ON GOSUB
ON GOTO

COMANDO FOR — NEXT

Sintaxe:

FOR variável = valor inicial para o valor final com incremento da variável (STEP).

NEXT valor da variável até atingir a condição exigida.

Este comando permite a repetição de um trecho (laço) do programa até se chegar a condição definida pela variável.

Ex.: Seja determinar os valores da variável definida pela expressão A = 5 a 20 incremento 3

```
100 FOR A = 5 TO 20 STEP 3
                (5+3), (8+3), (11+3)
                (14+3) (17+3)
110 NEXT A
```

Neste exemplo o trecho do programa é repetido 5 vezes após o término das repetições o programa continuará com o comando

110 NEXT A

INSTRUÇÕES GOSUB/RETURN

Sintaxe: GOSUB num. linha

RETURN

Este comando de entrada e saída, de um sub-rotina indicada pelo número de comando e a primeira linha da sub-rotina e a sua volta pelo RETURN ao programa, à próxima instrução após a instrução GOSUB.

Ex.:

```
10 GOSUB 50
20 PRINT "Programa Fonte"
30 STOP
100 PRINT "Sub-rotina SENDO":
110 PRINT "Executada"
120 RETURN
```

RUN

Sub-rotina sendo executada

Programa Fonte

BREAK IN 30

INSTRUÇÕES

IF — THEN — ELSE

Sintaxe: IF (expressão) — THEN (instrução — ELSE (instrução).

A decisão a ser tomada se baseia no resultado lógico das expressões.

Assim, se for verdadeiro, executa as instruões após THEN caso contrário ignora, e executará após ELSE.

```
100IFX = Y THEN PRINT
"X=Y" ELSE IFX<Y THEN
PRINT "X<Y" IFX>Y THEN
PRINT X < > Y
```

Obs.: O uso dos sinais de relações tanto para expressão numérica como Strings.

INSTRUÇÃO GOTO

Sintaxe: 100 GOTO número de linha.

Este comando desvia o programa em execução para o número de linha indicado na sintaxe ou condição definida.

Ex.: 20 — — — — —
30 — — — — —
100 I F X + Y = 56 THEN PRINT
"SIM
110 I F X + Y > 56 THEN
GOTO 30

INSTRUÇÕES IF — THEN — ELSE

Com operadores lógicos. AND; OR e NOT

100 I F X > Y END Y > 5 THEN PRINT "X > 5"
ELSE IF X>Y OR Y>X PRINT "X>5"

INSTRUÇÕES ON GOTO

Sintaxe: ON **expreşsão** GOTO **num. linha.**

Desviar o programa para uma linha especificada,

a partir do valor da expressão e não do número do comando.

```
Ex.: 100 INPUT COD
     120 ON COD GOTO 000; 22, 90
```

Se COD = 1 o programa irá para a linha 000
Se COD = 2 o programa irá para a linha 22
Se COD = 3 o programa irá para a linha 90

SINTAXE DE COMENTÁRIOS ENTRE ASPAS

```
    REM — comentário; definição.
    A = 3   B = 8   C = 10,3
10 REM
20 PRINT "A IGUALA", 3
30 PRINT "B IGUALA", 8
40 PRINT "C IGUALA", 10,3
50 END
   RUN
```

Temos na máquina:

```
RUN
3, 8, 10,3
A IGUAL A 3
B IGUAL A 8
C IGUAL A 10.3
  READY
```

INSTRUÇÕES CLEAR

Sintaxe: CLEAR expressão.

Permite colocar todas as variáveis numéricas em zero e todas as alfanuméricas em nulo.

(Carácter nulo de controle usado para preencher espaços em zero não numérico são inseridos ou removidos de uma linha sem alterar o significado, mas o controle pode ser alterado).

Ex.: CLEAR

CLEAR 10

INSTRUÇÃO ON - - - - GOSUB

Sintaxe: 50 ON expressão GOSUB número de linha.

Esta instruão desvia o programa para uma sub-rotina em relação a uma condição do valor de uma expressão.

ON permite a escolha de um dos vários comandos.

GOTO ou GOSUB

Ex.: 10 INPUT X
20 ON X GOSUB 100, 200

Se X = I O programa irá para sub-rotina 100
X = 2 o programa irá para sub-rotina 200

INSTRUÇÃO INKEY$

Sintaxe: INKEY$

Permite verificar se alguma tecla foi pressionada.

Ex.: — — — — —
100 B$ = INKEY$

O carácter da tecla que foi pressionada será armazenada na variável B$

COMANDO DIM

Variáveis dimensionadas.

Vetores e Matrizes.

Vetores — são definidos como matrizes de uma só dimensão; isto é, o vetor é representado por um número.

Assim a matriz com 5 elementos, 6; 10 etc.

A representação simbólica é feita por uma letra e o número de elementos entre parênteses.

Ex.: A (5) B (6) D (8) etc.

PARTE
II

NÍVEL I e II

OBSERVAÇÃO IMPORTANTE

A repetição dos módulos de programação se torna uma necessidade no aprendizado devido às combinações de comandos e instruções das operações cujos significados vão mudando segundo os níveis de aplicação.

Assim uma operação de união pertence ao nível fundamental ao passo que uma adição pertence ao nível I, uma multiplicação ao nível II etc.

Por esta razão e para melhor compreensão devemos repetir as combinações de comandos usados em um nível ou módulo para outra combinação que usa parte ou total desse comando em outro nível.

SISTEMA DE NUMERAÇÃO

Num circuito elétrico, passa ou não passa corrente elétrica. A chave está ou não ligada, ou seja, não existe meio termo. O computador, sendo uma máquina elétrica, tem que se basear seu funcionamento nisso. Por isso, todos os cálculos e memorizações, são baseados em "continhas" simples, só com "O" e "1".

CONCEITO DE VALOR DE POSIÇÃO

Para explicar o conceito de "valor de posição" vamos usar os números decimais.

Consideremos, por exemplo, o número:

	6	1	3	5
Posição ———	3	2	1	0

Este número pode ser assim decomposto:

$5 \text{ ——— } 5 \times 10^0 = 5$
$3 \text{ ——— } 3 \times 10^1 = 30$
$1 \text{ ——— } 1 \times 10^2 = 100$
$6 \text{ ——— } 6 \times 10^3 = 6000$

6135

SISTEMA DE NUMERAÇÃO BINÁRIA

O sistema binário usa somente dois símbolos: O e 1, ou seja, sua base é o número 2. Estes símbolos ou dígitos são denominados "BIT" (**B**inary dig**IT**, ou seja, número binário).

Assim como no sistema decimal ,o valor de cada símbolo, é função da sua posição, isto é, baseia-se na progressão da direita para a esquerda, das potências de 2 (2^n,. . . , 2^3, 2^2, 2^1, 2^0), enquanto que o sistema decimal, é baseado na progressão de potências de 10 (10^n,. . . , 10^3, 10^2, 10^1, 10^0).

Exemplos:

		1	1	0	1
Posição ———	3	2	1	0

1 ———	$1 \times 2^0 =$	1
0 ———	$0 \times 2^1 =$	0
1 ———	$1 \times 2^2 =$	4
1 ———	$1 \times 2^3 =$	8
		13

CONVERSÃO DE UM NÚMERO DECIMAL PARA BINÁRIO

Até agora temos visto, como passar de um número binário para um número decimal mas, podemos fazer também, a operação inversa, ou seja, passar um número decimal para binário. Precisamos de um método que nos proporcione a divisão de um número em parcelas tais que a cada parcela corresponda a uma potência de 2.

1) Divide-se o número por 2, chegando a um quociente inteiro e guarda-se o resto.

2) Separa-se em seguida ,o quociente obtido anteriormente e divide-se novamente por 2, guardando-se o resto. Repete-se esta operação sucessivamente até que o quociente dê O.

3) Separam-se então os restos na ordem inversa a que foram aparecendo ,isto é, do último para o primeiro, colocando-se um a um da esquerda para a direita. O número formado será o binário desejado.

Exemplo:

Se desejarmos achar o binário corerspondente ao decimal 32, teremos:

```
32 | 2
12   16 | 2
 0    0    8 | 2
           0   4 | 2
               0   2 | 2
                   0   1 | 2
                       1   0
```

$32_{10} = 100000_2$

SISTEMA DE NUMERAÇÃO HEXADECIMAL

O Sistema Hexadecimal utiliza 16 símbolos, que são, em ordem crescente: O, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. (convenção Internacional), portanto, tem por base o número **16.**

Esse sistema de numeração é útil quando se quer representar cifras, que no sistema binário representará uma extensa série de "O" e "1". **Cada cifra hexadecimal** permite representar **quatro** cifras binárias.

Façamos um confronto entre estes e aqueles do sistema decimal:

Comparação entre o sistema decimal e o sistema hexadecimal.

Sistema Decimal	Sistema Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Podemos observar que aos valores decimais 10, 11, 12, 13, 14, e 15, corresponde respectivamente os valores A, B, C, D, E, e F.

Observemos a seguir a correspondência entre números decimais ,binários e hexadecimais.

TABELA 1

Sistema Decimal	Sistema Binário	Sistema Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Como podemos ver, qualquer combinação de 4 bits pode ser representada por uma única cifra hexadecimal.

CONVERSÃO DE UM NÚMERO BINÁRIO PARA HEXADECIMAL

Exemplificando, para o número binário:

11/1010/0110/0101/0010

Separando-se em grupos de quatro bits **a partir da direita** e incluindo dois zeros a esquerda para completar o último grupo, teríamos:

0011/1010/0110/0101/0010

Baseando-se na tabela 1, substituimos cada grupo binário pelo seu correspondente símbolo hexadecimal.

Obtemos o número hexadecimal: 3A652

Como exercício, converta os números binários à seguir para hexadecimais:

1101000100111011 = ...

01010011101101 = ...

10001010111101110 = ...

CONVERSÃO DE UM NÚMERO HEXADECIMAL PARA BINÁRIO

Dado o número hexadecimal C74BA5, é suficiente, usando a tabela 1, converter cada símbolo hexadecimal na correspondente configuração binária, assim:

C 7 4 B A 5

1100 / 0111 / 0100 / 1011 / 1010 / 0101

CONVERSÃO DE UM NÚMERO DECIMAL PARA HEXADECIMAL

O método usado é análogo ao método para transformar um decimal em binário, com a diferença de que as divisões são feitas por 16. Ao utilizarmos os restos da divisão, nesta transformação, devemos ter o cuidado de usar o algarismo hexadecimal correspondente a esse resto, pois fazemos as divisões na forma decimal e portanto o resto obtido será decimal, devendo ser convertido para o número hexadecimal correspondente.

Exemplo:

```
1517 | 16
 077   94 | 16
  13   14   5 | 16
  ''   ''   5   0
  D    E    ''
            5
```

Portanto $(1517)_{10} = (5ED)_{16}$

Tente com o número 2523.

CONVERSÃO DE UM NÚMERO HEXADECIMAL PARA DECIMAL

Vamos ao exemplo:

Converteremos F3A para o número decimal correspondente:

F	3	A
''	''	''
15	3	10

$$F \text{ ——— } 10.16^0 = 10$$
$$3 \text{ ——— } 3.16^1 = 48$$
$$A \text{ ——— } 15.16^2 = 3840$$
$$\text{——}$$
$$3898$$

UNIDADE DE INFORMAÇÃO EM LINGUAGEM DE MÁQUINA
BIT e BYTE

BIT — dígito binário.

É a unidade elementar de uma palavra de máquina e que representa dois estados, duas condições ou dois valores numéricos: 0 ou 1 .

É o elemento usado para representar uma informação num sistema lógico de programação digital.

Segundo o sistema usado ele pode representar a condição de falso ou verdadeiro, de ligado ou desligado, dos valores 0 ou 1 na álgebra booleana, etc.

BYTE — termo binário **(Bynary Term).**

O BYTE é uma seqüência de bits.

A representação de um carcter é feita por 6 ou 8 bits consecutivos e recebe o nome de byte.

Assim temos um byte por caracter.

Um byte é considerado pelo computador como um caracter ou uma palavra de máquina, sendo a unidade mínima de informação que pode ser armazenada num espaço da memória principal.

Como o computador digital só trabalha com números, foi feita a representação dos caracteres pela correspondência dos caracteres por números binários.

Assim o código de caracteres estabeleceu o padrão universal — ASCII — que usa um byte para a representação de cada caracter.

Sendo o byte formado por 8 bits consecutivos, temos na notação binária 256 (2^8) valores que formam o número de caracteres.

Representação parcial da Tabela ASCII

TABELA ASC

Código			Representação binária na Linguagem de máquina
Dec.	**Hex.**	**Caracter**	
33	21	!	00100001
34	22	"	00100010
35	23	#	00100011
36	24	$	00100100
37	25	%	00100101
. . . . .			
48	30	0	00000000
. . . . .			
57	39	9	00111001
. . . . .			
65	41	A	01000001
. . . . .			
90	5A	Z	01011010
. . . . .			
97	61	a	01100001
. . . . .			
122	7A	z	01111010

Estes valores se encontram na memória do computador. Deste modo podemos concluir que um computador digital, trabalhando com dados numéricos no sistema binário, pode-se perfeitamente representar todos caracteres descritos por números binários e assim, usando-se a lógica e a álgebra booleana, podemos realizar todas as operações aritméticas e lógicas através dos circuitos integrados do microcomputador.

Foi então estabelecido o código de caracteres sob um padrão universal cujo nome é:

A S C I I

American National Standard Code for Information Interchange, cuja tradução é:

Código padrão de intercâmbio de informação nacional americano.

(Obs.: não precisa se preocupar em guardar a tabela de conversão, pois pode ser checada a qualquer hora no próprio computador.

FORMAÇÃO DE PALAVRAS

Podemos formar palavras assim:

Exemplo: CURSO DE BASIC

C	U	R	S	O
01000011	01010101	01010010	01010011	01001111
D	**E**			
01000100	01000101			
B	**A**	**S**	**I**	**C**
01010011	01001001	01000011	01000010	01000001

2) Como exercício, tente escrever em código ASCII, as seguintes palavras: **HOLLERITH,** 80, colunas, altura, ZONA.

Obs.: O sistema de codificação ASCII usa o sistema binário, mas outros sistemas de codificação usam o sistema hexadecimal.

CAMPOS, REGISTROS E ARQUIVOS

Campo

É um conjunto de bytes necessário para armazenar as informações.

Registro

É um conjunto de campos cujas informações estão relacionadas a um único elemento, por exemplo: nome, endereço, data do nascimento, se referem à uma pessoa.

Arquivo

É um conjunto de registros onde todos possuem uma característica comum , por exemplo: arquivo de clientes — a característica comum é o fato de cada um deles ser cliente de uma determinada empresa.

Resumindo-se, temos:

BIT O ou 1

BYTE 0 1 0 0 0 0 0 1 A (caracter)

CAMPO

A N A (nome

1 BYTE

REGISTRO

| A N A | 25/03/58 | (nome e data de
CAMPO 1 nascimento)

3) Quantos bits tem o byte "A"? 8

4) Quantos bytes tem o campo nome e quantos bits isso representa?

5) O registro nome e data de nascimento tem ..... campos e ..... bytes.

Arquivo |ANA b 25/03/58 | JOÃO b 03/05/60
| IVONE b 13/04/62 |
| REGISTRO 1 |
| REGISTRO 2 |
| REGISTRO 3 |

6) Quantos registros tem esse arquivo? Quantos campos tem cada um desses registros?

Obs.: b : significa espaço em branco, e também ocupa 1 byte.

BIT
BYTE
CAMPO
REGISTRO
ARQUIVO

LINGUAGEM DE MÁQUINA

A linguagem de um computador digital é feita através da correspondência entre os símbolos e os números em notação binária, cujos dígitos são 0 e 1.

Temos assim que todos os dados de um programa em linguagem de montagem simbólica passam a ser gravados em linguagem numérica binária ou de máquina para ser processado e resolvido pelo computador.

Concluimos então que as palavras são entendidas pela máquina e que as mesmas usam, um byte para cada caracter, byte esse que é a unidade mínima de informação.

Para melhor compreensão desta linguagem vamos citar diversos exemplos com exercícios.

EXERCÍCIOS

1) a = 01100001
   = 01011100
   * = 00101010
   & = 00100110
   2 = 00110010
   W = 01010111

2) **HOLLERITH**

ALTURA:

a	l	t
01100001	01101100	01110100
u	r	a
01110101	01110010	01100001

ZONA:

Z	O	N	A
01011010	01001111	01001110	01000001

3) 8 bits

4) 3 bytes, 24 bits

5) 2 campos, 11 bytes

6) 3 regsitros, 2 campos

REGRAS DA LINGUAGEM DE MÁQUINA

Essas regras se dividem em: **Gramaticais** e **Sintáticas** e são necessárias para a perfeita compreensão do significado dos grupos de símbolos (palavras, frases, programas).

Por exemplo: As palavras da frase: "Estou dados eu introdução de estudando ao processamento, linguagem e, estudarei uma futuramente", a frase está errada pois a frase não respeita as regras sintáticas da posição de cada elemento, embora as palavras estejam gramaticalmente correta.

O correto seria: "Eu estou estudando introdução ao processamento de dados e futuramente estudarei uma linguagem".

Na linguagem do computador ocorre o mesmo problema. Uma "frase' da linguagem do computador consiste de um certo número de instruções ordenadas sequencialmente, isto é, **uma depois da outra.**

Exemplo:
```
10 A = 100
20 B = 200
30 C = A*B
40 PRINT"O RESULTADO DA MULTIPLICAÇÃO A*B = "; C
```

A seqüência das instruções deve respeitar as regras sintáticas da linguagem. Tais regras são de dois tipos: algumas estabelecidas pela própria linguagem e que por isso, variam de acordo com cada linguagem. Outras dizem respeito ao modo mais "lógico" de se escrever um programa, ou seja, organizam a seqüência de instruções da melhor maneira possível.

OTIMIZAÇÃO DE UM PROGRAMA

O cuprimento dessas regras nos levam à "otimização" do programa em relação aos recursos disponíveis. Diz-se que um programa otimiza a utilização dos recursos de hardware (características do computador) e do software (características da linguagem) quando resolve um problema, utilizando o **menor número possível de instruções.**

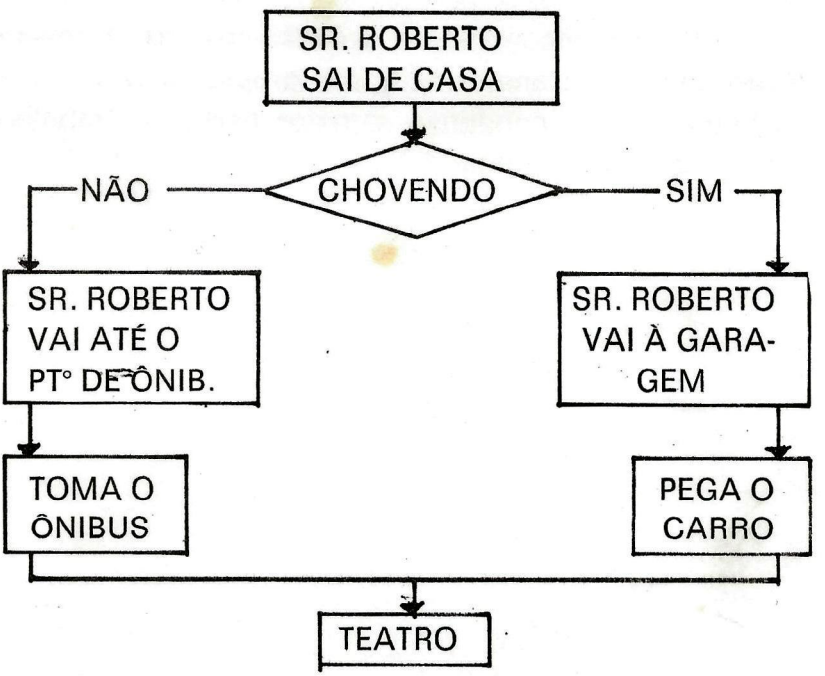
DIAGRAMAS E FLUXOGRAMAS
CONCEITO E USO DO FLOW-CHARTING

O **flow-chart**, ou **diagrama de fluxo**, ou ainda, **fluxograma**, é uma técnica que permite que se faça uma

análise dos problemas e uma representação gráfica da seqüência de operações necessárias para resolvê-los. É muito mais fácil resolver os problemas quando esquematizados graficamente do que tentar resolvê-los somente com o raciocínio, que dependendo da extensão do problema se torna impossível, ou muito trabalhoso.

Uma vez encontrada a solução gráfica, torna-se muito simmles transformá-la em programa, visto que no flow-chart se condensa a maior parte do trabalho lógico de um problema.

Ex.: 1) Sr. Roberto resolve ir ao teatro, para tanto sai da casa. Se estiver chovendo vai à garagem e pega o carro. Caso contrário, vai até o ponto de ônibus e toma o ônibus.

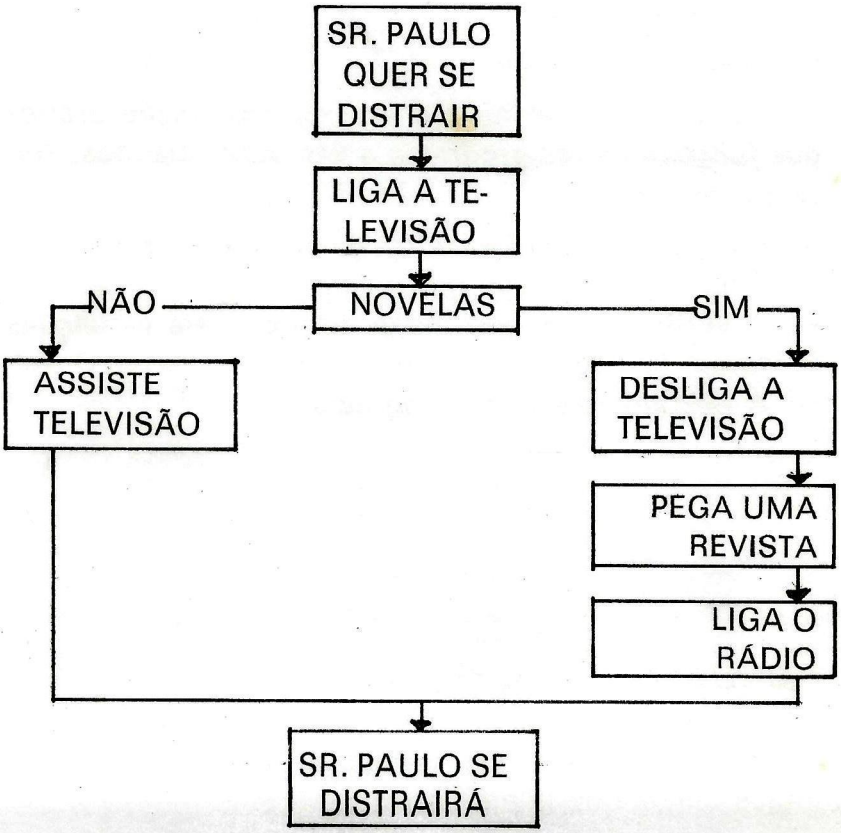


Cada frase é escrita em um retângulo. O elemento discriminante, chove ou não, é representado por um losango. A seqüência das várias fases é representada pelas linhas que unem os retângulos.

Além do retângulo e do losango, usaremos muitos outros símbolos que estão especificados no xerox — **SIMBOLOGIA DE SISTEMAS** (fluxograma).

Ex.: 2) Sr. Paulo resolve se distrair, para isso, usa a televisão. Se estiver passando novelas, desliga a televisão, pega uma revista e liga o rádio, caso contrário, assiste televisão.
De qualquer forma, Sr. Paulo se distrairá.

Vamos fazer o diagrama de blocos dessa situação.



Como exercício ,tente fazer o diagrama de blocos para as duas situações que se seguem:

1) Fazer uma chamada telefônica através de um telefone público (orelhão). Suponha que você já tenha 1 ficha na mão e já está diante do telefone. OBS.: Suponha também que 1 ficha é suficiente para a sua conversação.

2) Descrever todos os passos que você faz para chegar à Prológica, desde o momento que sai de casa, até chegar aqui.

Resumindo-se, temos:

O diagrama de blocos é a **representação gráfica das funções de um programa e das suas relações.** Portanto, temos:

— Uma representação gráfica da **solução** do problema;
— Uma representação gráfica da **lógica** do problema;
— A verificaão de que todas as possíveis condições foram levadas em consideração;
— A documentação do programa.

DADOS NUMÉRICOS E STRINGS, CONSTANTES E VARIÁVEIS

Uma constante string é a seqüência de caracteres alfanuméricos que representam os dados de um programa.

Esses caracteres podem ser quaisquer dos símbolos que o computador pode gerar.

VARIÁVEL

A variável é o elemento ou dado que pode ser modificado sem alterar a estrutura do programa que vai ser processado.

REPRESENTAÇÃO DE UMA VARIÁVEL

A variável é representada por um nome.

O nome é um conjunto de caracteres alfanuméricos não superior a 30 caracteres e o mínimo de 6 usado para representar uma variável, uma matriz, uma função ,etc.

Na linguagem BASIC a variável é representada por um nome de 1 a 2 caracteres sendo que o primeiro deve ser obrigatoriamente uma letra do alfabeto.

Ex.: A B C

AX A1 B2 C1

CONSTANTE NUMÉRICA

Valor ou dado fixo de um programa. Invariável.

Temos três tipos de constantes, a saber:

Constantes numéricas, strings, e falso/verdadeiro.

O BASIC aceita os valores constantes numéricos entre os números em notação comum de

$\pm 1{,}0 \times 10^{-99}$ a $\pm 1{,}0 \times 10^{99}$

na forma exponencial.

E na forma decimal

de 1,0 a 99999.

Quando se tenta introduzir dados com valores superiores aos da capacidade de armazenamento do computador este acusa a condição de erro que é definida com overflow (OV) quando esta quantidade for menor que não-zero, quantidade underflow.

Quando uma variável for representada por mais de 2 caracteres somente os 2 primeiro serão considerados como nome de variável na linguagem BASIC.

TIPOS DE VARIÁVEIS

Chama-se variável simples quando ela assume um único valor do conjunto de valores ou dados que ela pode assumir.

Ex.: A = "SIM"
B = "BOM"
A2 = "VERDADE"
C = 5

VARIÁVEL DIMENSIONADA

Chama-se variável dimensionada quando ela assume um conjunto de valores definidos.

MATRIZ

Quando esse conjunto de valores definidos pela variável dimensionada for de uma dimensão ela recebe o nome de Vetor e quando for de duas ou mais dimensões ela recebe o nome de Matriz.

O vetor ou a matriz é representada por uma só letra e o número de seus elementos colocados entre parênteses.

Ex.: A(2) B(5)
C(2, 3) B(, 5, 7)

COMO O BASIC TRATA AS VARIÁVEIS

Capacidade de armazenamento de dados.

O BASIC pode armazenar até 255 dimensões.

Os dados ou valores de uma variável são armazenados num limite de espaço definido pelo interpretador BASIC na memória RAM.

O Comando usado para indicar a utilização de toda a área reservada da memória RAM para armazenar os dados do programa é ENTER (ou o comando indicado no manual de seu computador).

A numeração das linhas de programa em BASIC vão de 0001 a 9.999.

EXPRESSÃO STRING OU ALFANUMÉRICA

É a expressão que define um termo relativo ao dado ou informação de um programa, composta por caracteres alfabéticos, numéricos ou especiais ou uma combinação desses caracteres.

VARIÁVEIS STRINGS OU EXPRESSÕES ALFANUMÉRICAS

Todos os nomes de variáveis que representam dados alfanuméricos são chamados strings e recebe o nome de variável string.

Estas variáveis não são processadas pelo computador, isto é, o processador central ignora os dados strings.

O número máximo de caracteres de uma string é de 255 caracteres.

ARMAZENAMENTO DE UMA STRING NA MEMÓRIA INTERNA

A armazenagem de uma string na memória interna é feita através do uso do Código ASCII que utiliza um byte para cada caracter.

VARIÁVEIS NUMÉRICAS

Os nomes dos dados numéricos ou de expressões aritméticas recebem o nome de **variável numérica.**

As expressões numéricas ou aritméticas são formadas por números e operandos.

Assim a variável numérica é tratada ou processada pelo processador na linguagem BASIC do seguinte modo:

1.°) NÚMEROS INTEIROS

São considerados inteiros os números negativos positivos e o fracionário com até 6 casas decimais.

O mesmo acontece com os números irracionais e com a dízima e neste caso dizemos números inteiros de simples precisão.

Obs.: Quando se quer modificar este tratamento dessas variáveis em relação a precisão, então teremos outros recursos porem pertence a outro nível e são usados declarações especiais de comando.

Ex.:

A = 2.578

usa-se A% = 2.578

e teremos a gravação na memória do valor 2.

Assim 17 A = SQR (17)

temos A% = SQR(17) teremos o resultado 4 gravado na memória e não como muitos escrevem erradamente.

17 = 4 ERROR

EXPRESSÕES RELACIONAIS
OPERAÇÕES RELACIONAIS; OPERADORES

Quando se necessita da comparação entre duas expressões aritméticas são usados os operadores relacionais.

OPERADORES

Vamos repetir aqui esses operadores que são:

> maior que

< menor que

>= ou =< maior ou igual a

=< ou <= menor ou igual a

= igual

<> menor ou maior que

>< maior ou menor que

estes dois últimos são equivalentes ao nosso conhecido diferente que não é reconhecido nesta linguagem.

Estas operações são realizadas pelo processador quando usamos a instrução RUN

Ex.: a > b 2 < 5

− 2 > − 5 2 <> 5

Resumo da ação dos operadores booleanos segundo as leis da álgebra de BOOLE

AND quando todas condições são verdadeiras então o resultado é 1, caso contrário é 0;

OR quando uma condição é verdadeira a resultante será 1 e somente quando todas forem falsas a resultante será 0

NOT troca de valores 0 para 1 ou 1 para 0.

OPERADORES LÓGICOS
OR, AND, NOT

Normalmente as operações indicadas por estes operadores são feitas pelo comando

IF / THEN / ELSE

As expressões lógicas podem ser feitas entre duas expressões aritméticas.

OPERAÇÕES COM EXPRESSÕES STRINGS

(não realizadas pelo processador central)

As operações com estas expressões são tratadas pelo computador em BASIC segundo as leis de sua construção.

Em BASIC o único operador string é + (mais), que significa a união de duas ou mais strings e recebe o nome de operação de concatenação.

Nesta operação a ordem das strings é feita seguindo a formulação da operação.

Ex.: PRINT "SIM" + "NÃO" + "PEDRO"
fica SIM NAO PEDRO

OPERAÇÕES DE RELAÇÃO ENTRE STRINGS

Os operadores usados nestas aplicações são:

> o termo que segue o 1.° tem maior prioridade

< o termo que segue tem menor prioridade

= igualdade de prioridade

>< tem prioridade de níveis diferentes

<>

<= o termo que **segue** tem **menor** ou igual prioridade

>= o termo que **segue** tem **maior** ou igual prioridade.

As comparações strings são feitas no BASIC caracter a caracter.

Quando se executa uma comparação entre 2 strings e são encontrados caracteres diferentes na mesma posição das expressões, aquela de caracter com menor número de código ASCII terá maior prioridade, isto é, é impresso primeiro que o outro.

Esta propriedade tem larga aplicação na classificação alfabética.

Ex.: "B" < "C" o n.° do código ASCII é

B é 66 C é 67

então teremos a impressão B C.

Para o caso de não encontrar caracteres não-coincidentes então a string mais curta será considerada de maior prioridade.

OUTROS EXEMPLOS

Temos a ordenação das strings segundo a seqüência ASCII das expressões.

Assim teremos:

1.° termo > 2.° termo — o primeiro termo vai após o 2.°, tem menor prioridade.

"B" > "A" — O n.° de B no código é 66
O n.° de A no código é 65

1.° termo < 2.° termo

neste caso teremos:

"A" < "B" cuja prioridade você pode deduzir.

Obs.: As margens e espaços também são considerados nas strings.

Assim:

" B" < "B" — o n.° de código espaço é 32 portanto o " B" tem prioridade sobre "B".

FUNÇÕES NUMÉRICAS E STRINGS

Uma função é uma subrotina residente no interpretador BASIC cuja execução é feita pelo uso de uma palavra chave seguida de um parâmetro.

Esse parâmetro deve vir sempre entre parênteses e separados entre si por vírgula.

Ex.:

```
S Q R (27)
S Q R (A + B − 20)
```

FUNÇÃO STRING

Ex.:

```
PRINT ASC (String)
```

fornece o código ASCII do primeiro caracter da string.

REPRESENTAÇÃO DOS DADOS DE UM PROGRAMA EM BASIC

Os dados de um programa são definidos por um **nome de variável.**

Esses dados podem ser constantes ou variáveis; (recebe o nome representado por uma letra ou uma letra e um número conforme já vimos).

NUMÉRICOS OU STRINGS

Todos eles recebem um nome de variável que indicará em cada nome o conteúdo que é o dado do programa, cuja indicação é feita através do sinal de =.

Todos os dados que não forem numéricos serão indicados como strings cuja notação será entre aspas e o nome da variável será acompanhado do sinal $.

Ex.:

A$ = "ENDEREÇO, NOME".
B$ = "JOAQUIM ANTUNES".
C$ = "10 DE OUTUBRO".

OUTRAS LINGUAGEM

COBOL
FORTRAN
PASCAL
PL / I

Neste campo já exige todo conhecimento anterior e mais o superior.

As linguagens de programação já são mais avançadas.

Temos neste campo a linguagem PI / I.

Programming Language one — Linguagem de programação n.° um.

Esta linguagem é mais poderosa que as outras, oferecendo maiores recursos em comandos e operações e incluem ainda os recursos das linguagens anteriores tanto BASIC. COBOL, FORTRAN.

ÍNDICE

Impresso nas oficinas da
EDITORA PARMA LTDA.
Fones: 66-3095 - 912-0790 - 912-0802 - 912-0819
Av. Antônio Bardela, 180
Guarulhos - São Paulo - Brasil
Com filmes fornecidos pelo Editor