

ALGORITMOS

Prof. Nilton

Tipos de Dados

Uma das principais funcionalidades de um computador é a manipulação de informações. Por isso é necessário que haja formas de se trabalhar com diferentes tipos de dados em um programa. Apesar de internamente o computador manipular unicamente números, as linguagens de programação permitem que utilizemos outros tipos de informação nos programas de forma transparente.

Podemos definir um tipo de dados como um conjunto de objetos que tem comportamento similar.

Ler Anexo A.

Tipos de Dados

Existem 3 tipos básicos de dados:

Numéricos

Os dados numéricos se dividem em 2 grupos:

- inteiros
- reais

O conjunto dos números inteiros (\mathbb{Z}) contém um número infinito de elementos.

$$\mathbb{Z} = \{ -\infty, \dots, -3, -2, -1, 0, 1, 2, 3, \dots, +\infty \}$$

O conjunto dos números reais (\mathbb{R}) podem ser positivos ou negativos e possuem parte fracionária:

Exemplo:

$$\mathbb{R} = \{ -34.88, 0.0, 23.45, 346.89 \}$$

Tipos de Dados

Caractere ou Literal

São dados formados por um único caracter ou por uma cadeia de caracteres. Esses caracteres podem ser letras maiúsculas ou minúsculas, números (não podem ser usados para cálculo) e os caracteres especiais (&, #, @, ?, +).

Exemplo:

'aluno'

'1234'

'@ Internet'

'0.34'

'1 + 2'

Tipos de Dados

Lógicos

A existência deste tipo de dado é, de certo modo, um reflexo da maneira como os computadores funcionam. Muitas vezes, estes tipos de dados são chamados de booleanos, devido à significativa contribuição de GEORGE BOOLE à área da lógica matemática.

Podem assumir os valores **verdadeiro** e **falso**.

Exemplo:

V → valor lógico verdadeiro (true)

F → valor lógico falso (false)

Variáveis

Em um computador, manipulamos informações que, durante a execução de um programa, ficam armazenadas temporariamente em memória. Para que tais valores possam ser manipulados nas linguagens de programação, é preciso que haja algum identificador informando em que local da memória (endereço) esse elemento se encontra.

Podemos dizer que **Variáveis** são espaços (posições) reservados na memória do computador onde podemos armazenar valores de um determinado tipo (Conjunto de Valores)

Variáveis

As variáveis existem apenas em tempo de execução e durante este tempo elas são associadas a “nomes” chamados de **identificadores**.

Os identificadores são nomes que podem ser compostos de letras, números e do caractere “_” (sublinhado). Identificadores não podem ser iniciados por números.

Os nomes escolhidos devem explicitar seu conteúdo.

Exemplo:

var quantidade, numero : **inteiro**

nome : **caractere**

preco : **real**

aberto : **logico**

Exercícios

Classifique os dados especificados abaixo de acordo com seu tipo, assinalando com **I** os dados do tipo inteiro, com **R** os reais, com **L** os literais, com **B** os lógicos (booleanos), e com **N** aqueles para os quais não é possível definir a priori um tipo de dado.

() 0.21

() 1

() .V.

() "0."

() 1%

() "José"

() 0.35

() .F.

() +3257

() "a"

() "+3257"

() -1560

() "-0.0"

() "F"

() +/- 3

() "abc"

() C

() Maria

() +36

() .T.

() "V"

() 1.25

Variáveis

Atribuição

Já descrevemos onde os dados ficam armazenados, porém para que possamos manipulá-los é necessário primeiramente colocar um valor dentro da variável.

O ato de colocar ou atribuir um valor a uma variável é chamado de **atribuição** e é representado pelo símbolo \leftarrow ou $:=$.

Lembrando que somente valores do mesmo tipo da variável é que lhe podem ser atribuídos.

A atribuição é feita da seguinte maneira:

$\langle \text{identificador da variável} \rangle \leftarrow \langle \text{valor do mesmo tipo} \rangle$

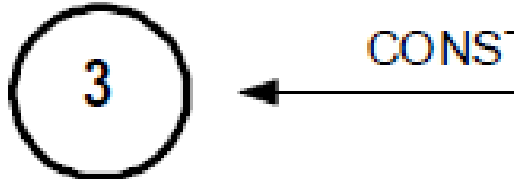
$\langle \text{identificador da variável} \rangle \leftarrow \langle \text{operações cujo resultado é do mesmo tipo da variável} \rangle$

Constantes

- Constante é um determinado valor fixo que não se modifica ao longo do tempo, durante a execução de um programa. Conforme o seu tipo, a constante é classificada como sendo numérica, lógica e literal.
- Exemplo de constantes:

$$\frac{N1+N2+N3}{3}$$

CONSTANTE

A diagram illustrating a constant. It features a circle containing the number 3. Above the circle is a horizontal line, and above that line is the expression N1+N2+N3. To the right of the circle, the word 'CONSTANTE' is written in blue capital letters. A horizontal arrow points from 'CONSTANTE' towards the circle containing the number 3.

Expressões

Em termos gerais, podemos definir uma expressão como uma fórmula matemática, em que um conjunto de variáveis e constantes numéricas se relacionam através de operadores aritméticos. Essa fórmula, quando avaliada, resulta num valor. Observe abaixo:

- $a = b^2 + c^2 - d$

NÃO UTILIZAR $\rightarrow 1 + \frac{3}{7} + \left[4 \times \frac{8-5}{9} \right] =$

UTILIZAR $\rightarrow 1 + 3/7 + (4*(8-5))/9 =$

Exercícios

1) Realizando a sequencia de atribuições abaixo, qual o valor final de X, Y e A

$X \leftarrow 10$

$Y \leftarrow 5$

$A \leftarrow 0$

$A \leftarrow X$

$X \leftarrow Y$

$Y \leftarrow A$

Exercícios

2) Realizando a sequencia de atribuições abaixo, qual o valor de nome, nome1, nome2 e nome3:

nome ← 'Paulo'

nome ← 'João'

nome1 ← '15'

nome2 ← '30'

nome3 ← nome1 + nome2

Exercícios

3) Realizando a sequencia de atribuições abaixo, qual o valor de soma e media:

num1 \leftarrow 200

num2 \leftarrow 300

soma \leftarrow num1 + num2

media \leftarrow soma / 2

Teste de Mesa

Para programadores iniciantes, ou não tão iniciantes assim, é difícil verificar se o algoritmo está executando corretamente a tarefa para o qual foi projetado. Lembrando que um algoritmo só está correto se produz o resultado esperado para qualquer entrada possível.

O Teste de Mesa é um meio pelo qual podemos acompanhar a execução de um algoritmo, passo a passo, ou instrução a instrução. Desta forma podemos encontrar erros e confirmar se a lógica utilizada está correta.

Teste de Mesa

Veja o Exemplo:

Nota da Prova 1

Nota da Prova 2

Nota da Prova 3

Nota da Prova 4

P1	P2	P3	P4	MEDIA
8	?	?	?	?
8	7	?	?	?
8	7	9	?	?
8	7	9	8	?
8	7	9	8	8

Operadores

Os operadores são meios pelo qual incrementamos, decrementamos, comparamos e avaliamos dados dentro do computador. Temos três tipos de operadores:

- Operadores Aritméticos
- Operadores Relacionais
- Operadores Lógicos

Operadores Aritméticos

Os operadores aritméticos são os utilizados para obter resultados numéricos. Além da adição, subtração, multiplicação e divisão, podem utilizar também o operador para exponenciação.

Os símbolos para os operadores aritméticos são:

Operação	Símbolo
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Exponenciação	**

Operadores Aritméticos

Hierarquia das Operações Aritméticas

1^o () Parênteses

2^o Exponenciação

3^o Multiplicação, divisão (o que aparecer primeiro)

4^o + ou – (o que aparecer primeiro)

Exemplo:

Total = Preço * Quantidade

$$1 + 7 * 2^{**}2 - 1 = 28$$

$$3 * (1 - 2) + 4 * 2 = 5$$

Outros Operadores

pot: é a potência entre dois números.

$$10 \text{ pot } 2 = 100$$

$$6 \text{ pot } 3 = 216$$

raiz: é a operação de radiciação entre dois numeros.

$$9 \text{ raiz } 2 = 3$$

$$216 \text{ raiz } 3 = 6$$

div: é resultado inteiro de uma divisão.

$$10 \text{ div } 4 = 2$$

$$22 \text{ div } 6 = 3$$

resto: é o resto da divisão inteira de dois números inteiros.

$$10 \text{ resto } 4 = 2$$

$$22 \text{ resto } 6 = 4$$

Operadores Relacionais

Os operadores relacionais são utilizados para comparar String de caracteres e números. Os valores a serem comparados podem ser caracteres ou variáveis.

Estes operadores sempre retornam valores lógicos (verdadeiro ou falso/ True ou False)

Para estabelecer prioridades no que diz respeito a qual operação executar primeiro, utilize os parênteses.

Operadores Relacionais

Operadores Relacionais
são:

Descrição	Símbolo
Igual a	=
Diferente de	<>
Maior que	>
Menor que	<
Maior ou igual a	>=
Menor ou igual a	<=

Tendo duas variáveis $A = 5$ e $B = 3$

O resultado das expressões
seriam:

Expressão	Resultado
$A = B$	FALSO
$A <> B$	VERDADEIRO
$A > B$	VERDADEIRO
$A < B$	FALSO
$A >= B$	VERDADEIRO
$A <= B$	FALSO

Operadores Lógicos

Os operadores lógicos servem para combinar resultados de expressões, retornando se o resultado final é verdadeiro ou falso.

Os operadores lógicos são:

Operador	
E	AND
OU	OR
NÃO	NOT

E / AND Uma expressão AND (E) é verdadeira se todas as condições forem verdadeiras

OR/OU Uma expressão OR (OU) é verdadeira se pelo menos uma condição for verdadeira

NOT Um expressão NOT (NÃO) inverte o valor da expressão ou condição, se verdadeira inverte para falsa e vice-versa.

Operadores Lógicos

A	B	A OU B	A E B	NÃO A
F	F	F	F	V
F	V	V	F	V
V	F	V	F	F
V	V	V	V	F

Tabela Verdade

E	OU	NÃO
V e V = V	V ou V = V	Não V = F
V e F = F	V ou F = V	
F e V = F	F ou V = V	Não F = V
F e F = F	F ou F = F	

Exercícios

1) Tendo as variáveis SALARIO, IR e SALLIQ, e considerando os valores abaixo. Informe se as expressões são verdadeiras ou falsas.

SALARIO	IR	SALLIQ	EXPRESSÃO	V OU F
100,0	0,00	100,0	$(SALLIQ \geq 100,0)$	
200,0	10,00	190,0	$(SALLIQ < 190,0)$	
300,0	15,00	285,0	$SALLIQ = SALARIO - IR$	

2) Sabendo que $A=3$, $B=7$ e $C=4$, informe se as expressões abaixo são verdadeiras ou falsas.

a) $(A+C) > B$ ()

b) $B \geq (A + 2)$ ()

c) $C = (B - A)$ ()

d) $(B + A) \leq C$ ()

e) $(C+A) > B$ ()

3) Sabendo que $A=5$, $B=4$ e $C=3$ e $D=6$, informe se as expressões abaixo são verdadeiras ou falsas.

a) $(A > C) \text{ AND } (C \leq D)$ ()

b) $(A+B) > 10 \text{ OR } (A+B) = (C+D)$ ()

c) $(A \geq C) \text{ AND } (D \geq C)$ ()

Anexo A

Armazenamento de Dados na Memória

Cada um dos diversos tipos de dados apresentados nesta apostila necessita de uma certa quantidade de memória para armazenar a informação representada por eles.

Esta quantidade é em função do tipo de dado considerado, do tipo da máquina (computador) e do tipo de linguagem de programação. Por isso, o que será exposto nos subitens seguintes não deve ser tomado como padrão, apenas como exemplo.

Anexo A

Armazenamento de Dados do Tipo Literal

Devemos sempre ter em mente que um byte consegue representar 256 (2^8) possibilidades diferentes.

Uma informação do tipo literal nada mais é do que um conjunto de caracteres que podem ser letras, dígitos ou símbolos especiais.

A união de todos os caracteres existentes nos computadores resulta num conjunto com um número de elementos menor que 256. Deste resultado surgiu a ideia de associar a cada caractere um número (código) variando de 0 a 255 (256 possibilidades). No princípio, cada fabricante de computador adotava uma convenção diferente para este código. Mais recentemente, esta convenção foi padronizada a fim de facilitar a portabilidade (migração) de programas entre máquinas diferentes. Esta convenção é representada na forma de uma tabela de mapeamento de caracteres em números.

Anexo A

Assim, cada célula de memória (byte) pode conter um caractere, representado pelo seu código ASCII.

Retornando à questão do armazenamento de informações do tipo literal na memória, deve-se lembrar que um dado deste tipo possui um certo comprimento dado pelo número de caracteres nele contido. Portanto, para guardar um dado do tipo literal devemos alocar (reservar) um espaço contíguo de memória igual ao comprimento do mesmo, destinando um byte para cada caractere da informação.

Anexo A

Exemplificando, a informação do tipo literal "banana" possui seis caracteres e, portanto, seis bytes são necessários para reter a referida informação na memória. A princípio, estes bytes podem estar em qualquer lugar da memória, mas é conveniente que estejam juntos (posições contíguas). A primeira posição deste conjunto de bytes é absolutamente arbitrária e sua escolha geralmente é feita automaticamente pelo compilador (isto é, pelo programa que traduz um outro escrito em alguma linguagem de programação para outra geral, a linguagem de máquina do computador com que se trabalha).

Endereço	Informação
0	b (98)
1	a (97)
2	n (110)
3	a (97)
4	n (110)
5	a (97)

Anexo A

Armazenamento de Dados do Tipo Lógico

Uma informação do tipo lógico só possui dois valores possíveis: .V. ou .F.. Assim, a princípio, um único bit seria suficiente para armazenar uma informação deste tipo. Contudo, deve-se lembrar que a menor porção de memória que se pode acessar é o byte. Portanto, uma informação do tipo lógico é armazenada em um byte de memória. De certa forma, se por um lado isto pode ser como um "desperdício" de memória, por outro simplifica bastante a arquitetura de memória dos computadores (por motivos que fogem ao contexto desta apostila). Além do mais, isto não é tão relevante, uma vez que na prática o número de ocorrências de dados do tipo lógico é bastante inferior ao de ocorrências de dados do tipo literal ou numérico.

Anexo A

Armazenamento de Dados do Tipo Inteiro

O conjunto dos números inteiros (\mathbb{Z}) contém um número infinito de elementos:

$$\mathbb{Z} = \{ -\infty, \dots, -3, -2, -1, 0, 1, 2, 3, \dots, +\infty \}$$

Obviamente é inviável o armazenamento de todos os números deste conjunto num computador. Faz-se necessário realizar um estudo para que se limite o número de elementos representáveis deste conjunto.

Se apenas um byte fosse utilizado para armazenar os dados do tipo inteiro, existiriam apenas 256 números diferentes neste conjunto:

$$\{-127, -126, \dots, -2, -1, 0, 1, 2, \dots, 127, 128\}$$

Esta restrição é bastante forte, uma vez que boa parte das aplicações práticas necessitam de números inteiros maiores que estes.

Se forem utilizados dois bytes para armazenar um número inteiro, o universo de números representáveis cresce para $28 \times 28 = 216 = 65.536$ possibilidades:

$$\{-32767, -32766, \dots, -2, -1, 0, 1, 2, \dots, 32767, 32768\}$$

Anexo A

Armazenamento de Dados do Tipo Real

O conjunto dos números reais (R) contém um número infinito de elementos e, pelas mesmas razões que o conjunto dos números inteiros, precisa ser limitado.

Para dados deste tipo julgou-se apropriado adotar quatro bytes para sua representação interna nos computadores.

São muito comuns situações como as aplicações científicas em que é necessária uma maior precisão de cálculo, intimamente ligada ao número de casas decimais dos dados. Para este caso, em analogia com o que acontece com os dados do tipo inteiro, algumas linguagens de programação decidiram criar dados do tipo real estendido (com oito bytes).