

Algoritmos e Lógica de Programação II

Introdução

Prof. MSc. Rafael Staiger Bressan

Introdução

C/C++



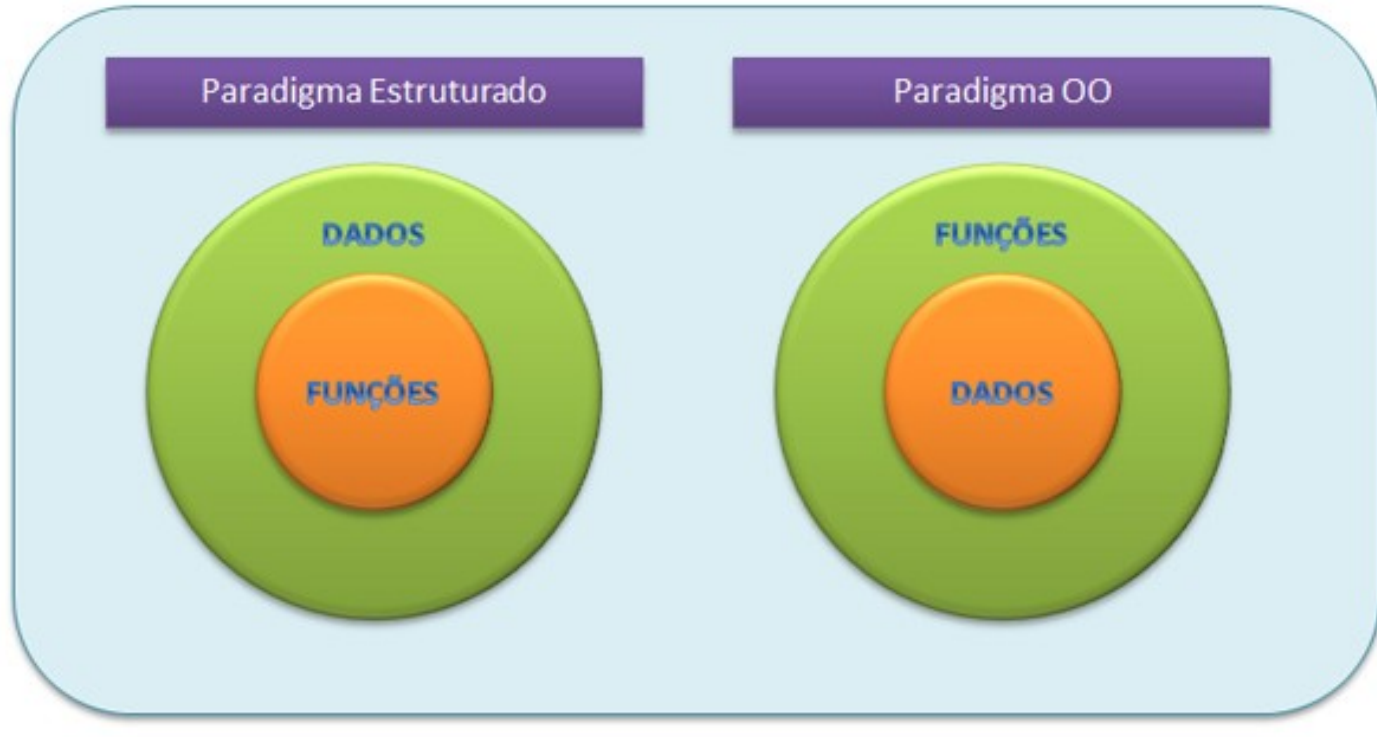
Introdução

Paradigmas de Programação

- Intimamente ligado com a **forma de pensar do programador** e como ele **busca soluções para os problemas**.
 - É o paradigma que permite ou proíbe a utilização de algumas técnicas de programação.
- Paradigma estruturado:
 - Conhecido como *imperativo* ou *procedural*;
 - Qualquer problema pode ser quebrado em problemas menores;
 - Podem ser reduzidos a apenas três estruturas: **sequência, decisão e iteração**
- Paradigma orientado a objeto:
 - Compreende o problema como uma coleção de objetos interagindo por meio de trocas de mensagem.

Introdução

Paradigmas de Programação





Introdução Termologias

- ANSI (American National Standards Institute).
 - ("Instituto Nacional Americano de Padrões");
 - Serve como guia na escrita de **compiladores** e de programas nesta **linguagem de programação**.
- Compiladores.
 - Um compilador é um programa de computador (ou um grupo de programas) que, a partir de um código fonte escrito em uma linguagem compilada, cria um programa semanticamente equivalente.



Introdução

Termologias

- Linguagem de Programação.
 - É um método padronizado para comunicar instruções para um computador;
 - É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.
- Erros de sintaxe.
 - A escrita dos programas contém erros.
- Erros de lógica.
 - A estratégia “gizada” no algoritmo está errada.



Introdução

Termologias

- Linguagem de Programação.
 - É um método padronizado para comunicar instruções para um computador;
 - É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.
- Erros de sintaxe.
 - A escrita dos programas contém erros.
- Erros de lógica.
 - A estratégia “gizada” no algoritmo está errada.



Linguagem C

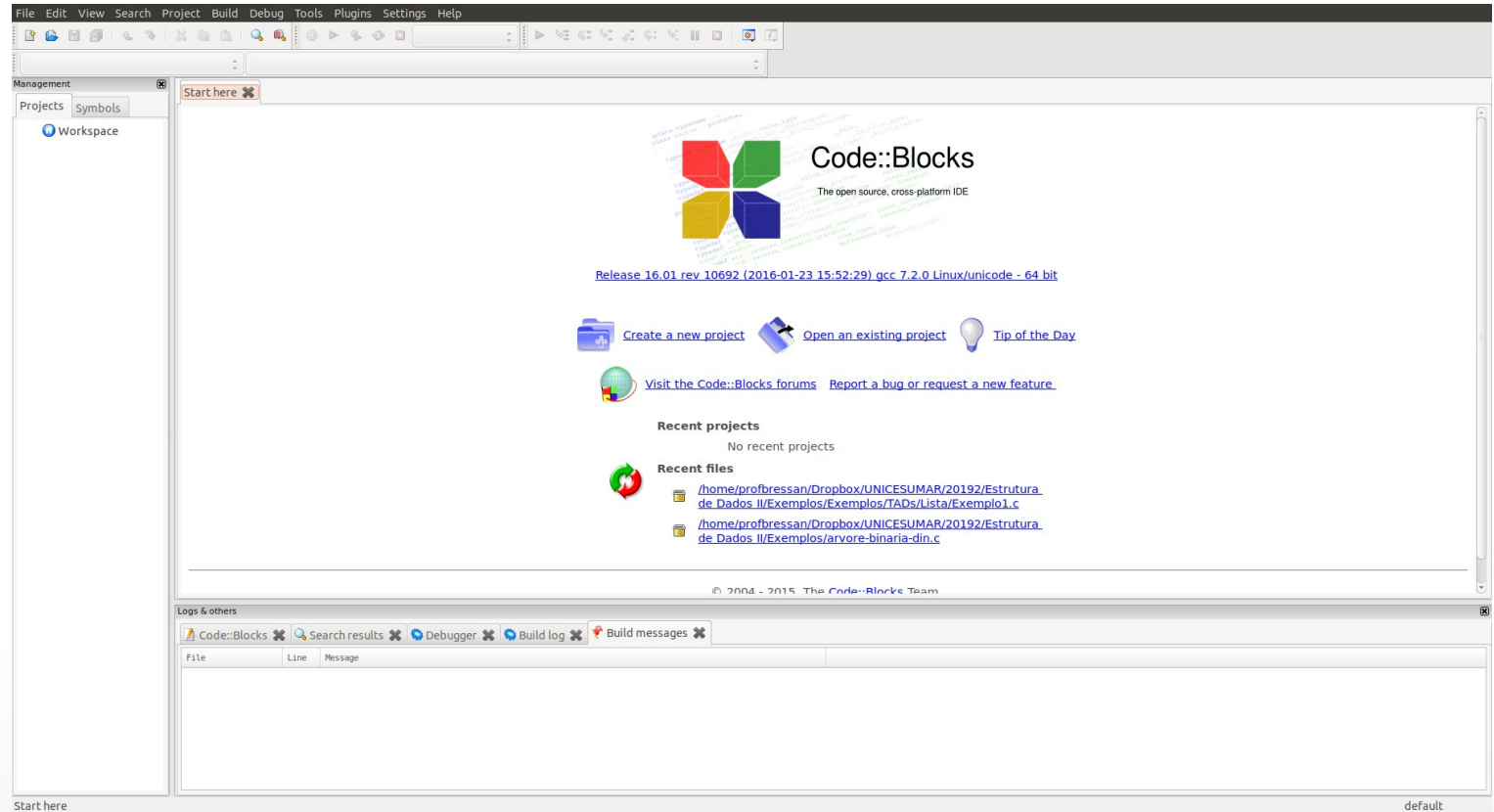
- 1970 (Denis Ritchie),
 - Uso em um computador DEC PDP-11 em Unix
- BCPL \rightarrow B \rightarrow C \rightarrow C++
- C++ é uma extensão da linguagem C que aceita a programação orientada a objetos.
- O sistema Unix é escrito em C e C++



Linguagem C

- Encontra-se na literatura diversos compiladores para linguagem C
 - **Code::Blocks IDE**
 - GCC
 - Dev C++
 - C++ Builder
 - Turbo C
 - ...

Linguagem C



2020

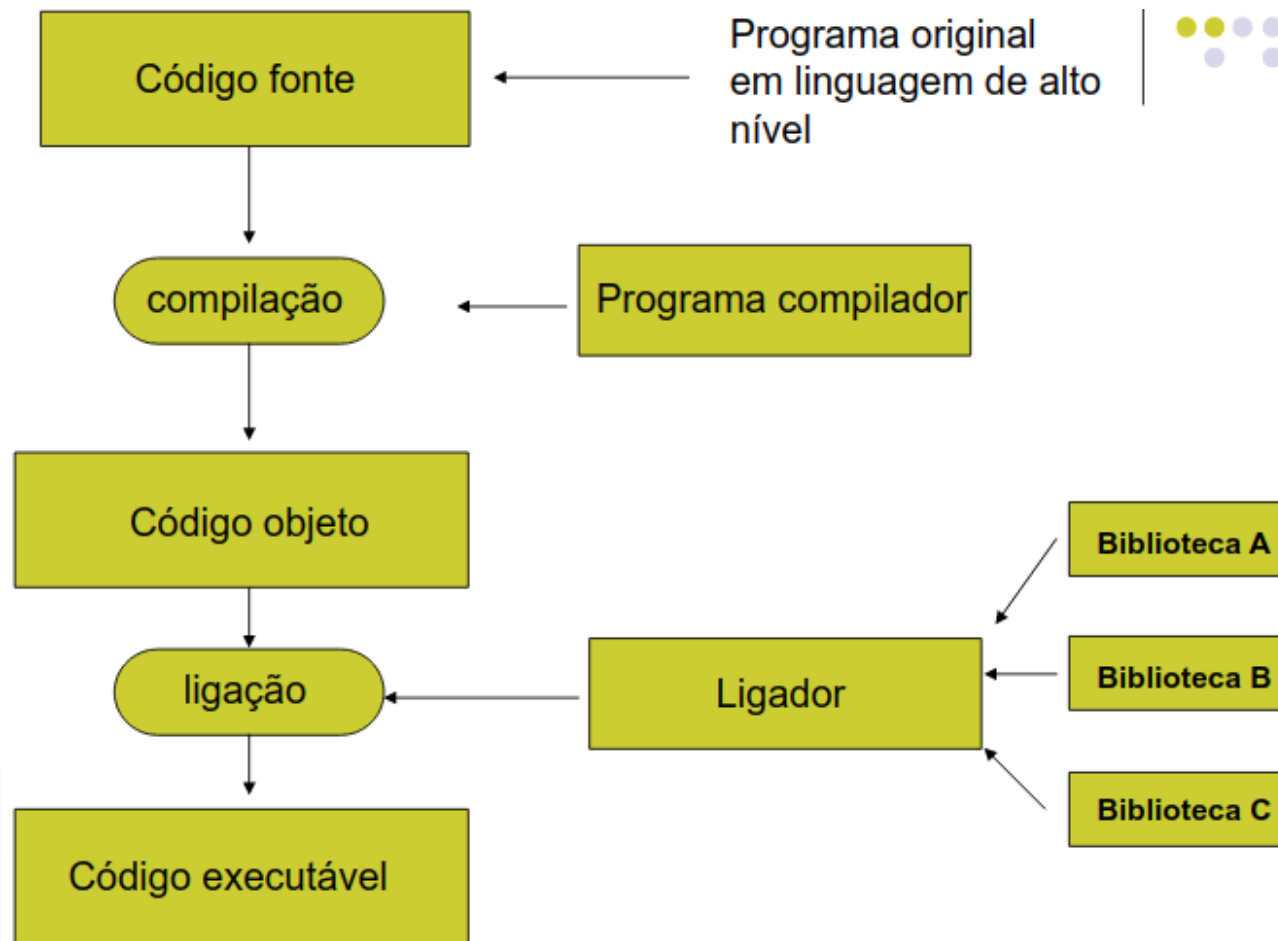
Algoritmos e Lógica de Programação II



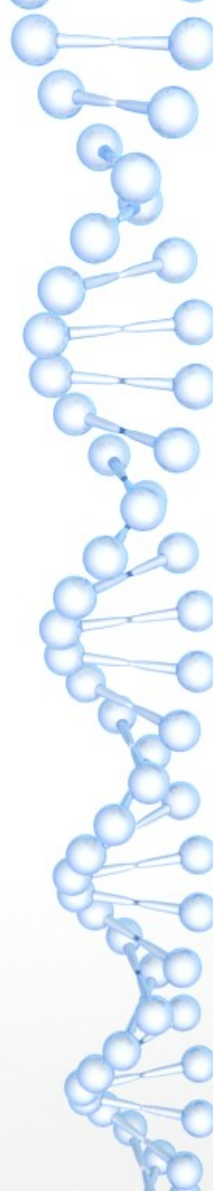
Linguagem C

- Download
 - <http://www.codeblocks.org/downloads/binaries>
- Tutorial para instalação do Code Blocks.
 - <http://linguagemc.com.br/tutorial-para-instalacao-do-code-blocks/>

Conceitos Iniciais de Programação



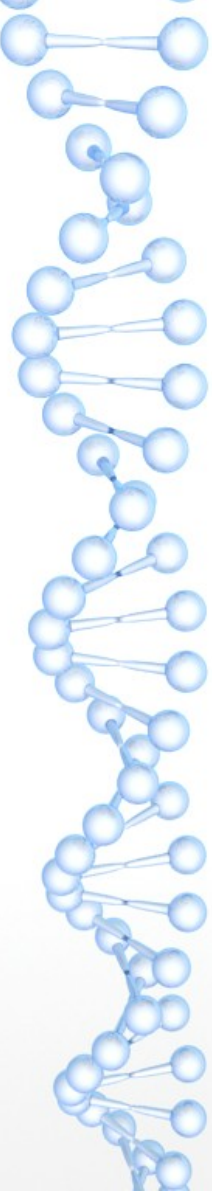
Estrutura de um Programa em C



```
1  ┌─┐ main(){
2      // Conjunto de instruções
3
4      return 0;
5  └─┘ }
```

The diagram illustrates the structure of a C program. It shows a vertical list of line numbers 1 through 5 on the left. To the right of these numbers is a code block. Line 1 contains the opening curly brace of the `main()` function. Line 2 contains a comment `// Conjunto de instruções`. Line 3 is blank. Line 4 contains the `return 0;` statement. Line 5 contains the closing curly brace. A vertical bar is positioned between the line numbers and the code. A bracket on the left side of the code block spans from line 1 to line 5, indicating the entire function body.

Estrutura de um Programa em C



```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4
5  main(){
6      setlocale(LC_ALL, "Portuguese");
7      printf("Alá mundo!");
8
9      return 0;
10 }
```



Identificadores

- Os identificadores são os **nomes das variáveis**, dos programas, das constantes, das rotinas etc. As regras básicas são:
- Os caracteres que você pode utilizar são: os números, as letras maiúsculas, as letras minúsculas e o caractere sublinhado.
- O primeiro caractere deve ser sempre uma letra ou caractere sublinhado.
- Não são permitidos espaços em branco e caracteres especiais (@,\$, +,-,%,!).
- Não podemos usar palavras reservadas nos identificadores, ou seja, palavras que pertençam a uma linguagem de programação.

Tipos de Dados

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Início	Fim
Char	8	%c	-128	127
unsigned char	8	%c	0	255
Signed char	8	%c	-128	127
Int	16	%i ou %d	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i ou %d	-32.768	32.767
Short int	16	%hi ou %hd	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi ou %hd	-32.768	32.767
Long int	32	%li ou %ld	-2.147.483.648	2.147.483.647
signed long int	32	%li ou %ld	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
Float*	32	%f, %e ou %g	3,4E-38	3.4E+38
Double*	64	%lf, %le ou %lg	1,7E-308	1,7E+308
long double*	80	%Lf	3,4E-4932	3,4E+4932



Palavras reservadas

auto
break
case
char
const
continue
default
do
double
else
enum

extern
float
for
goto
if
int
long
register
return
short
signed

sizeof
static
struct
switch
typedef
union
void
volatile
while



Variáveis

- Em C, as variáveis são declaradas **após a especificação de seus tipos.**
- Ex.: <tipo> identificador;
 - int numero;
 - float total;
 - int n1, n2, n3;
 - char sexo;
 - char endereco[30];



Constantes

- Constantes armazenam informações que não variam com o tem. Em C temos a seguinte sintaxe:
 - `#define <itendificador> valor`
 - Não é utilizado o (;) no final.
 - Exemplo:
 - `#define valorPI 3.1416`



Expressões e Operadores

Operação	Operador	Prioridade
Soma	+	1 (menor)
Subtração	-	1
Multiplicação	*	2
Divisão	/	2
Exponenciação	pow(variável, expoente)	3 (maior)
Resto	%	3



Expressões e Operadores

Operação	Operador	Exemplo
Igual	==	A==B ---- B==A
Diferente	!=	A != B
Maior	>	A > 5
Menor	<	A < 5
Maior ou igual a	>=	A >= 5
Menor ou igual a	<=	A <= 5



Expressões e Operadores

Operação	Operador	Exemplo
Conjunção	&&	((A>B) && (B<C))
Disjunção		((A>B) (B<C))
Negação	!	!A

Expressões e Operadores

Operador	Exemplo	Equivalencia
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
++	x++ y = ++x y = x++	x = x + 1 y = x + 1 depois y = x y = x depois x = x + 1
--	x-- y = --x y = x--	x = x - 1 y = x - 1 depois y = x y = x depois x = x - 1



Funções Intrínsecas

- <http://linguagemc.com.br/a-biblioteca-math-h/>
- Função Descrição do comando
 - floor() arredonda para baixo
 - ceil() arredonda para cima
 - sqrt() Calcula raiz quadrada
 - pow(variável, expoente) potenciação
 - sin() seno
 - cos() cosseno
 - tan() Tangente
 - log() logaritmo natural
 - log10() logaritmo base 10



Atribuições

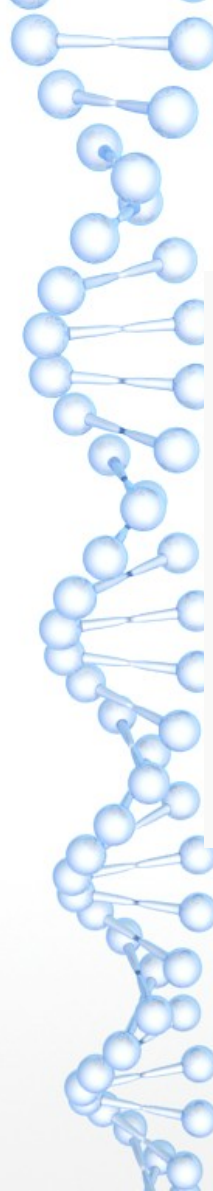
- O comando de atribuição é usado para conceder valores ou operações a variáveis.
 - Identificador = expressão
 - $X = 3$
 - $X = X + 5$
 - $X = 34.78$
 - nome = "Rafael"



Entrada de Dados

- Permite receber os dados digitados pelo usuário armazenando-os em variáveis.
- Existem diversas funções para entrada de dados
 - cin
 - gets
 - scanf

Entrada de Dados



```
1  #include <stdio.h>
2  int main(){
3      int idade;
4      printf("Digite a sua idade: ");
5      scanf("%d", &idade);
6      printf("A idade digitada foi %d \n", idade);
7      return 0;
8  }
9
```

A diagrammatic representation of a C program for reading age. On the left, a vertical line is numbered 1 through 9. A blue arrow points from the line between 5 and 6 to the `scanf` function in the code. A small white box with a horizontal line is positioned to the left of the opening curly brace of the `main` function on line 2.



Entrada de Dados

CÓDIGO	SIGNIFICADO
%c	Leitura de um único caractere.
%d	Leitura de um número decimal inteiro.
%i	Leitura de um decimal inteiro.
%u	Leitura de um decimal sem sinal.
%e	Leitura de um número em ponto flutuante com sinal opcional.
%f	Leitura de um número em ponto flutuante com ponto opcional.
%g	Leitura de um número em ponto flutuante com expoente opcional.
%o	Leitura de um número em base octal.
%s	Leitura de uma string.
%x	Leitura de um número em base hexadecimal.
%p	Leitura de um ponteiro.

Saída de Dados

- A saída de dados permite mostrar dados ao usuário. Em C utilizaremos a função (**printf**).
 - printf("expressão de controle", lista de argumentos);
 - Exemplo:
 - printf("Olá Mundo! ");
 - printf("A idade digitada foi %d \n", idade);

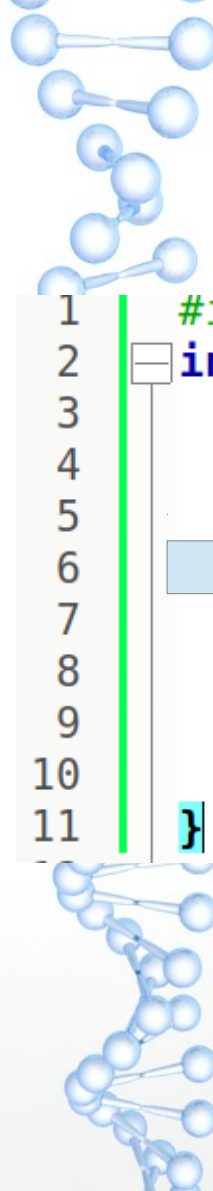





Saída de Dados

- Códigos especiais
 - `\n` → Nova linha
 - `\t` → TAB
 - `\b` → Retrocesso
 - `\"` → Aspas
 - ...

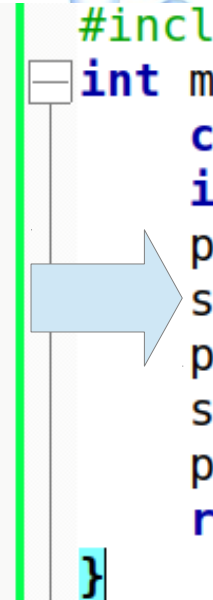
Construindo um Programa




```
1  #include <stdio.h>
2  int main(){
3      char nome[50];
4      int idade;
5      printf("Digite o seu nome: ");
6      scanf("%s", nome);
7      printf("Digite a sua idade: ");
8      scanf("%d", &idade);
9      printf("\n Olá %s. \n A idade digitada foi %d \n\n", nome, idade);
10     return 0;
11 }
```



Construindo um Programa



```
1  #include <stdio.h>
2  int main(){
3      char nome[50];
4      int idade;
5      printf("Digite o seu nome: ");
6      scanf("%[^\n]s", nome);
7      printf("Digite a sua idade: ");
8      scanf("%d", &idade);
9      printf("\n Olá %s. \n A idade digitada foi %d \n\n", nome, idade);
10     return 0;
11 }
```





Atividades

- Faça um programa que leia quatro números, calcule e apresente a média.
- Faça um programa que leia dois números inteiros (num1, num2) calcule e apresente:
 - Soma
 - Subtração
 - Multiplicação
 - Divisão.

```
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <locale.h>
9
10 int main() {
11     setlocale(LC_ALL, "Portuguese");
12     float num1, num2, num3, num4, media;
13     printf("Digite o primeiro número : ");
14     scanf("%f", &num1);
15     printf("Digite o segundo número : ");
16     scanf("%f", &num2);
17     printf("Digite o terceiro número : ");
18     scanf("%f", &num3);
19     printf("Digite o quarto número : ");
20     scanf("%f", &num4);
21     media = (num1 + num2 + num3 + num4) / 4;
22     printf("Média aritmética : %.2f", media);
23     return 0;
24 }
```

```
14  int main() {
15      setlocale(LC_ALL, "Portuguese");
16      int num1, num2, soma, subtracao, multiplicacao;
17      float divisao;
18      printf("Digite o primeiro número : ");
19      scanf("%d", &num1);
20      printf("Digite o segundo número : ");
21      scanf("%d", &num2);
22      soma = num1 + num2;
23      subtracao = num1 - num2;
24      multiplicacao = num1 * num2;
25      divisao = (float) num1 / num2;
26      printf("Soma          : %d \n", soma);
27      printf("Subtração       : %d \n", subtracao);
28      printf("Multiplicação    : %d \n", multiplicacao);
29      printf("Divisão          : %.2f \n", divisao);
30      return 0;
31  }
```



Atividades

- Exercícios Resolvidos e Propostos.
 - “Fundamentos da programação de computadores: algoritmos, Pascal, C/C++ e Java”
 - Livro Base (Algoritmos e Lógica de Programação II)



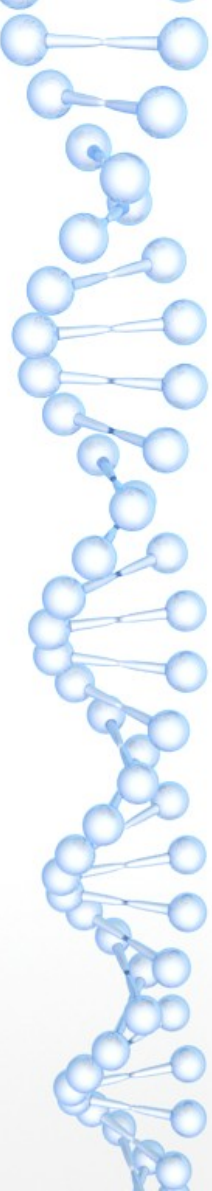
Referencias

MANZANO, José Augusto NG. **Algoritmos lógica para desenvolvimento de programação de computadores**. Saraiva Educação SA, 2010.

ASCENCIO, Ana Fernandes Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores: algoritmos, Pascal e C/C++ e Java**. São Paulo: Pearson Prentice Hall, 2010

CORMEN, Thomas H. et al. **Algoritmos: teoria e prática**. Editora Campus, v. 2, p. 2, 2002.

XAVIER, Gley Fabiano Cardoso. **Lógica de programação**. Senac, 2018.



*“Só existem dois dias no ano que nada pode ser feito.
Um se chama ontem e o outro se chama amanhã”*

Dalai Lama