Universidade Católica de Brasília

Algoritmo e Programação Estruturada

Aula 3

Prof. Osmam Brás de Souto

TIPOS DE DADOS

Tipos primitivos ou escalares de informação processada pelo computador:

- **Inteiro**: toda informação numérica inteira (não fracionária) negativa, nula ou positiva. Exemplo: 100, 0, -3
- **Real**: toda informação numérica pertencente ao conjunto dos números reais (inteiros ou fracionários), (negativa, nula ou positiva). Exemplo: 12, 0, -3, 1.7, 101.5, 4.0 = símbolo de ponto
- **Caracter:** contém uma letra, um número ou um símbolo especial (caractere alfanumérico)
  - deve ser indicada entre aspas simples(' '):Exemplo: ' A', '1', '] '
- **Texto:** sequência contendo letras, números e símbolos especiais (caracteres alfanuméricos)
  - essa sequência deve ser indicada entre aspas duplas (" ")
  - Exemplo: "Taguatinga - DF", "356-9025", "Desconto 10%"
  - também chamado de string, cordão ou cadeia de caracteres
- **Logico**: conjunto de valores - *falso* ou *verdadeiro*
  - Esse tipo só apresenta um desses valores (excludentes)
  - também chamado de booleano

TIPOS DE DADOS

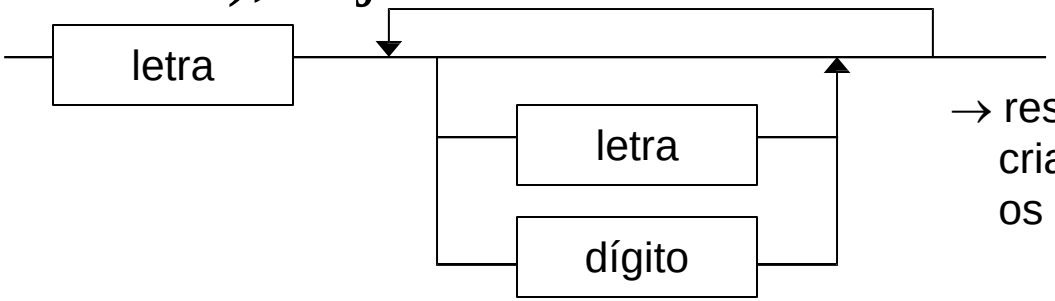
TIPOS DE DADOS BÁSICOS (escalares)

- Todos os outros tipos de dados em C são baseados em um desses cinco tipos (escalares)
- A definição do tipo de dados permite que o computador aloque e mantenha livre um espaço exato de memória que vai ser utilizado pelo “programa”

Tipo	**Tamanho em bits**	**Faixa**
char (caractere)	8	0 a 255
int (inteiro)	32	2147483647 a -2147483648
float (real)	32	3.4E-38 a 3.4E38 seis dígitos de precisão
double (real)	64	1.7E-308 a 1.7E308 dez dígitos de precisão
void (sem tipo)	0	sem valor

VARIÁVEL

- Tudo aquilo que é sujeito a variações, que é incerto, instável ou inconstante
- Representa um nome de um local onde se pode colocar um valor ou conjunto de valores
- O nome da variável é um **identificador (deve ser significativo -** cuidado com letras maiúsculas e minúsculas), cuja sintaxe é :

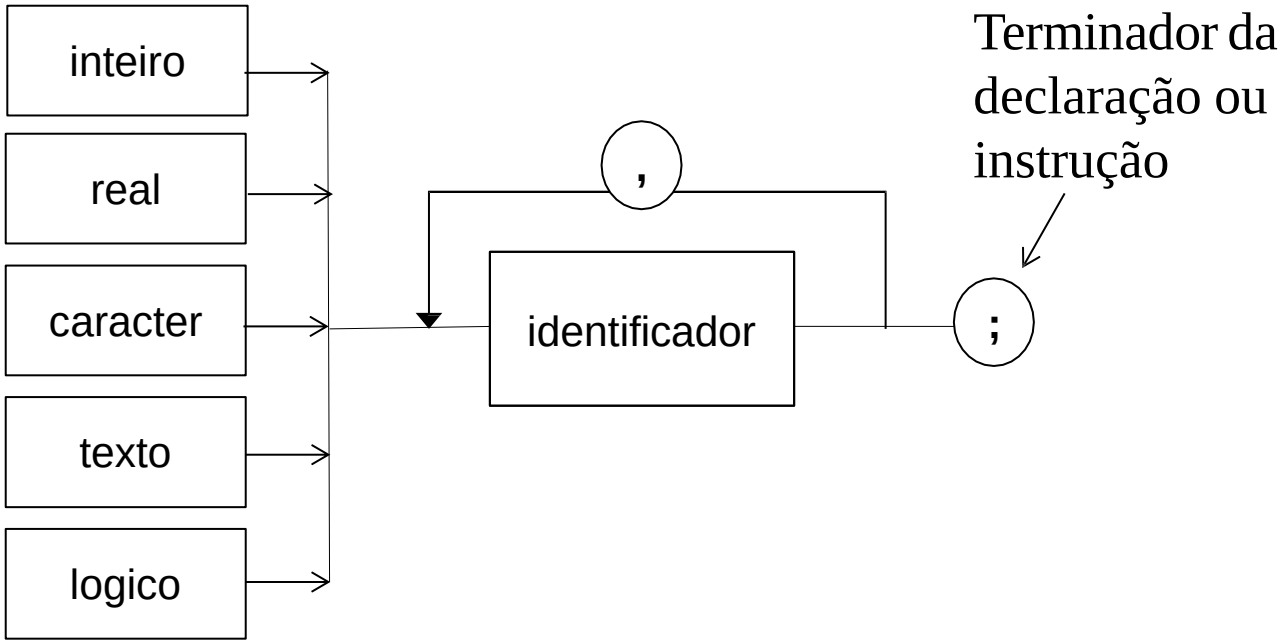


→ respeitar as regras para criar nomes válidos para os identificadores

Exemplo: nome, ~~1a~~, ~~#55~~, ~~o{2}~~, k4, achou, ~~"ano"~~, y1

DECLARAÇÕES DE VARIÁVEIS

- Todo dado a ser colocado na memória deve ser previamente identificado (usar nomes significativos)
  - primeiro saber qual o seu tipo e depois fazer o seu armazenamento
- A declaração de variável corresponde a criação de locais na memória com o nome da variável (identificador) marcado com o tipo que pode assumir
- Sintaxe:



DECLARAÇÕES DE VARIÁVEIS

Exemplos:

inteiro x1, numCorreto, conta;

real valor;

texto nome, frase, enderecoCliente;

caracter resposta;

logico achou;

Logo:

- **x1** é um local da memória onde só pode ser armazenado números inteiros
- **nome** é um local de memória que só pode ser armazenado caracteres alfanuméricos
- palavras sublinhadas são palavras-reservadas

OPERADORES

- **ARITMÉTICOS**

  +, - (binário ou unário), ***, / (** resultado da divisão real**)**

  \ (resultado da divisão inteira**),**

  % ou mod (resto da divisão inteira)

- **RELACIONAIS**

  == , != , > , >= , < , <=

- **LÓGICOS**

  E para conjunção

  OU para disjunção

  ! ou NAO para negação

- **CARACTERES**

  + (concatenação)

  (este operador só pode ser aplicado sobre textos**)**

EXPRESSÕES ARITMÉTICAS

EXPRESSÕES - Operadores Aritméticos

Operadores Binários

=	atribuição
+	adição
-	subtração
*	multiplicação
/	divisão (resultado depende dos operandos)
%	módulo (*mod*-resto da divisão)

Operadores Unário

+ -	menos (troca de sinal)

Incremento e Decremento

++	incremento
--	decremento

Prefixo a++ ou a--

Sufixo ++a ou --a

Precedência

1	++ --
2	* / %
3	+ -
4	=

EXPRESSÕES ARITMÉTICAS

EXPRESSÕES - Operadores Aritméticos de Atribuição

Estes operadores (+= , -= , *= , /= , %=) são usados com uma variável a sua esquerda e uma expressão a sua direita. A operação consiste em atribuir um novo valor à variável que dependerá do operador e da expressão à direita

<variável> <operador> = <expressão>

Exemplos:	i + = 2;	equivale	i = i + 2;
	x * = y + 1;	equivale	x = x * (y + 1);
	t / = 2.5;	equivale	t = t / 2.5;
	p % = 5;	equivale	p = p % 5;
	d - = 3;	equivale	d = d – 3;

EXPRESSÕES ARITMÉTICAS

- Precedência dos operadores:

  +, - (unitários)

  *, /, \ , % (mod)

  +, - (binários)

- **Cuidado**:

  2+3*5 ≠ (2+3)*5

  2+3/5 ≠ (2+3)/5

Dica: faça uso de parênteses no caso de dúvidas

EXPRESSÕES ARITMÉTICAS

- Exemplo

```
int main() {
    float valor, resultado;
    printf("Qual o valor agregado: \n");
    scanf("%f",&valor);
    resultado=(valor*2-1);
    printf("O resultado %f", resultado);
}
```

EXPRESSÕES RELACIONAIS

EXPRESSÕES - Operadores Relacionais

Todas as expressões relacionais retornam true ou false (verdadeiro ou falso respectivamente)

Operadores	
igualdade	= =
diferente	! =
maior que	>
menor que	<
maior ou igual	> =
menor ou igual	< =

IMPORTANTE

O valor zero (0) é considerado **FALSO** e qualquer valor diferente de zero é considerado **VERDADEIRO**, sendo representado pelo valor inteiro um (1), normalmente.

EXPRESSÕES RELACIONAIS

- ✓ Uso de operadores relacionais
- ✓ Utilizado em comparações (relações)
- ✓ Precedência dos operadores:

< , <=

==

> , >=,

!=

Dica: faça uso de parênteses no caso dúvidas

EXPRESSÕES LÓGICAS

- Uso de operadores lógicos
- Semântica pela **Tabela Verdade**:
  - Conjunto de todas as possibilidades combinatórias entre valores de diversas variáveis lógicas, que se encontram em duas situações e um conjunto de operadores lógicos

A	B	A e B	A ou B	não A
F	F	F	F	V
F	V	F	V	V
V	F	F	V	F
V	V	V	V	F

Precedência dos operadores: nao, e, ou

- Exemplo
  - 5 > 2 e 100/2.0 == 50 (**V**)
  - se é Sábado e nao é feriado, entao tem aula (Hoje tem aula) (**V**)
  - nao verdadeiro (**F**)

EXPRESSÕES LÓGICAS

EXPRESSÕES - Operadores Lógicos

Operador		Símbolo
E	(*and*)	**&&**
OU	(*or*)	‖
NÃO	(*not*)	**!**

Exemplos:

(E) se ((x == 5) **e** (y == 5)) entao
if ((x = = 5) **&&** (y = = 5)) ⇒ em programação

(OU) se ((x == 5) **ou** (y == 5)) entao
if((x==5)‖(y==5)) ⇒ em programação

(NÃO) se (**nao** (x == 5)) entao
if (**!** (x = = 5)) ⇒ em programação

→ que seria o mesmo que: se (x != 5) entao ou if (x != 5)

PRECEDÊNCIA GERAL

Primeiro: parênteses e funções

Segundo: expressões aritméticas

+, - (unitários)

*, /, \ , % (**mod)**

+, - (binários)

Terceiro: comparações

<, <=, == , >= , >, !=

Quarto: não , !

Quinto: e

Sexto : ou

Expressões Lógicas: Negação

Exemplos:

- nao (verdadeiro) ➔ falso
- nao (a e b) ➔ nao a ou nao b ➔ !a ou !b
- nao (a ou b) ➔ nao a e nao b
- nao (a == b) ➔ a **!=** b
- nao (a > b) ➔ a <= b
- ... (entre outras propriedades)

Universidade Católica de Brasília

Referência de Criação e Apoio ao Estudo

Material para Consulta e Apoio ao Conteúdo

- FARRER, H. et all, Algoritmos Estruturados, Editora LTC, 3ª . edição, 1999. - livro
  - Capítulo 0

- MANZANO, J. e Oliveira, J., Algoritmos, Lógica para desenvolvimento de programação, Editora Ética, 1996.
  - Capítulo 1