

Laboratório de Programação

Prof. Dr. Paulo Rogério de Almeida Ribeiro

Coordenação do Curso de Engenharia da Computação

Variáveis, entrada e saída de dados



Roteiro

- Variáveis;

Roteiro

- Variáveis;
- Atribuições;

Roteiro

- Variáveis;
- Atribuições;
- Estrutura básica de um programa;

Roteiro

- Variáveis;
- Atribuições;
- Estrutura básica de um programa;
- Entrada e saída de dados.

Variáveis

Variáveis

- Locais para armazenamento de valores (memória);

Variáveis

- Locais para armazenamento de valores (memória);
- Devem ter: **TIPO** e **NOME**;

Variáveis

- Locais para armazenamento de valores (memória);
- Devem ter: **TIPO** e **NOME**;
- Declara-se antes de usá-la.

Variáveis

Tipos numéricos básicos

- `int`

Variáveis

Tipos numéricos básicos

- **int**
 - Armazena valores numéricos inteiros;

Variáveis

Tipos numéricos básicos

- **int**
 - Armazena valores numéricos inteiros;
 - Exemplo: 0, -3, 5000, 10000;

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;
- 4 bytes de memória.

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;
- 4 bytes de memória.

- **double**

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;
- 4 bytes de memória.

- **double**

- Armazena números com ponto flutuante com precisão dupla.

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;
- 4 bytes de memória.

- **double**

- Armazena números com ponto flutuante com precisão dupla.
- Exemplo: 3.14159265;

Variáveis

Tipos numéricos básicos

- **int**

- Armazena valores numéricos inteiros;
- Exemplo: 0, -3, 5000, 10000;
- *short, long, signed, unsigned.*
- 2 bytes de memória.

- **float**

- Armazena números com ponto flutuante com precisão simples.
- Exemplo: 3.14, 7.2;
- 4 bytes de memória.

- **double**

- Armazena números com ponto flutuante com precisão dupla.
- Exemplo: 3.14159265;
- 8 bytes de memória.

Variáveis

Tipo textual básico

- char

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;
- Texto entre aspas simples 'a';

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;
- Texto entre aspas simples 'a';
- Exemplo: 'd', 'z', '3'.

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;
- Texto entre aspas simples 'a';
- Exemplo: 'd', 'z', '3'.
- 1 byte de memória.

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;
- Texto entre aspas simples 'a';
- Exemplo: 'd', 'z', '3'.
- 1 byte de memória.

C não tem string!

Variáveis

Tipo textual básico

- **char**

- Armazena caracteres simples;
- Texto entre aspas simples 'a';
- Exemplo: 'd', 'z', '3'.
- 1 byte de memória.

C não tem string!

Utiliza-se um vetor de char para frases.

Variáveis

Outros tipos básicos

- void

Variáveis

Outros tipos básicos

- **void**

- Em inglês void quer dizer vazio;

Variáveis

Outros tipos básicos

- **void**

- Em inglês void quer dizer vazio;
- Aplicações: função sem retorno, função sem parâmetro, ponteiro etc;

Variáveis

Outros tipos básicos

- **void**

- Em inglês void quer dizer vazio;
- Aplicações: função sem retorno, função sem parâmetro, ponteiro etc;

C não tem booleano!

Variáveis

Outros tipos básicos

- **void**

- Em inglês void quer dizer vazio;
- Aplicações: função sem retorno, função sem parâmetro, ponteiro etc;

C não tem booleano!

Mas pode-se utilizar um int: 0 (falso) e $\neq 0$ (1, 2 etc - verdadeiro)

Tipos de dados - padrão ANSI

Tipo	Tamanho aproximado em bits	Faixa mínima
char	8	-127 a 127
unsigned char	8	0 a 255
signed char	8	-127 a 127
int	16	-32.767 a 32.767
unsigned int	16	0 a 65.535
signed int	16	O mesmo que int
short int	16	O mesmo que int
unsigned short int	16	0 a 65.535
signed short int	16	O mesmo que short int
long int	32	-2.147.483.647 a 2.147.483.647
signed long int	32	O mesmo que long int .
unsigned long int	32	0 a 4.294.967.295
float	32	Seis dígitos de precisão
double	64 -	Dez dígitos de precisão
long double	80	Dez dígitos de precisão

Variáveis

Como declarar?

Variáveis

Como declarar?

```
tipo _variável nome _variável;
```

Variáveis

Como declarar?

tipo _variável nome _variável;

Exemplo: int num;

Variáveis

Como declarar?

tipo_variável nome_variável;

Exemplo: int num;

Exemplo: char letra;

Variáveis

Como declarar?

tipo _variável nome _variável;

Exemplo: int num;

Exemplo: char letra;

Exemplo: float num2;

Variáveis

Como declarar?

tipo _variável nome _variável;

Exemplo: int num;

Exemplo: char letra;

Exemplo: float num2;

C é uma linguagem fortemente tipada.

Nome das variáveis

- *Case sensitive*: diferencia maiúsculas e minúsculas
 - Ex: Nome, NOME e NoMe



- Pode conter letras maiúsculas, minúsculas, números e _



- Nome não pode **começar** com números
 - Ex: 0nome, 1nome



- Não podemos utilizar como parte do nome:
{ (+ - * / \ ; . , ?



Nome das variáveis

Palavras reservadas

auto	double	int	struct	break
enum	register	typedef	char	extern
return	union	const	float	short
unsigned	continue	for	signed	void
default	goto	sizeof	volatile	do
if	static	while		

Variáveis

Atribuição de valor

Variáveis

Atribuição de valor

Atribui valores para variáveis

Variáveis

Atribuição de valor

Atribui valores para variáveis

Variável = valor ;

Variável = expressão ;

Variáveis

Atribuição de valor

Atribui valores para variáveis

Variável = valor ;

Variável = expressão ;

Ex: a = 2;
 b=4;
 soma= a+b;

A declaração da variável pode ser com ou sem atribuição

A declaração da variável pode ser com ou sem atribuição

- Sem atribuição de valor: **TIPO NOME;**
- Exemplo:
 - `int num;`
 - `char nome;`

A declaração da variável pode ser com ou sem atribuição

- Sem atribuição de valor: **TIPO NOME;**
- Exemplo:
 - `int num;`
 - `char nome;`
- Com atribuição de valor: **TIPO NOME = VALOR;**
- Exemplo:
 - `int num=3;`
 - `int a,b,c=0;`

Estrutura básica de um programa em C

#include <stdio.h>

// Declaração de bibliotecas/headers usados

int main(void) {

Obrigatória

//Função principal do programa -

corpo do programa

//Comandos (variáveis, atribuições, entradas...)

return 0;

}

Estrutura básica de um programa em C

#include <stdio.h>

// Declaração de bibliotecas/headers usados

int main(**void**) {

Obrigatória

//Função principal do programa -

corpo do programa

//Comandos (variáveis, atribuições, entradas...)

return 0;

}



Atenção ao ; (ponto e vírgula)

Comentários

Comentários

- Uma linha: `//`
 - `printf("Hello world!");` `//Esta parte não é considerada`

Comentários

- Uma linha: `//`
 - `printf("Hello world!");` `//Esta parte não é considerada`
- Várias linhas: `/* */`
 - `printf("Hello world!");` `/* Esta parte não é considerada */`

Saída de dados

Função printf

- Exibição de mensagens

```
printf("mensagem");
```

```
printf("texto com %d", variavel);
```


Exemplo de código com printf
Primeiro programa em C

```
#include <stdio.h>
int main() {
printf ("Hello world");
return 0;
}
```

Exemplo de código com printf

```
#include <stdio.h>
int main(void) {
    int x=10;
    printf("O valor de x é %d", x);
    return 0;
}
```

Relação código e tipo

Código	Elemento armazenado
%d ou %i	inteiro
%c	um único caractere
%f	número em ponto flutuante
%o	inteiro do sistema octal
%x	inteiro do sistema hexadecimal
%ld	valor do tipo long
%e	número na notação científica
%s	uma cadeia de caracteres

Entrada de dados

Função scanf

- Permite a recepção de dados de entrada
`scanf("tipo de dado", &nomeDaVariavel);`

Exemplo de código com scanf

```
#include <stdio.h>
int main() {
int dias;
scanf("%d", &dias);    // scanf("tipo de dado",
&variável);
return 0;
}
```

Exemplo de código com scanf

```
#include <stdio.h>
int main() {
int dias;
scanf("%d", &dias);    // scanf("tipo de dado",
&variável);
return 0;
}
```

Como mostrar o valor armazenado na variável “dias” ?

Outras funções para entrada de dados


- `getchar()`
 - `caractere = getchar();`
- `gets()`
 - `gets(caractere);`

Operadores matemáticos


Operador	Ação
-	Subtração, também menos unário
+	Adição
*	Multiplicação
/	Divisão
%	Módulo da divisão (resto)
--	Decremento
++	Incremento

Operadores matemáticos

- Incremento:

$x = x + 1$  $x++$ (pos-incremento)
 $++x$ (pre-incremento)

- Decremento:

$x = x - 1$  $x--$ (pos-decremento)
 $--x$ (pre-decremento)

Operadores relacionais

Operador

>

>=

<

<=

==

!=

Ação

Maior que

Maior que ou igual

Menor que

Menor que ou igual

Igual

Diferente

Operadores lógicos

Operador

&&

||

!

Ação

AND

OR

NOT

Exercícios

Exercícios

- 1) Faça um programa que calcule a área de um retângulo.

Exercícios

- 1) Faça um programa que calcule a área de um retângulo.
- 2) Faça um programa que receba 3 notas e calcule a média aritmética entre elas.