

Programação de Computadores IV

Introdução ao C - Variáveis

Slides de Bruno Augusto Dorta

Marques

Roteiro da aula

- Introdução a linguagem C
 - Organização do código-fonte
 - Método main
- Variável
 - Tipos básicos
 - Declaração
 - Atribuição

Programação em C

- Compilador compatível:
 - **GCC (Linux), Mingw(gcc)(Windows)**, Visual Studio, Intel Compiler, ...
- Editor de código-fonte:
- IDE:
 - **Code::Blocks**, Dev-C++, Visual Studio, QtCreator, ...
- Editores de texto:
 - Atom, Sublimetex, Gedit, nano, emacs, vim, ...

Boas práticas de programação

- Documentação do Código Fonte
 - A documentação do código fonte é feita com o uso de linhas de comentários.
 - Um comentário descreve a finalidade um comando ou de um bloco de código
 - Deve ser feito em linguagem natural (português ou inglês)
- Sintaxe para um comentário:
 - Um comentário sempre se inicia com o símbolo barra dupla “//”.
 - Sempre quando o compilador identificar um comentário “//” o restante da linha será ignorada pelo computador.
- Exemplo de comentário:

// Esse é um programa de exemplo

// A função abaixo imprime na tela uma mensagem de boas vindas

Sintaxe da linguagem C

- Regras básicas:
- Todo comando é delimitado pelo símbolo ; (ponto-virgula)
 - Exceção: comandos que abrem ou fecham escopo com o símbolo { ou }
- É permitido linhas em branco no código fonte. O compilador irá ignorá-las.
- Palavras reservadas devem ser escritas em minúsculo

Método main

- O método main é o ponto inicial do programa.
- Quando executamos um programa, o método main é o primeiro a ser chamado.
- O método main possui um bloco de comando, esse bloco é delimitado por chaves { }
- A ultima linha antes de fechar o bloco de comando é “return 0;”
- Sintaxe em C:

```
int main (void)
{

return 0;
}
```

Primeiro comando

- Imprimir na tela. Esse comando imprime uma sequência de caracteres na tela do computador.

SINTAXE:

```
printf("essa é uma mensagem");
```

Meu primeiro programa

```
// Este programa imprime a expressão
// "Hello World" na saída padrão

#include<stdio.h>

int main(void)
{
    printf("Hello World!");
    return 0;
}
```

Boas práticas de programação

- Indentação

- Sempre ao abrir um novo bloco de código (escopo), você deve dar um TAB ou espaços para agrupar o código em uma mesma indentação.
- Ela serve como um parágrafo para os códigos.

```
// Este programa imprime a expressão
// “Hello World” na saída padrão

#include<stdio.h>

int main(void)
{
    printf(“Hello World!”);
    return 0;
}
```

Variável

- Representa uma região na memória, onde pode ser armazenado um dado
- Possui :
 - Tipo
 - Nome (ou identificador)
 - Valor

Variável

- Durante a execução do algoritmo, a variável pode ter seu valor alterado (seu valor pode variar)
- Mudança de valor:
 - Entrada de dados: Ler Variavel1
 - Por atribuição: Variavel2 $\leftarrow$ 34
- Ao utilizarmos o nome de uma variável em um algoritmo, estamos acessando o ultimo valor armazenado nela.

Tipos básicos

- Inteiro
 - Denota valores numéricos que pertencem ao conjunto dos inteiros (negativos, positivos ou nulos)
 - Pode ter um valor máximo ou mínimo dependendo da linguagem de programação.
 - Na linguagem C a quantidade máxima é de -2^{31} a 2^{31}
- Exemplos de números inteiros:
 - Quantidade de alunos em um curso : 30
 - Idade de uma pessoa : 18
 - Número de uma casa: 213

Tipos básicos

- Real
 - Denota valores numéricos que pertencem ao conjunto dos números reais (negativos, positivos ou nulos)
 - Pode ter um valor mínimo e máximo
 - Em C o número real pode variar de 10^{-308} a 10^{308}
- Exemplos de números reais:
 - Média de um aluno: 6,5
 - Valor de um produto: 450,37
 - Peso de uma pessoa: 65,2

Tipos básicos

- Caractere
 - Denota valores que pertencem ao conjunto de sequência de caracteres
 - Alfabéticos: a-z, A-Z
 - Numéricos: 0-9
 - Especiais: ! @ # \$ % * () ? { } []
- Exemplos
 - Nome de uma pessoa: “Bruno Marques”
 - Endereço: “Rua Passo da Pátria, 97, Boa Viagem, Niterói”

Tipos básicos

- Lógico
 - Denota duas situações: Verdadeiro ou Falso
 - Na linguagem C não temos o tipo lógico por padrão.
- Exemplos
 - Aluno Aprovado: Verdadeiro
 - $(2 > 3)$: Falso

Exemplo de tipos de variáveis

Encontre os erros:

- Inteiro: endereço, nFilhos
- Caractere: idade, x
- Real: preco, peso
- Lógico: passouteste

Exemplo de tipos de variáveis

Encontre os erros:

- Inteiro: endereço, nFilhos
- Caractere: idade, x
- Real: preco, peso
- Lógico: passouteste

Declaração correta:

- Inteiro: nfilhos, idade, x
- Caractere: endereço
- Real: preço, peso, x
- Lógico: passouteste

Nomes ou identificadores

- Os nomes das variáveis (ou identificadores), indicam a área de memória onde o valor fica armazenado
- Regras para criação de identificadores
 - O primeiro caractere será sempre uma letra
 - Os nomes não possuem espaço em branco, nem símbolos especiais
 - Não pode ter acentos (linguagem Java é uma exceção)
 - Não pode ser uma palavra reservada da linguagem.
 - Em C, há distinção entre caractere maiúsculo e minúsculo.

Quais identificadores a seguir são válidos?

- Ah!
- X
- 1identificador
- Idade
- CEP
- Endereco_Cliente
- Dia/Mês/Ano
- Mes1
- AB*C
- Minha Idade
- Mês
- TeStE

Quais identificadores a seguir são válidos?

- Ah!
- X
- 1identificador
- Idade
- CEP
- Endereco_Cliente
- Dia/Mês/Ano
- Mes1
- AB*C
- Minha Idade
- Mês
- TeStE

Declaração de Variáveis

- Antes de atribuir um valor a uma variável é necessário declará-la
- A declaração avisa ao compilador para alocar espaço de memória, onde será armazenado o valor corrente.
- Ao declarar uma variável é necessário indicar o tipo de variável:
 - O tipo define a quantidade de memória a ser armazenada
 - Permite ao compilador fazer verificações semânticas

Declaração de Variáveis

- Devemos declarar todas as variáveis que utilizarmos.
- A declaração deve ser feita **antes** de utilizarmos uma variável
- Uma Variável só guarda dados de um único tipo
 - A variável será do tipo dos dados que ela pode guardar

Variáveis na linguagem C

- Tipo Inteiro:
- Utilizamos a palavra reservada `int`

```
// Esta é um exemplo de declaração de variáveis do tipo int:
```

```
int numero ;  
int x, y, z ;  
int variavel ;
```

Variáveis na linguagem C

- Tipo Real:
- Temos variáveis de precisão simples (32 bits) e precisão dupla (64 bits)

Precisão simples:

- Para variáveis de precisão simples utilizamos a palavra reservada **float**
 - Permite armazenar valores numéricos reais com precisão de até sete casas decimais
 - Permite armazenar valores de magnitude de 10^{-38} a 10^{38}
 - O ponto separa a parte inteira da fracionada
- Exemplo: 0.8946314

Variáveis na linguagem C

- Tipo Real:
- Temos variáveis de precisão simples (32 bits) e precisão dupla (64 bits)

Precisão dupla:

- Para variáveis de precisão dupla utilizamos a palavra reservada **double**
 - Permite armazenar valores numéricos reais com precisão de até quinze casas decimais
 - Permite armazenar valores de magnitude de 10^{-308} a 10^{308}
 - O ponto separa a parte inteira da fracionada
- Exemplo: 8.5654321361614

Variáveis na linguagem C

- Tipo Real:
- Temos variáveis de precisão simples (32 bits) e precisão dupla (64 bits)

```
// Este é um exemplo de declaração de variáveis do tipo float e  
// double:
```

```
float numero ;
```

```
float x, y, z ;
```

```
double variavel ;
```

```
double omega, theta ;
```

Variáveis na linguagem C

- Tipo Caractere
- Para representar um único character, utilizamos a palavra reservada **char**
 - Valores possíveis : **'a', 'b', 'c', 'd'**

```
// Esta é um exemplo de declaração de variáveis do tipo character:
```

```
char letra;    // isso é uma variável que guarda um único caractere
```

Variáveis na linguagem C

- Tipo String ou sequência de caracteres
- Para representar um texto ou palavra utilizamos uma string.

A declaração de uma string segue uma regra específica:

- `char nome [tam]`
 - `Nome` é o nome da variável
 - `Tam` é o tamanho máximo da string
- Valores possíveis : “palavra”, “nome”, “rua niterói, número 50”

```
// Esta é um exemplo de declaração de variáveis do tipo string:
```

```
char variável[50];    //isso é uma variável que guarda sequencia de caracteres de tamanho 50
```

```
char endereço[100], nome[100];
```

Variáveis na linguagem C

- Tipo Lógico
- Não existe na linguagem C
- Podemos simular uma variável lógica utilizando um `int` onde:
- Valor 0 corresponde ao FALSO
- Valor 1 corresponde ao VERDADEIRO

Atribuições de Variáveis

- Na linguagem de programação C a atribuição de um valor é dada pelo símbolo = (igual)
- O símbolo igual é utilizado para atribuição de valores, não confundir com o operado lógico de igualdade !
- Na declaração de variáveis é obrigatório o uso da seguinte regra:
 - A variável vai a esquerda e o valor a direita do símbolo de atribuição (igual)

Sintaxe:

```
// Este é um exemplo de declaração e atribuição de valor:
```

```
int x;    // declaração de um inteiro
```

```
x = 42;   // atribuição do valor 42 a variável x
```

Atribuições de Variáveis

Sintaxe:

```
// Este é um exemplo de declaração e atribuição de valor à variáveis:
```

```
int x, y;      // declaração de inteiros  
float preco;   // declaração de número real  
double sigma; // declaração de número real  
char c;        // declaração de caractere
```

```
x = 9;  
y = -13;  
preco = 2.99;  
sigma = 0.8252646641;
```

```
c = 'a';
```

Atribuições de Variáveis para String

Para manipulação de String, utilizamos a biblioteca `string.h` do C.

Para utiliza-la, insira no inicio do código (antes do método `main`) o seguinte trecho:

```
#include<string.h>
```

Sintaxe:

```
// Este é um exemplo de declaração e atribuição de valor à variáveis:  
  
char palavra[100]; //declaração de sequência de caractere  
  
strcpy(palavra, "programação");
```

Declaração com valor inicial

É possível agrupar um comando de declaração com atribuição de um valor inicial

```
// Este é um exemplo de declaração e atribuição de valor inicial:
```

```
int x = 0;  
float preco = 1.99;  
char mensagem[100] = "Olá, bem vindo";
```

Exercícios

Encontre os erros na declaração e atribuição de variáveis. Altere o código para que não haja mais erros.

```
// O Código abaixo esta incorreto, corrija os erros:
```

```
int w, z, h;           // declaração de inteiros
float preco;           //declaração de número real
char[100] palavra;     //declaração de sequência de caractere
char produto[100];     //declaração de sequência de caractere
```

```
w = 3
z = 2.85;
x = 5;
8 = h;
preco = 2,99;
palavra = "programação";
strcpy(produto, notebook );
```

Exercícios

Encontre os erros na declaração e atribuição de variáveis. Altere o código para que não haja mais erros.

Solução:

```
int w, h, x;           // declaração de inteiros
float preco, z;        // declaração de número real
char palavra[100];     // declaração de sequência de caractere
char produto[100];
```

```
w = 3;
z = 2.85;
x = 5;
h = 8;
preco = 2.99;
strcpy(palavra, "programação");
strcpy(produto, "notebook");
```

Exercícios

Supondo que as variáveis **nota**, **nome**, **mat** e **sexo** sejam utilizadas para armazenar a nota do aluno, o nome do aluno, o número da matrícula e o sexo de um aluno. Declare as variáveis corretamente, associando o tipo primitivo adequado ao dado que será armazenado.

Exercícios

Supondo que as variáveis **nota**, **nome**, **mat** e **sexo** sejam utilizadas para armazenar a nota do aluno, o nome do aluno, o número da matrícula e o sexo de um aluno. Declare as variáveis corretamente, associando o tipo primitivo adequado ao dado que será armazenado.

Solução:

```
float nota;  
int matricula;  
char nome[100], sexo[100];
```

Exercícios

Complete o código abaixo, atribuindo um valor válido a cada variável

Código do exercício:

```
float nota;  
int matricula;  
char nome[100] , sexo[100];
```

Exercícios

Complete o código abaixo, atribuindo um valor válido a cada variável

Código do exercício:

```
float nota;  
int matricula;  
char nome[100] , sexo[100] ;  
  
nota = 9.5;  
matricula = 19283712;  
strcpy(nome, "Bruno Marques");  
strcpy(sexo, "Masculino");
```