# SSC0101 - ICC1 – Teórica

## Introdução à Ciência da Computação I

# Estrutura de Programas e Tipos de Dados Simples

Prof. Vanderlei Bonato: vbonato@icmc.usp.br
Prof. Claudio Fabiano Motta Toledo: claudio@icmc.usp.br

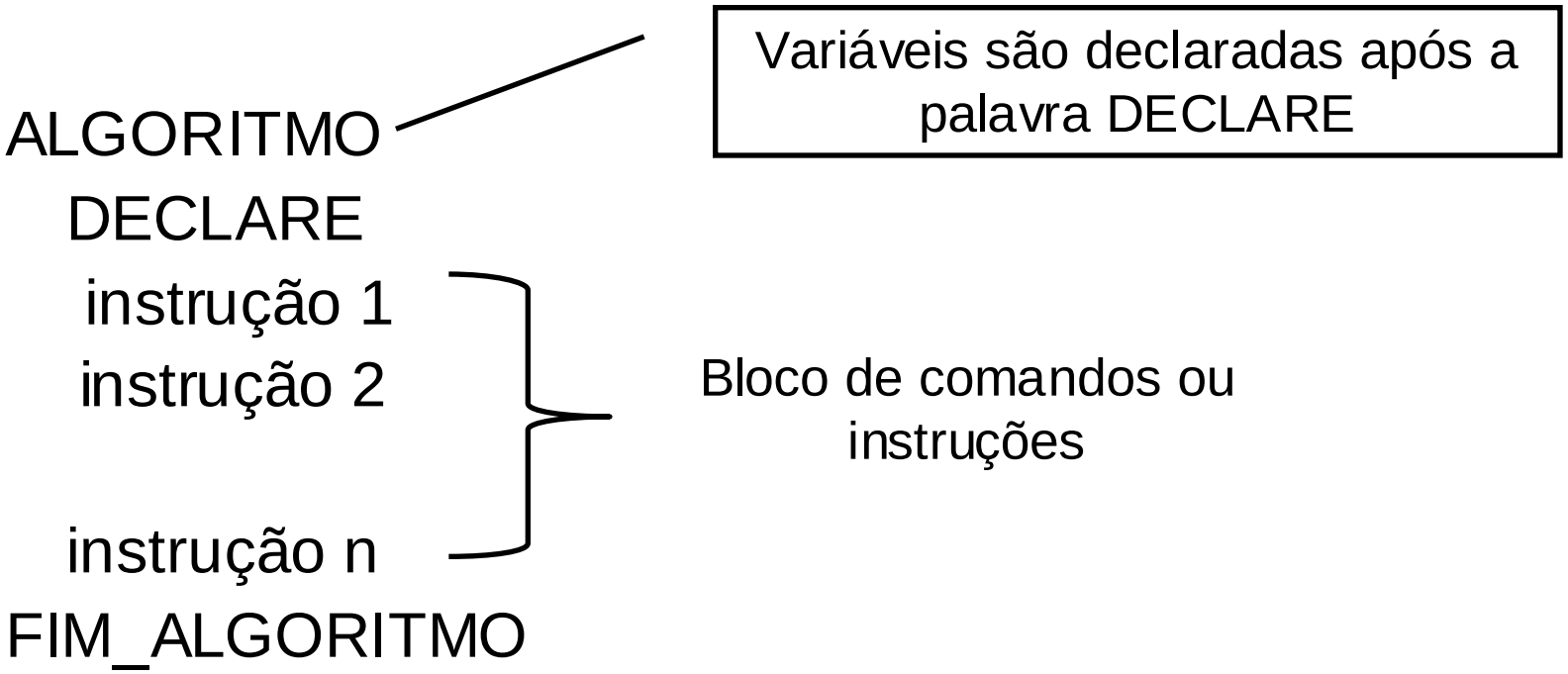
# Sumário

- Linguagem C
- Estrutura de Programas
- Tipos de Dados Simples
- Constantes em Linguagem C
- Entrada/Saída (E/S)

# Linguagem C

- Linguagem de propósito geral conhecida por ser eficiente, econômica e portável
- Padrão ANSI C
  - ✓ ANSI - American National Standards Institute
- Alguns compiladores que suportam ANSI C
  - ✓ GCC (GNU Compiler)
  - ✓ Microsoft Visual C/C++ Compiler
  - ✓ Borland C++ Compiler
  - ✓ ...

# Estrutura de programas - Algoritmos




- As instruções são executadas seqüencialmente a partir das declarações de variáveis, até que uma instrução de desvio ou FIM_ALGORITMO seja encontrada

# Estrutura de programas - Linguagem C

- Um programa em C é uma coleção de diretivas, pragmas, declarações, definições, blocos de comandos/instruções e funções
- Veja as diferenças em:
  - ✓ http://msdn.microsoft.com/en-us/library/aa315887(VS.60).aspx
- Um programa pode ser dividido em um ou mais arquivos fontes
  - ✓ É necessário compilar cada arquivo fonte e fazer o “link” dos arquivos objetos resultantes para tornar um programa executável
- Constantes e macros são normalmente organizadas em arquivos separados conhecidos como “header files”ou “include files” que podem ser referenciados a partir de arquivos fontes

# Estrutura de programas - Linguagem C

```
#include <nome_da_biblioteca>
void main()
{
   instrução 1;
   instrução 2;

   instrução n;
}
```

- As instruções são executadas seqüencialmente a partir da função main(), até que uma instrução de desvio ou de retorno seja encontrada

# Tipos de Dados Simples - Algoritmos

- Serão considerados três tipos básicos em algoritmos:

  NUMÉRICO

  LITERAL

  LÓGICO.

- Tipos de Dados Numéricos: inteiros ou reais

| **Inteiros** | **Reais** |
|---|---|
| -23 | 23.45 |
| 98 | 346.89 |
| 0 | 0.0 |
| 237 | -34.88 |
| -2 | -247.0 |

# Tipos de Dados Simples - Algoritmos

- Tipos de Dados Lógicos ou booleanos
  - ✓ Assumem valores verdadeiro ou falso

- Tipos de Dados Literais ou Caracteres
  - ✓ Formados por um único caractere ou por uma cadeia de caracteres
  - ✓ Os caracteres podem ser letras maiúsculas, minúsculas, números e caracteres especiais (&, #, @, ?, +).
  - ✓ Exemplo: “aluno”, “123”, “@ internet”, “0.34”, “1+2”

- Os números, enquanto caracteres, não podem ser usados para cálculos

# Tipos de dados simples – Linguagem C

- Há 4 principais tipos de dados simples em C:
  - ✓ int
  - ✓ float
  - ✓ double
  - ✓ char

- Qualificadores:
  - ✓ short e long: definem tamanhos
  - ✓ unsigned: define valores que são sempre positivos

# Tipos de dados simples – Linguagem C

- Tipo **int**
  - ✓ Tamanho de 16 bits com valor máximo 32767 $=2^{15}-1$
  - ✓ **short int** ocupa normalmente 16 bits com o mesmo intervalo
  - ✓ **long int** ocupa 32 bits com valor máximo 2.147.483.647 $=2^{31}-1$
  - ✓ **unsigned int** representa valores inteiros entre 0 a 65535
  - ✓ **unsigned int** representa valores inteiros entre 0 a 4.294.967.295
  - ✓ **short** não pode ser maior que **int**, que não é maior do que **long**
  - ✓ Declaração:

```
int start, end;
short int  valores;
long int   volume;
```

# Tipos de dados simples – Linguagem C

- Tipo **float** e **double**: ponto flutuante
  - ✓ **float** representa valores em precisão simples
  - ✓ **double** representa valores em precisão dupla
  - ✓ Aumentam a precisão, mas exigem maior tamanho:
    32 bits para **float** e 64 bits para **double**
  - ✓ **long double** representa ponto flutuante com precisão estendida: 128 bits
  - ✓ Exemplo: 3.14 e 7.89E+5
  - ✓ Declaração:
    **float** peso;
    **double** area;
    **long double** volume;

# Tipos de dados simples – Linguagem C

- Tipo **char**

  - ✓ Armazena conjunto de caracteres: pontuação, letras, dígitos, espaço, etc

  - ✓ O caractere ocupa 1 byte variando de -128 a 127

  - ✓ **unsigned char** representa valores entre 0 e 255

  - ✓ Exemplo: sexo='M';

  - ✓ Declaração:

    **char** sexo, nome[];

# Tipos de dados simples – Linguagem C

| Tipo | Tamanho (bytes) | Abrangência dos Valores |
|---|---|---|
| char | 1 | -128 a 127 |
| unsigned char | 1 | 0 a 255 |
| int | 2 | -32768 a 32767 |
| unsigned int | 2 | 0 a 65535 |
| short int | 2 | -32768 a 32767 |
| long int | 4 | -2.147.483.648 a 2.147.483.647 |
| unsigned long int | 4 | 0 a 4.294.967.295 |
| float | 4 | $\pm 3,4 \cdot 10^{-38}$ a $\pm 3,4 \cdot 10^{38}$ |
| double | 8 | $\pm 1,7 \cdot 10^{-308}$ a $\pm 1,7 \cdot 10^{308}$ |
| long double | 10 | $\pm 3,4 \cdot 10^{-4932}$ a $\pm 3,4 \cdot 10^{4932}$ |

- Faixa (*range*) de acordo com o padrão ANSI (considerada mínima)

# Tipos de dados simples – Linguagem C

| Tipo | Tamanho (bytes) | Abrangência dos Valores |
|---|---|---|
| char | 1 | -128 a 127 |
| unsigned char | 1 | 0 a 255 |
| **int** | **4** | **-2.147.483.648 a 2.147.483.647** |
| **unsigned int** | **4** | **0 a 4.294.967.295** |
| short int | 2 | -32768 a 32767 |
| long int | 4 | -2.147.483.648 a 2.147.483.647 |
| unsigned long int | 4 | 0 a 4.294.967.295 |
| float | 4 | $\pm 3,4 \cdot 10^{-38}$ a $\pm 3,4 \cdot 10^{38}$ |
| double | 8 | $\pm 1,7 \cdot 10^{-308}$ a $\pm 1,7 \cdot 10^{308}$ |
| long double | 10 | $\pm 3,4 \cdot 10^{-4932}$ a $\pm 3,4 \cdot 10^{4932}$ |

- O tamanho e faixa de valores podem variar de acordo com o compilador ou processador

# Constantes em Linguagem C

- Constantes inteiras
  - **int**: 7438
  - **long**: 100431044**L** ou 100431044**l**
  - **unsigned long**: 100431044**UL** ou 100431044**ul**
- Constantes de ponto flutuante
  - **double** por definição: 7438.95 ou 1.23e-3
  - **float**: 7438.95**f** ou 1.23e-3**F**: float (obrigatório sufixo f ou F)
  - Contêm ponto decimal (7438.95) ou expoente (1.23e-3)

# Constantes em Linguagem C

- Constantes de caractere
  - ‘x’ ou ‘0’
  - Constantes de caractere são mais utilizadas em comparações com outros caracteres
- Constante de enumeração
  - Lista valores inteiros constantes, começando em 0
  - Exemplos:
  - ✓ enum booleano{FALSE, TRUE} - FALSE tem valor 0 e TRUE tem valor 1
  - ✓ enum dias_uteis {SEGUNDA=1, TERCA, QUARTA, QUINTA, SEXTA}

TERCA =2, QUARTA = 3, QUINTA=4, SEXTA=5.

# Constantes em Linguagem C

- ✓ O valor de uma constante não pode ser alterado
- ➢ Qualificador **#define**
  - ➢ Substituído na fase de pré-processamento

```
#define MAX 1000
char vetor[MAX+1];
```

- ➢ Qualificador **const**
  - ➢ Utilizado/Explorado em tempo de compilação

```
const double pi = 3.14159265;
const char nome[] = “Pedro”;
const char nome[5] = “Pedro”;
const char sexo = ‘F’;
```

- ➢ Qualificador **enum**

```
enum dias_uteis {SEGUNDA=-2, TERCA, QUARTA, QUINTA, SEXTA};
printf("%d\n",QUINTA); //IMPRIME = 1
```

# Comandos de entrada e saída - Algoritmos

- O comando de entrada recebe dados digitados pelo usuário e o comando de saída mostra os dados na tela

- Comando de entrada utilizado em algoritmos:

LEIA X

- Exemplo: considere X uma variável do tipo Numérico e Y uma variável do tipo Literal

LEIA X

- ✓ Armazena um valor numérico digitado pelo usuário na variável X

LEIA Y

- ✓ Armazena um ou vários caracteres digitados pelo usuário na variável Y

# Comandos de entrada e saída - Algoritmos

- Comando de saída utilizado em algoritmos:

ESCREVA X

- ✓ Os dados podem ser conteúdos de variáveis ou mensagens

- Exemplo:

ESCREVA X

- ✓ exibe o valor armazenado na variável X

ESCREVA “Conteúdo de Y = “, Y

- ✓ Exibe a mensagem “Conteúdo de Y= “ e em seguida o valor armazenado na variável Y

# Comandos de entrada e saída em C

- Comandos de entrada mais utilizados:

```
/*armazena um ou mais caracteres na variável “nome” */

gets(nome);

/*armazena um valor em x */
scanf(&x);
```

- Comando de saída

```
/*mostra o número inteiro armazenado na variável “x”)
printf (“%d”, x);
```

# Exemplo com o scanf e printf

```
#include <stdio.h>
#include <stdlib.h>

void main()
{
  /* Declaracao de variaveis int, float e char */
  int parcela_1, parcela_2, resultado_a;
  char oper;

  /* E/S – solicitacao de dados digitados no teclado */
  printf(“Informe as parcelas 1 e 2: ”);
  scanf(“%d %d”, &parcela_1, &parcela_2);

  /* E/S – apresentacao do resultado da soma */
  oper = ‘+’;
  resultado_a = parcela_1 + parcela_2;
  printf(“%d %c %d = %d \n”, parcela_1, oper, parcela_2, resultado_a);
  system(“PAUSE”);
}
```

# Exercícios

Apresente o algoritmo em pseudocódigo e o código em linguagem C em cada um dos exercícios

1. Faça um programa que receba quatro números inteiros, calcule e mostre a soma desses números

2. Faça um programa que receba duas notas, calcule e mostre a média ponderada dessas notas, considerando peso 2 para a primeira e peso 3 para a segunda

3. Faça um programa que receba uma hora (uma variável para hora e outra para minutos), calcule e mostre:
   a) a hora digitada convertida em minutos
   b) o total dos minutos
   c) o total dos minutos convertidos em segundos

# Referências


- Ascencio, A.F.G; Campos, E.A.V. *Fundamentos da Programação de Computadores:* algoritmos, pascal, C/C++ e java. 2ª Edição. São Paulo: Pearson Prentice Hall, 2007, 434p.

- Kernighan, B.W.; Ritchie, D.M. C, *A Linguagem de Programação*: padrão ANSI. 2ª Edição. Rio de Janeiro: Campus, 1989, 290p.

# FIM Aula 3