

CT62A

COMPUTAÇÃO 1

Aula 04 - Introdução à Linguagem C

Profs. Rafael **Mantovani** e **Adalberto** Lazarini

Roteiro



- 1** Introdução
- 2** Estrutura Básica em C
- 3** Entrada e Saída
- 4** Referências

Roteiro



- 1** Introdução
- 2** Estrutura Básica em C
- 3** Entrada e Saída
- 4** Referências

Introdução

- **Linguagem de programação** é conjunto de ferramentas, regras de sintaxe e símbolos, ou códigos que nos permitem escrever programas de computador
- A primeira e mais primitiva linguagem de computador é a própria linguagem de máquina (0s e 1s)
- Antigamente, um programa era difícil de ser criado, o processo era longo e caro
- Essa complexidade levou à necessidade de desenvolver novas técnicas e ferramentas

Introdução

- A linguagem C nasceu na década de 70 (Dennis Ritchie) sendo padronizada pela ANSI (1989)
- A linguagem C foi desenvolvida a partir das linguagem B (Ken Thompson) e BCPL (Martin Richards)
 - Inseriu conceito de tipos de dados
 - Independente de hardware
- A linguagem C é genérica e utilizada para a criação de diversos tipos de programas:
 - Planilhas e editores de texto, sistemas operacionais, programas de automação, banco de dados, automação industrial, etc ...

Introdução

- Comumente chamada de linguagem de “nível médio”
 - Combina elementos de alto nível com funcionalidade assembly (linguagem de baixo nível)
 - Manipula bits, bytes e endereços

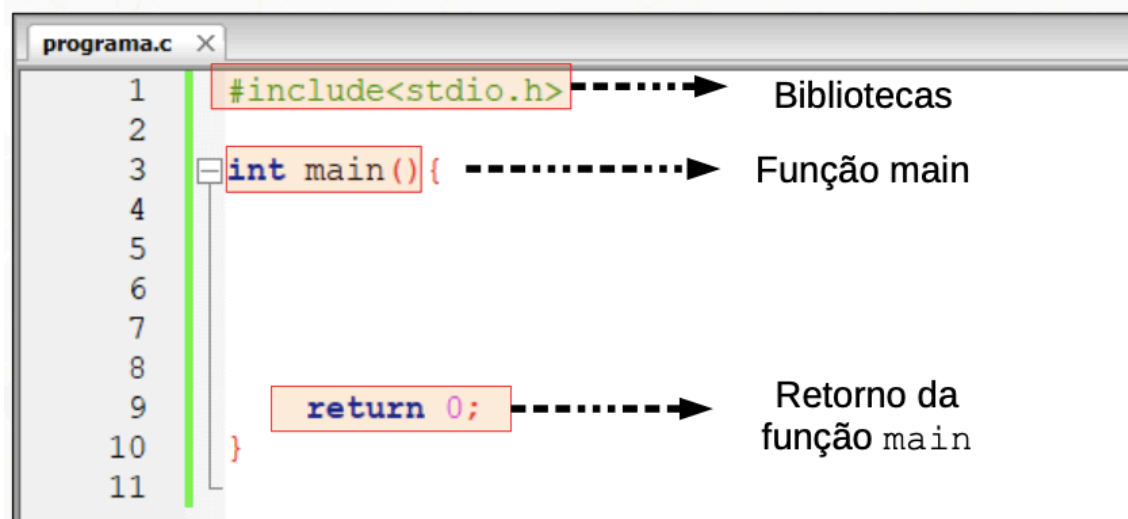
- Linguagem portátil
 - Possível adaptar o mesmo software a diferentes tipos de computadores
 - **Obs:** desde que siga os padrões (ANSI, etc)

Roteiro



- 1 Introdução
- 2 Estrutura Básica em C
- 3 Entrada e Saída
- 4 Referências

Estrutura básica em C



Indica o fim da execução da função *main*

- **Bibliotecas** contém conjunto de funções e instruções previamente estabelecidas e que podem ser usadas pelo programa
- A execução do programa inicia-se pela função **main**

IDE

- ***Integrated Development Environment (IDE)***
 - Interface gráfica
 - Facilita a edição do código-fonte
 - Facilita a compilação e análise de código
 - Facilita a depuração do código
 - Possibilita o desenvolvimento de projeto de software maiores



Informações Gerais

- Uma variável sempre deve ser declarada antes de ser usada
 - Declarar indica ao compilador para reservar um espaço na memória para guardar o valor dessa variável

Exemplos de declaração

```
int main(){  
    int num1;  
    int num2;  
    float resultado;  
    float multiplicacao;  
    .  
    .  
    .
```

=

```
int main(){  
    int num1, num2;  
    float resultado, multiplicacao;  
    .  
    .  
    .
```

Roteiro



- 1** Introdução
- 2** Estrutura Básica em C
- 3** Entrada e Saída
- 4** Referências

Entrada e saída de dados



Entrada:

```
scanf(formato, &variável);
```

Exemplos:

```
scanf("%f", &n);  
scanf("%d", &num);  
scanf("%d", &cont);
```

Saída:

```
printf(conteúdo, [parâmetros]);
```

Exemplos:

```
printf("%f", n);  
printf("Exame");  
printf("O numero e' %d", num);  
printf("Sao necessarias %d moedas", cont);
```

Ambas as funções scanf e printf pertencem a biblioteca stdio.h

Saída de dados

Sintaxe geral do comando printf

```
printf("Mensagem escrita na tela", lista_argumentos);
```

Exemplos:

```
int num = 5;
```

```
char letra = 'C';
```

```
float pi = 3.1415;
```

```
printf("O valor de num e' %d", num);
```

```
printf("O valor de num e' %d e o valor de letra e' %c", num, letra);
```

```
printf("num = %d, letra = %c e pi = %f", num, letra, pi);
```

%c → caractere (char)
%d ou %i → inteiro (int)
%f → ponto flutuante (float)

Saída de dados

- Códigos usados na função `printf()`

Código	Formato
<code>%c</code>	Caractere
<code>%d</code> ou <code>%i</code>	Inteiro decimal com sinal
<code>%e</code>	Notação científica
<code>%f</code>	Ponto flutuante decimal
<code>%g</code>	Menor representação entre <code>%f</code> e <code>%e</code>
<code>%o</code>	Octal sem sinal
<code>%s</code>	<i>String</i> de caracteres
<code>%u</code>	Inteiro sem sinal (<code>unsigned int</code>)
<code>%x</code>	Hexadecimal sem sinal
<code>%%</code>	Símbolo %

Saída de dados

Caractere	Significado
\a	Caractere (invisível) de aviso sonoro (bip)
\n	Caractere (invisível) de nova linha
\t	Caractere (invisível) de tabulação horizontal
\\	Caractere de barra invertida '\'
\'	Caractere de aspas simples
\''	Caractere de aspas duplas
\?	Sinal de interrogação

Obs.: Não deve conter espaço entre \ e o símbolo desejado

Saída de dados com acentos

```
#include<stdio.h>
#include<locale.h> //para usar a função setlocale()

int main(){
    printf("Apareça acentos: á, ê, ã ...\\n");
    //Altera a codificação de saída para o padrão do S.O., em Português...

    setlocale(LC_ALL, "");
    printf("Apareça acentos: á, ê, ã ...\\n");

    return 0;
}
```

Observação: A alteração no Windows vale também para número! O padrão para casas decimais é mostrar com ponto ".", e com o `setlocalte()` mostrará com vírgula ","

Entrada de dados

- A função `scanf()` permite que o usuário forneça dados pelo teclado para o programa

```
scanf("expressão de controle", lista de argumentos);
```

Indica ao compilador o TIPO de dado está se esperando que o usuário digite

- **“Expressão de controle”:** `%formato`, parecido com o `printf...`
- **Argumento:** `&nome_variável`
 - Indica o endereço de memória onde se armazena o conteúdo de `nome_variável`

```
scanf("%i", &variavel_inteira);  
scanf("%f", &variavel_float);  
scanf("%c", &variavel_char);
```

Entrada de dados

- Relembrando alguns aspectos sobre variáveis:
 - Possui um nome e tipo;
 - É associado a um endereço de memória;
 - Ao utilizar uma variável, acessa-se o seu conteúdo/valor localizado no endereço de memória associado a essa variável;
 - Ao atribuir um valor para uma variável, modifica-se o seu conteúdo/valor;
 - O **&** serve para obter o endereço de memória associado a uma variável;

```
int idade = 10;
printf("Endereco de memoria associado a variavel idade: %p \n",&idade);
printf("Valor armazenado por idade: %i \n", idade);
scanf("%i", &idade); //recebe um int e armazena no end. de memória associado à variável idade
printf("Endereco de memoria associado continua igual: %p \n",&idade);
printf("Novo valor armazenado por idade: %i \n", idade);
```

Exemplo de saída e entrada

```
#include<stdio.h>

int main(){
    int idade;

    printf("Digite sua idade: ");
    scanf("%i", &idade);

    printf("Sua idade e': %i anos", idade);
    printf("Daqui 5 anos você terá %i anos", idade + 5);

    return 0;
}
```

Comentários


- Em regra, são utilizados para a documentação do código;
- Objetiva facilitar o entendimento do programa por outros programadores ou o próprio criador;
- Comentários na linguagem C podem ser feitos de duas formas:

`/* <código> */` Para um determinado trecho de código (várias linhas)

`// <código>` Para uma linha de código, sendo colocado em seu início

Roteiro



- 1** Introdução
 - 2** Estrutura Básica em C
 - 3** Entrada e Saída
 - 4** Referências
- 

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br

Prof. **Adalberto** Lazarini

adalbertoz@utfpr.edu.br