

INTRODUÇÃO



LINGUAGEM C
(PARTE 1)

DIA 2

TÓPICOS DE HOJE



- **VISÃO GERAL**
 - UTILIDADES
 - DIFERENÇAS
- **CÓDIGOS EM C**
 - BLOCOS
 - DECLARAÇÃO E ATRIBUIÇÃO
 - INCLUDES
 - ENTRADA E SAÍDA
 - FUNÇÃO MAIN
 - TIPOS PRIMITIVOS
 - OPERAÇÕES
 - FUNÇÕES

DIN

Departamento de
Informática



VISÃO GERAL

Utilidades

- Robótica
- Sistemas embarcados
- Base para outras linguagens
- Base para sistemas operacionais

VISÃO GERAL

Diferenças da linguagem C

Em comparação com outras linguagens, C difere em:

- Compilação (diferente das interpretadas)
- Sintaxe (uso de "{}" e ";")
- Tipagem (fortemente tipada)

CÓDIGOS EM C

Blocos de código

- Em C, o código é escrito em blocos de código
- Os blocos de código iniciam por "{" e encerram por "}"
- Cada instrução no bloco é finalizada com ";"

CÓDIGOS EM C

Exemplo

```
1 {  
2     int a;  
3     a = 3;  
4     int b = 4;  
5     a = a + b;  
6 }
```

CÓDIGOS EM C

Blocos de código

Observações:

- Não é necessária a indentação
- A instrução é delimitada por ";"
 - Pode haver várias instruções em uma linha
 - Pode haver uma instrução em várias linhas

CÓDIGOS EM C

Exemplo

```
1  {int a; a = 3; int b = 4; a =
2  a + b;}
```

CÓDIGOS EM C

Declaração de variável

- A declaração de uma variável pode ser entendida como a sua "criação"
- Uma variável só poderá receber um valor após ser declarada em algum ponto anteriormente

CÓDIGOS EM C

Declaração de variável

- A declaração é, na realidade, alocar memória para a variável, isto é, reservar um lugar na memória para guardar o seu valor
- Uma declaração de variável é escrita no formato:

<tipo> <nome>;

CÓDIGOS EM C

Exemplo

```
1 short contador_repeticao;
2 int quantidade_de_dias;
3 float porcentagem_de_vendas;
4 double valor_funcao_exponencial;
```

CÓDIGOS EM C

Bloco - escopo local

- Uma observação importante é que variáveis declaradas (criadas) dentro de um bloco de código estão no escopo local do bloco
 - Isto significa que elas serão destruídas após o término do bloco

CÓDIGOS EM C

Código com erro

```
1 {  
2     int a;  
3     a = 2  
4 }  
5 a = a + 1;
```

CÓDIGOS EM C

Bloco – escopo local

- No exemplo visto, a variável é declarada dentro do bloco de código e tenta-se atribuir um valor à esta variável fora do bloco de código
- Como a variável "não existe" fora do bloco de código, o código apresenta erro

CÓDIGOS EM C

Bloco - escopo local

- Desta forma, a variável deve ser declarada em um escopo condizente com as operações que serão realizadas com ela
- No exemplo, a variável deve ser declarada fora do bloco de código para que possa ser utilizada fora do bloco de código

CÓDIGOS EM C

Código sem erro

```
1 int a;
2 {
3     a = 2
4 }
5 a = a + 1;
```

CÓDIGOS EM C

Atribuição de valor

- Já atribuição de valor é guardar um valor no espaço de memória relacionado à uma variável
- É realizado no código no formato
 <nome> = <valor ou expressão>;

CÓDIGOS EM C

Atribuição de valor

- A atribuição de valor à uma variável pode ser feita juntamente à sua declaração
- Neste caso, escreve-se:
`<tipo> <nome> = <valor ou expressão>;`

CÓDIGOS EM C

Exemplo

```
1 int dias_por_ano = 365;
2 char valor_pixel = 64;
3 double pi = 3.141592653;
```

CÓDIGOS EM C

Includes

- Para utilizar recursos (por exemplo funções) de uma biblioteca, é necessário incluí-los ao início do código
- Também é possível incluir recursos de outros códigos em C

CÓDIGOS EM C

Exemplo

```
1 // Incluir a biblioteca stdio
2 // (standard input-output):
3 #include <stdio.h>
4 // Incluir o código
5 // funcoes_matematicas
6 #include "funcoes_matematicas.c"
```

CÓDIGOS EM C

Entrada e saída de dados

- Na linguagem C, é possível realizar entrada e saída de dados utilizando a biblioteca stdio (Standard Input-Output)
- Para tanto, como visto anteriormente é necessário realizar o include da biblioteca ao início do código

CÓDIGOS EM C

Entrada e saída de dados

- Tendo incluído a biblioteca, é possível utilizar a função printf() para saída de dados e scanf() para entrada de dados
- printf("Hello, World!"); por exemplo, exibirá a frase entre aspas no ambiente em que o programa está sendo executado

CÓDIGOS EM C

Entrada e saída de dados

- Já scanf("%d %d", &x, &y); atribuirá a x e y dois valores inteiros digitados pelo usuário separados por um espaço (o primeiro valor - antes do espaço - para x, e o segundo valor - depois do espaço - para y)

CÓDIGOS EM C

Entrada e saída de dados

- `printf("%d\n", x);` exibe o valor de x e acrescenta uma quebra de linha após

CÓDIGOS EM C

Exemplo

```
1 #include <stdio.h>
2
3 int numero_1 = 1;
4 int numero_2 = 2;
5 printf("%d + %d = %d",
6         numero_1, numero_2,
7         numero_1 + numero_2);
8
9 // Resultado exibido:
10 // 1 + 2 = 3
```

CÓDIGOS EM C

Função main

- A função main corresponde ao fluxo geral do código em C
- Veremos a definição e como utilizar funções ao final desta aula
- O código deve ser escrito a partir da função main

CÓDIGOS EM C

Função main

- A função main pode ser escrita:
 - void main(){bloco de código}
- ou:
 - int main(){bloco de código com return}
- Ou seja, caso o tipo de retorno seja diferente de void deve haver um return acessível no bloco

CÓDIGOS EM C

Exemplo

```
1 #include <stdio.h>
2
3 void main(){
4     printf("Hello, World!");
5 }
```

CÓDIGOS EM C

Exercício

- Perguntar ao usuário o seu nome e sua idade e exibir na tela as informações digitadas no formato:

Nome: <nome informado>

Idade: <idade informada>

CÓDIGOS EM C

Tipos de dados

- Em C, há diversos tipos de dados para representar valores diferentes. Podemos dividí-los em:
- Valores inteiros
- Valores fracionários
- Valores textuais (caracteres)

CÓDIGOS EM C

Tipos de dados

Em ordem crescente de tamanho (em bytes)

- Inteiros:
 - char (1), short (2), int (4), long (8)
- Fracionários:
 - float (4), double (8)
- Textual (caracter):
 - char (1)

CÓDIGOS EM C

Tipos de dados

- Para escrever um valor inteiro, basta escrever sua representação decimal
- Para um float, escreve-se:

 <valor inteiro>. <valor fracionário>f

Observação: no float há um "f" no final

- Para um double, escreve-se:

 <valor inteiro>. <valor fracionário>

CÓDIGOS EM C

Tipos de dados

- Para escrever um char, escreve-se:
'<carácter>'

Observação: o char também pode ser usado como um inteiro de -128 à 127

- Para escrever uma string (vetor ou array de char = char[]), escreve-se:
"conteúdo da string"

CÓDIGOS EM C

Exemplo

```
1 char valor_pixel = 123;
2 short dias_por_ano = 365;
3 int tempo_decorrido_em_segundos = 5670478;
4 long milissegundos_decorridos = 976547899;
```

CÓDIGOS EM C

Exemplo

```
1 float valor_logaritmo = 2.15842f;  
2 double valor_pi = 3.141592653;  
3 char primeira_letra = 'a';  
4 char vogais[] = "aeiou";
```

CÓDIGOS EM C

Operações

- Em C, é possível realizar operações (que recebem dois ou mais operandos e geram um resultado)
- É possível realizar operações aritméticas (soma, multiplicação, etc.), relacionais e lógicas

CÓDIGOS EM C

Operações aritméticas

- Dentre as operações aritméticas há a operação de soma (+), subtração (-), multiplicação (*), divisão (/) e resto da divisão inteira (%)

CÓDIGOS EM C

Exemplo

```
1 double soma = 1.3 + 5.1; // = 6.400000
2 int subtracao = 240 - 123; // = 117
3 float multiplicacao = 323 * 0.7f; // = 226.099991
```

CÓDIGOS EM C

Operações aritméticas

- O resto da divisão inteira é exemplificado por:
- $10 \% 3 = 1$, pois $10 = 3 * 3 + 1$,
- ou $13 \% 5 = 3$, pois $13 = 5 * 2 + 3$
- Além disso, esta operação só pode ser realizado entre inteiros

CÓDIGOS EM C

Exemplo

```
1 int resto_1 = 10%3; // = 1
2 int resto_2 = 13%5; // = 3
3 int resto_3 = 21%7; // = 0
```

CÓDIGOS EM C

Operações aritméticas

- No caso da divisão utilizando "/", pode ser tanto a divisão inteira quanto a divisão fracionária

CÓDIGOS EM C

Operações aritméticas

- Caso ambos os operandos da divisão sejam inteiros, será realizada divisão inteira
- Caso contrário (pelo menos um dos operandos é fracionário), será realizada divisão fracionária

CÓDIGOS EM C

Exemplo

```
1 short resultado_inteiro_1 = 9/2; // = 4
2 int resultado_inteiro_2 = 20/7; // = 2
3 double resultado_fracionario_1 = 1.1/10.0; // = 0.110000
4 float resultado_fracionario_2 = 5/2.0f; // = 2.500000
```

CÓDIGOS EM C

Exercícios

- Fazer um programa que calcula a área de um círculo dado seu diâmetro
- Fazer um programa que calcula a temperatura em graus Celsius dada uma temperatura em graus Fahrenheit

CÓDIGOS EM C

Operações relacionais

- Em C, 0 é interpretado como valor lógico falso e os demais valores como valor lógico verdadeiro
- Operadores relacionais comparam valores e retornam um valor lógico

CÓDIGOS EM C

Operações relacionais

- Os operadores relacionais são:
x igual a y ($x == y$), x diferente de y ($x != y$), x maior que y ($x > y$), x maior ou igual a y ($x >= y$), x menor que y ($x < y$), e, por fim, x menor ou igual a y ($x <= y$)

CÓDIGOS EM C

Operações relacionais

- Por exemplo $5 \leq 13$ produz valor lógico verdadeiro, pois 5 é menor ou igual a 13.
- Já $13 \leq 5$ produz valor lógico falso, pois 13 não é menor ou igual a 5

CÓDIGOS EM C

Exemplo

```
1 int valor_logico_1 = 5.5 == 5.5; // = 1 (Verdadeiro)
2 int valor_logico_2 = 3.7f == 4.1f; // = 0 (Falso)
3 int valor_logico_3 = 12 != 10; // = 1 (Verdadeiro)
4 int valor_logico_4 = 2 < 7; // = 1 (Verdadeiro)
5 int valor_logico_5 = 13 > 11; // = 1 (Verdadeiro)
6 int valor_logico_6 = 4 <= 4; // = 1 (Verdadeiro)
7 int valor_logico_7 = 11 >= -13; // = 1 (Verdadeiro)
```

CÓDIGOS EM C

Operações lógicas

- Além de operações relacionais, operações lógicas também produzem valores lógicos. Porém estas têm como operandos também valores lógicos
- São exemplos de operações lógicas o “e” (`&&`), o “ou” (`||`) e o “não” (`~`)

CÓDIGOS EM C

Operações Lógicas

- $x \&& y$ produz VL (valor lógico) verdadeiro quando x e y são todos verdadeiros e VL falso caso contrário
- Já $!x$ produz VL verdadeiro caso x falso e VL falso caso x verdadeiro

CÓDIGOS EM C

Operações Lógicas

- Por fim $x \text{ || } y$ produz VL verdadeiro quando x , y , ou ambos são verdadeiros. Ou seja, quando pelo menos um dos operandos é verdadeiro. Caso contrário, produz VL falso

CÓDIGOS EM C

Exemplo

```
1 int valor_logico_1 = 1 && 1; // = 1
2 int valor_logico_2 = 0 && 1; // = 0
3 int valor_logico_3 = 0 || 0; // = 0
4 int valor_logico_4 = 1 || 0; // = 1
5 int valor_logico_5 = !0; // = 1
6 int valor_logico_6 = !1; // = 0
```

CÓDIGOS EM C

Funções

- Em C, é possível criar funções. Funções são trechos de códigos que podem realizar diversas ações, porém comumente seu objetivo principal é receber valores de entrada que foram passados e retornar um valor de saída

CÓDIGOS EM C

Funções

- Uma função pode ser usada diversas vezes
- Internamente às funções, estas são semelhantes ao programa principal, por isso ocasionalmente são também chamadas de subprogramas

CÓDIGOS EM C

Funções

- Uma função é declarada por:
 - tipo_retorno
nome(tipo_e_nome_parametro_1,
tipo_e_nome_parametro_2...)
{bloco_de_codigo}
- Para todos os tipos de retorno, exceto void é necessário haver um return acessível dentro da função

CÓDIGOS EM C

Funções

- Uma função é declarada no início do código, após os includes. Alternativamente nesta região do código é possível deixar apenas a assinatura da função (sem o bloco de código) e escrever a função completa ao final do código

CÓDIGOS EM C

Funções

- Uma função é chamada (ou seja acionada) utilizando-se:
 - nome(argumento_1, argumento_2...)
- Para funções que não tem tipo de retorno “void”, o valor retornado por elas é inserido no ponto do código em que foram chamadas

CÓDIGOS EM C

Exemplo

```
1 int soma(int a, int b){  
2     int resultado_da_soma = a + b;  
3     return resultado_da_soma;  
4 }
```

CÓDIGOS EM C

Exemplo

```
1 void hello_world() {
2     printf("Hello, World!");
3 }
```

CÓDIGOS EM C

Exemplo

```
1 double media_harmonica(double a, double b){  
2  
3     double inverso_a = 1.0/a;  
4     double inverso_b = 1.0/b;  
5  
6     double soma_inversos = inverso_a + inverso_b;  
7  
8     double valor_media_harmonica = 2/soma_inversos;  
9     return valor_media_harmonica;  
10 }
```

CÓDIGOS EM C

Exercício

- Faça uma função que recebe a quantidade de dias que já se passaram desde o começo do ano e retorne a quantidade de dias que faltam até o final do ano

CONGRATULATIONS



MUITO OBRIGADO



DIN Departamento de
Informática