

Curso Técnico de Informática Integrado

# **LINGUAGEM C:**

## **Estrutura, tipos de dados e operadores**

**Professor Welison de Brito**

**Disciplina** Lógica e Linguagem de Programação

# Objetivos

- **Aplicar os conceitos à linguagem C:**
  - Tipos de dados
  - Variáveis e constantes
  - Funções printf e scanf.
  - Operadores aritméticos

# LINGUAGEM “C”

- **Linguagem de alto nível**
- **Linguagem compilada**
- Utilizado em programas como:
  - Pacote Office
  - Sistemas Operacionais Windows, Apple OS
  - Bancos de dados como o MySQL

# Nosso 1º programa

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

```
int main ( ){  
    printf("Hello world!");  
}
```

Imprimirá a mensagem **“Hello world!”**

# Estrutura

- Como toda linguagem de programação, há uma estrutura que o programa deve seguir para sua execução;

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

```
int main ( ){  
    printf("Hello world!");  
}
```

# Estrutura

C →

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

Portugol →

```
algoritmo "Nome "
var
    teste: caractere
inicio
    escreva("Hello world!")
fimalgoritmo
```

# Arquivo stdio.h

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

```
int main ( ){  
    return 0;  
}
```

- A linha `#include <stdio.h>` diz ao compilador que ele deve incluir o arquivo- cabeçalho `stdio.h`.
- Neste arquivo existem declarações de funções úteis para entrada e saída de dados
- **Stdio:**
  - **std** = standard, padrão em inglês;
  - **io** = Input/Output, entrada e saída;

# *Função principal*

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

```
int main ( ){  
    return 0;  
}
```

- A linha **int main()** indica que estamos definindo uma função de nome **main**.
- Todos os programas em C têm que ter uma função **main**.
  - É esta função que será chamada quando o programa for executado.



# *Delimitadores de bloco*

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

```
int main ( ){  
    return 0;  
}
```

- O conteúdo da função é delimitado por chaves { }.
- Todo bloco de instrução será delimitado pelas chaves;
- As instruções serão encerradas por um ponto e vírgula (;)

# Retorno da função

```
#include <stdio.h>

int main ( ){
    return 0;
}
```

- A última linha do programa, **return 0**, indica o número inteiro que está sendo retornado pela função;
- **Indica que está tudo certo;**
- Não discutiremos agora
  - Não é obrigatório

# *Mostrando informação na tela*

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

```
int main ( ){  
    printf("Informática");  
}
```

- O "printf" é usado para saída, que normalmente é enviada para o monitor.
- Equivale ao **escreva**, usado no português

# Tipos de variáveis

VisuAlg	C	Tamanho (bits)	Exemplo de dado	Como declarar
Inteiro	int	16 bits	12	int idade;
real	float	32 bits	6.371	float raio;
	double	64 bits	-1.602176565E-19;	double carga
logico	bool	1 bit	TRUE ou FALSE	bool ok;
caractere	char	8 bits	'm'	char sexo;
	string		"Masculino"	char nome[255] char nome [ ]
	void	nenhum		

# String (cadeia de caracteres)

- Uma string é um conjunto de caracteres entre aspas.
- Por exemplo, “**você é um vencedor**” é uma string
  - composta por várias letras que formam a frase.
- Não confunda strings com caractere.
  - Um caractere simples fica entre dois apóstrofos, por exemplo 'a'.
  - Entretanto “a” é uma string que contém somente uma letra.

# Declaração de Variáveis (PORTUGOL)

**<nome>:<tipo>;**

**Idade: inteiro;**



Define uma variável  
do tipo inteiro

# Declaração de Variáveis

**<tipo> <nome>;**

**int idade;**



Define uma variável  
do tipo inteiro

**int a = 678;**



Define uma variável  
do tipo inteiro e  
define o valor de  
678;

# Declaração de Variáveis

- **a = 678** → Ocorre a atribuição do valor do tipo inteiro.

```
#include <stdio.h>
int main (){
    float x;
    float y;
    float z;
}
```

Variáveis do mesmo tipo  
em linhas distintas;



```
#include <stdio.h>

int main (){
    float x, y, z;
}
```

Variáveis do mesmo tipo  
na mesma linha;



# Atribuição de valores

- Enquanto no portugol usávamos os símbolos
  - 
    - `num ← 10`
  - 
    - `num:=10`
- Em C usamos apenas a **igualdade (=)**
  - `Num = 10`

# Atribuição de valores

```
#include <stdio.h>
int main (){
    int num, a = 678;
    num = 10;
    printf("Hello world");
}
```

- **a = 678** → Ocorre a atribuição do valor do tipo inteiro.
- **num = 10** → Também ocorre atribuição do tipo inteiro.

**Qual a diferença?**

# CONSTANTES

- São identificadores cujos valores **não podem ser alteradas** no programa.
- Podem ser declarado como quaisquer dos tipos abordados anteriormente

# CONSTANTES

- Podem ser declaradas (criadas) de duas formas:

- Diretiva de pré-processamento

**#define**

- Palavra-chave **const**.

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

```
#define PI 3.14
```

```
const int MIN=0, MAX=1000;
```

```
int main (){
```

```
const float X = 0.2;
```

```
printf("%d", a);
```

```
}
```

# PRINCIPAIS DELIMITADORES

## DELIMITADORES

Visualg	Função	C
//	delimitadores de comentário	/* */ ou //
Não faz uso	separar comandos e terminar declarações	;
,	separar identificadores e parâmetros	,
<=	atribuição de valor	=
inicio ... fim algoritmo	delimita blocos de instruções	{ ... }
( ... )	delimita teste de condição	( ... )

# Imprimindo valor da variável

- A função **printf()** tem a seguinte forma geral:

```
printf (string_de_controle, lista_de_argumentos);
```

- A string de controle mostra
  - caracteres que devem ser colocados na tela
  - Quais as variáveis e suas respectivas posições usando a notação **%**

# Imprimindo valor da variável

`printf (string_de_controle, lista_de_argumentos);`

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

```
int main (){
```

```
    int a = 10;
```

```
    printf("%d", a);
```


```
    printf("%s", "Welison");
```

```
}
```

Imprime valor da  
variável



Imprime a string  
"Welison"



# Formatação

String de controle	Tipo de dado que representa
<b>“%d”</b>	Inteiro
<b>“%c”</b>	caractere
<b>“%f”</b>	ponto flutuante (real)
<b>“%s”</b>	string
<b>“%lf”</b>	double



# Imprimindo valor da variável

```
#include <stdio.h>

int main (){
    float a = 1000;
    printf("%.2f", a);
}
```

%.2f controla  
quantidade casas  
decimais

# Caracteres Especiais (escape)

Caractere	Significado	Exemplo em C	Saída esperada
<code>\n</code>	Nova linha	<code>printf("Olá\nMundo");</code>	Olá Mundo
<code>\t</code>	Tabulação (tab)	<code>printf("A\tB");</code>	A B (espaço em tab)
<code>\'</code>	Plica simples (aspas simples)	<code>printf("\'Olá\');</code>	'Olá'
<code>\"</code>	Aspas duplas	<code>printf("\"Olá\");</code>	"Olá"
<code>\\</code>	Barra invertida (\)	<code>printf("\\");</code>	\

# Imprimindo valor da variável

As strings de controle podem ser utilizadas em conjunto com textos

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

```
int main (){  
    int idade = 15;  
    char nome[100] = "Welison";  
    printf("Olá %s, sua idade é %d anos", nome, idade);  
}
```

Representa a variável  
**nome**

Representa a variável  
**idade**

# Dúvidas?



Clique para adicionar o título

# Obrigado!