

Algoritmos e Lógica de Programação

- Douglas Baptista de Godoy

Ementa

- Projeto e representação de algoritmos.
- Estruturas de controle de fluxo de execução: sequência, seleção e repetição.
- Tipos de dados básicos e estruturados (vetores e registros).
- Rotinas. Arquivos.
- Implementação de algoritmos usando uma linguagem de programação.

Objetivo

- Analisar problemas computacionais e projetar soluções por meio da construção de algoritmos.

Avaliação

- Nota1 - Avaliar os conhecimentos adquiridos no 1º bimestre - Nota 1
 - 15/10/2020
- Nota2 - Avaliar os conhecimentos adquiridos no 2º bimestre - Nota 2
 - 10/12/2020
- Recuperação - Substituirá a menor nota do aluno. O aluno só poderá fazer se tirar menos do que 6 em uma das duas provas. - Recuperação abordando todo o conteúdo da disciplina no semestre.
 - 17/12/2020

- Princípios de programação

- Linguagem C/C++
- Segundo Schildt(1996), Dennis Ritchie inventou a linguagem C e foi o primeiro a implementá-la usando um computador DEC PDP-11, que utilizava o sistema operacional Unix.
- A linguagem C++ é uma extensão da linguagem C, e as instruções que fazem parte desta última representam um subconjunto da primeira. Os incrementos encontrados na linguagem C++ foram feitos para dar suporte à programação orientada a objetos, e a sintaxe dessa linguagem é basicamente a mesma da linguagem C.

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

- Paradigmas de programação

- **Paradigmas de Programação**

- Um paradigma de programação esta intimamente relacionado à forma de pensar do programador e como ele busca a solução para os problemas.

- **Paradigma Estruturado**

- O paradigma estruturado, qualquer problema pode ser quebrado em problemas menores, de fácil solução, chamados de sub-rotinas ou funções e ainda, que todo processamento pode ser realizado pelo uso de três tipos de estrutura: sequencial, condicional e iterativa (de repetição)

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

- Paradigmas de programação

- Paradigma Orientado a Objetos
- Paradigma orientado a objetos compreende o problema como uma coleção de objetos interagindo por meio de trocas de mensagem. Os objetos são estruturas de dados contendo estado (dados) e comportamento (logica). Dessa maneira, um conjunto de objetos com informações comuns e com o mesmo comportamento da origem a uma classe

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

- Conceitos de Lógica de Programação e construção de algoritmos

- Algoritmo em pseudocódigo

Estrutura básica:
Este é o mínimo para
todos os Algoritmo
em pseudocódigo

ALGORITMO

DECLARE nome_da_variável tipo_da_variável

bloco_de_comandos

FIM_ALGORITMO.

• Conceitos de Lógica de Programação e construção de algoritmos

- Algoritmo em pseudocódigo
- Declaração de variáveis em algoritmos
- As variáveis são declaradas após a palavra DECLARE e os tipos mais utilizados são: NUMÉRICO (para variáveis que receberão números), LITERAL (para variáveis que receberão caracteres) e LÓGICO (para variáveis que receberão apenas dois valores: verdadeiro ou falso).
- Exemplo:

DECLARE X NUMÉRICO
Y, Z LITERAL
TESTE LÓGICO

• Conceitos de Lógica de Programação e construção de algoritmos

- Algoritmo em pseudocódigo
- Comando de atribuição em algoritmos
- O comando de atribuição é utilizado para conceder valores ou operações a variáveis, sendo representado pelo símbolo \leftarrow . (=)

Exemplo:

$x \leftarrow 4$

$x \leftarrow x + 2$

$y \leftarrow \text{"aula"}$

$\text{teste} \leftarrow \text{falso}$

• Conceitos de Lógica de Programação e construção de algoritmos

- Algoritmo em pseudocódigo
- Comando de entrada em algoritmos
- O comando de entrada é utilizado para receber dados digitados pelo usuário, que serão armazenados em variáveis. Esse comando é representado pela palavra LEIA.

Exemplo:

LEIA X

Um valor digitado pelo usuário será armazenado na variável X.

LEIA Y

Um ou vários caracteres digitados pelo usuário serão armazenados na variável Y.

• Conceitos de Lógica de Programação e construção de algoritmos

- Algoritmo em pseudocódigo
- **Comando de saída em algoritmos**
- O comando de saída é utilizado para mostrar dados na tela ou na impressora. Esse comando é representado pela palavra ESCREVA, e os dados podem ser conteúdos de variáveis ou mensagens.

Exemplo:

ESCREVA X

Mostra o valor armazenado na variável X.

ESCREVA "Conteúdo de Y = ",Y

Mostra a mensagem "Conteúdo de Y = " e, em seguida, o valor armazenado na variável Y.

Algoritmo em pseudocódigo

- Exemplo: Faca um algoritmo para mostrar o resultado da divisão de dois números

ALGORITMO

DECLARE N1, N2, D NUMÉRICO

ESCREVA "Digite dois Números"

LEIA N1, N2

SE N2 = 0 ENTÃO

 ESCREVA "Impossível dividir"

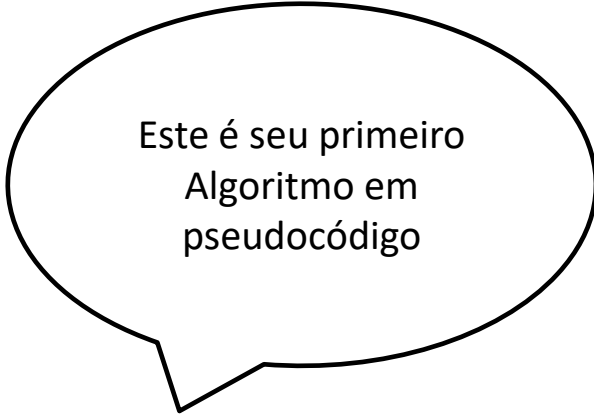
SENÃO INÍCIO

$D = N1/N2$

 ESCREVA "Divisão = ", D

 FIM

FIM_ALGORITMO



Este é seu primeiro
Algoritmo em
pseudocódigo

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Princípios de programação

- Princípios de programação
 - Paradigmas de programação;
 - **Conceitos de usabilidade de sistemas;**
 - Linguagens de programação e códigos fonte, objeto e arquivo executável

Princípios de programação

- Princípios de programação
 - Paradigmas de programação;
 - Conceitos de usabilidade de sistemas;
 - Linguagens de programação e códigos fonte, objeto e arquivo executável

Comandos da linguagem de programação

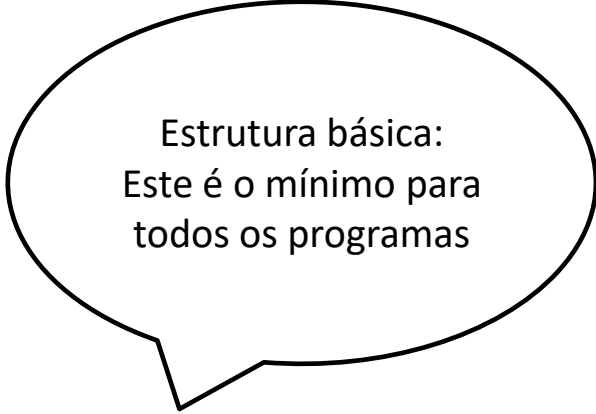
- Funções pré-definidas
- *Expressões e tabela da verdade
- *Tratamento de erros e exceções
- Memória, tipos de dados e variáveis
- Entrada, saída e conversão de tipos
- Operadores aritméticos, relacionais e lógicos.

*Será visto mais a frente

Comandos da linguagem de programação

- Estrutura Sequencial

```
#include<nome_da_biblioteca>
int main()
{
    bloco_comandos;
    return 0;
}
```



Estrutura básica:
Este é o mínimo para
todos os programas

Definição e criação de Variáveis e Constantes

Tipo	Faixa de Valores	Tamanho
char	- 128 a 127	8 bits = 1 bytes
int	-2.147.483,648 a 2.147.483,647	32 bits = 4 bytes
float	3.4×10^{-38} a 3.4×10^{38}	32 bits = 4 bytes
double	1.7×10^{-308} a 1.7×10^{308}	64 bits = 8 bytes

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Definição e criação de Variáveis e Constantes

- Validação de Informações

//C:

//Limites do Tipo de dados INT

```
#include <stdio.h>
#include <limits.h>
const int min_int = INT_MIN;
const int max_int = INT_MAX;
int main()
{
    printf(" %d\n",min_int);
    printf(" %d\n",max_int);
}
```

```
-2147483648
2147483647
```

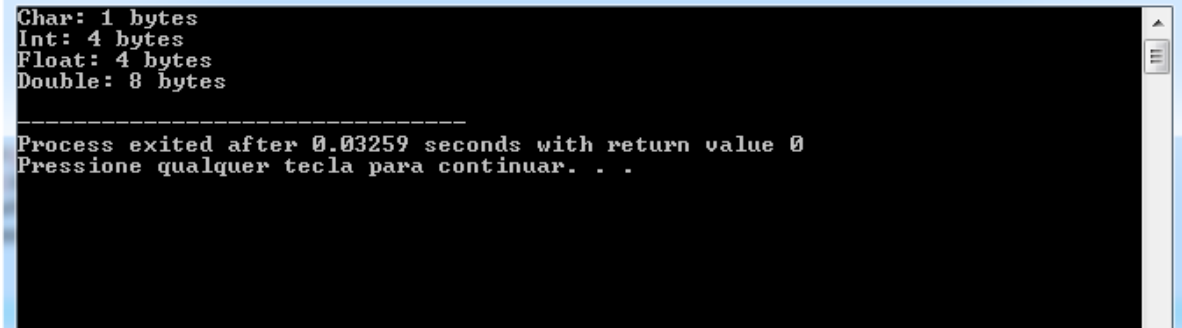
```
Process returned 0 (0x0)   execution time : 0.141 s
Press any key to continue.
```

```
-
```

Definição e criação de Variáveis e Constantes

- Validação de Informações

```
// Faça um programa em C que mostra quantos bytes
//ocupam cada uma das variáveis: char, int, float e double.
#include <stdio.h>
int main(void)
{
    printf("Char: %d bytes\n", sizeof(char));
    printf("Int: %d bytes\n", sizeof(int));
    printf("Float: %d bytes\n", sizeof(float));
    printf("Double: %d bytes\n", sizeof(double));
    return 0;
}
```



```
Char: 1 bytes
Int: 4 bytes
Float: 4 bytes
Double: 8 bytes

-----
Process exited after 0.03259 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Definição e criação de Variáveis e Constantes

- **Declaração de variáveis em C/C++**
- As variáveis são declaradas após a especificação de seus tipos. Os tipos de dados mais utilizados são: int, float e char.
- Exemplo

float x;

float y,z;

char sexo;

```
1  #include <stdio.h>
2  int main ()
3  {
4      int n1, n2, n3, n4, soma;
5      // Mostra mensagem antes da leitura dos quatro números
6      // \n - coloca o cursor na linha de baixo
7      printf("\nDigite quatro números\n");
8      // Recebe os quatro números
9      scanf("%d%c",&n1);
10     scanf("%d%c",&n2);
11     scanf("%d%c",&n3);
12     scanf("%d%c",&n4);
13     // Soma os números digitados
14     soma = n1 + n2 + n3 + n4;
15     // Mostra mensagem e o resultado da soma
16     printf("\nResultado da soma = %d\n",soma);
17     // Pára o programa a espera de um ENTER
18     getch();
19     return 0;
20 }
```

Definição e criação de Variáveis e Constantes

- **Declaração de constantes em C/C++**
- As constantes são declaradas depois das bibliotecas e seus valores não podem ser alterados durante a execução do programa. A declaração deve obedecer à seguinte sintaxe: “ #define nome valor ”
- Exemplo
- #define x 7
- #define y 4.5
- #define nome “MARIA”

```
1  #include <stdio.h>
2  #define x 7
3  int main ()
4  {
5      int n1, n2, n3, n4, soma;
6      // Mostra mensagem antes da leitura dos quatro números
7      // \n - coloca o cursor na linha de baixo
8      printf("\nDigite quatro números\n");
9      // Recebe os quatro números
10     scanf("%d%c",&n1);
11     scanf("%d%c",&n2);
12     scanf("%d%c",&n3);
13     scanf("%d%c",&n4);
14     // Soma os números digitados
15     soma = n1 + n2 + n3 + n4;
16     // Mostra mensagem e o resultado da soma
17     printf("\nResultado da soma = %d\n",soma);
18     printf("valor de x %d",x);
19     // Pára o programa a espera de um ENTER
20     getchar();
21     return 0;
22 }
```

Comandos de Entrada, Processamento e Saída

- **Comando de entrada em C/C++**
- O comando de entrada é utilizado para receber dados digitados pelo usuário. Os dados recebidos são armazenados em variáveis.
- Exemplo
 - `scanf("%d%c",&x);`
 - `scanf("%f%c",&z);`
 - `scanf("%c%c",&sexo);`

```
1  #include <stdio.h>
2  int main ()
3  {
4      int n1, n2, n3, n4, soma;
5      // Mostra mensagem antes da leitura dos quatro números
6      // \n - coloca o cursor na linha de baixo
7      printf("\nDigite quatro números\n");
8      // Recebe os quatro números
9      scanf("%d%c",&n1);
10     scanf("%d%c",&n2);
11     scanf("%d%c",&n3);
12     scanf("%d%c",&n4);
13     // Soma os números digitados
14     soma = n1 + n2 + n3 + n4;
15     // Mostra mensagem e o resultado da soma
16     printf("\nResultado da soma = %d\n",soma);
17     // Para o programa a espera de um ENTER
18     getchar();
19     return 0;
20 }
```

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Comandos de Entrada, Processamento e Saída

- **Comando de saída em C/C++**

- O comando de saída é utilizado para mostrar dados na tela ou na impressora.

- Exemplo

- `printf("%f",y);`
- `printf("Conteudo de Y = %f",y);`
- `printf("Aula");`
- `printf("\nFácil");`

```
1  #include <stdio.h>
2  int main ()
3  {
4      int n1, n2, n3, n4, soma;
5      // Mostra mensagem antes da leitura dos quatro números
6      // \n - coloca o cursor na linha de baixo
7      printf("\nDigite quatro números\n");
8      // Recebe os quatro números
9      scanf("%d%c",&n1);
10     scanf("%d%c",&n2);
11     scanf("%d%c",&n3);
12     scanf("%d%c",&n4);
13     // Soma os números digitados
14     soma = n1 + n2 + n3 + n4;
15     // Mostra mensagem e o resultado da soma
16     printf("\nResultado da soma = %d\n",soma);
17     // Pára o programa a espera de um ENTER
18     getchar();
19     return 0;
20 }
```

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Comandos de Entrada, Processamento e Saída

- **Comando de atribuição em C/C++**

- O comando de atribuição é utilizado para conceder valores ou operações a variáveis, sendo representado por = (sinal de igualdade).

- Exemplo

- $x = 4;$
- $x = x + 2;$
- $\text{sexo} = \text{'F'};$

```
1  #include <stdio.h>
2  int main ()
3  {
4      int n1, n2, n3, n4, soma;
5      // Mostra mensagem antes da leitura dos quatro números
6      // \n - coloca o cursor na linha de baixo
7      printf("\nDigite quatro números\n");
8      // Recebe os quatro números
9      scanf("%d%c",&n1);
10     scanf("%d%c",&n2);
11     scanf("%d%c",&n3);
12     scanf("%d%c",&n4);
13     // Soma os números digitados
14     soma = n1 + n2 + n3 + n4;
15     // Mostra mensagem e o resultado da soma
16     printf("\nResultado da soma = %d\n",soma);
17     // Pára o programa a espera de um ENTER
18     getchar();
19     return 0;
20 }
```

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Comandos de Entrada, Processamento e Saída

- **Comentários em C/C++**

- Comentários são textos que podem ser inseridos em programas com o objetivo de documentá-los. Eles não são analisados pelo compilador.
- Os comentários podem ocupar uma ou várias linhas, devendo ser inseridos nos programas utilizando-se os símbolos `/* */` ou `//`.

- Exemplo

- `//` comentário de uma linha
- `/*` comentário de múltiplas linhas `*/`
- Ctrl + Shift + c
- Ctrl + Shift + x

```
1  #include <stdio.h>
2  int main ()
3  {
4      int n1, n2, n3, n4, soma;
5      // Mostra mensagem antes da leitura dos quatro números
6      // \n - coloca o cursor na linha de baixo
7      printf("\nDigite quatro números\n");
8      // Recebe os quatro números
9      scanf("%d%c",&n1);
10     scanf("%d%c",&n2);
11     scanf("%d%c",&n3);
12     scanf("%d%c",&n4);
13     // Soma os números digitados
14     soma = n1 + n2 + n3 + n4;
15     // Mostra mensagem e o resultado da soma
16     printf("\nResultado da soma = %d\n",soma);
17     // Pára o programa a espera de um ENTER
18     getchar();
19     return 0;
20 }
```

Operadores Aritméticos

Operador	Exemplo	Comentário
=	$x = y$	O conteúdo da variável Y é atribuído à variável X (A uma variável pode ser atribuído o conteúdo de outra, um valor constante ou, ainda, o resultado de uma função).
+	$x + y$	Soma o conteúdo de X e de Y.
-	$x - y$	Subtrai o conteúdo de Y do conteúdo de X.
*	$x * y$	Multiplica o conteúdo de X pelo conteúdo de Y.
/	x / y	Obtém o quociente da divisão de X por Y. Se os operandos são inteiros, o resultado da operação será o quociente inteiro da divisão. Se os operadores são reais, o resultado da operação será a divisão. Por exemplo: $\text{int } z = 5/2; \rightarrow$ a variável z receberá o valor 2. $\text{float } z = 5.0/2.0; \rightarrow$ a variável z receberá o valor 2.5.
%	$x \% y$	Obtém o resto da divisão de X por Y.

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Operadores Aritméticos e Expressões Aritméticas

Operador	Exemplo	Comentário
$+=$	$x += y$	Equivale a $X = X + Y$.
$- =$	$x -= y$	Equivale a $X = X - Y$.
$* =$	$x *= y$	Equivale a $X = X * Y$.
$/ =$	$x /= y$	Equivale a $X = X / Y$.
$\% =$	$x \% = y$	Equivale a $X = X \% Y$.
$++$	$x ++$	Equivale a $X = X + 1$.
$++$	$y = ++ x$	Equivale a $X = X + 1$ e depois $Y = X$.
$++$	$y = x ++$	Equivale a $Y = X$ e depois $X = X + 1$.
$--$	$x --$	Equivale a $X = X - 1$.
$--$	$y = -- x$	Equivale a $X = X - 1$ e depois $Y = X$.
$--$	$y = x --$	Equivale a $Y = X$ e depois $X = X - 1$.

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Operadores Aritméticos e Expressões Aritméticas

- Faça um programa que receba três notas e seus respectivos pesos, calcule e mostre a media ponderada.

```
1  #include <stdio.h>
2  int main()
3  {
4      float nota1, nota2, nota3, peso1, peso2, peso3, media;
5      printf("Digite as três notas e seus pesos ");
6      // Recebe as três notas e seus pesos
7      scanf("%f%c",&nota1);
8      scanf("%f%c",&nota2);
9      scanf("%f%c",&nota3);
10     scanf("%f%c",&peso1);
11     scanf("%f%c",&peso2);
12     scanf("%f%c",&peso3);
13     // Calcula a média
14     media = (nota1 * peso1 + nota2 * peso2 + nota3 * peso3)/(peso1 + peso2 + peso3);
15     // Mostra o resultado da média
16     // Formatando a saída para mostrar no mínimo 3 caracteres
17     // e destes, 2 caracteres para a parte decimal
18     printf("%.3f\n",media);
19     // Pára o programa a espera de uma tecla
20     getch();
21     return 0;
22 }
```

Operadores Relacionais

Operador	Exemplo	Comentário
<code>==</code>	<code>x == y</code>	O conteúdo de X é igual ao conteúdo de Y.
<code>!=</code>	<code>x != y</code>	O conteúdo de X é diferente do conteúdo de Y.
<code><=</code>	<code>x <= y</code>	O conteúdo de X é menor ou igual ao conteúdo de Y.
<code>>=</code>	<code>x >= y</code>	O conteúdo de X é maior ou igual ao conteúdo de Y.
<code><</code>	<code>x < y</code>	O conteúdo de X é menor que o conteúdo de Y.
<code>></code>	<code>x > y</code>	O conteúdo de X é maior que o conteúdo de Y.

Fonte: Fundamentos da Programação de Computadores, Pearson Editora, 3ª edição

Funções pré-definidas

Funções Matemáticas - biblioteca math.h

Função	Exemplo	Comentário
ceil	ceil (X)	Arredonda um numero real para cima. Por exemplo, ceil (3.2) é 4.
cos	cos (X)	Calcula o cosseno de X (X deve estar representado em radianos).
exp	exp (X)	Obtém o logaritmo natural e elevado à potência X.
abs	abs (X)	Obtém o valor absoluto de X.
floor	floor (X)	Arredonda um número real para baixo. Por exemplo, floor (3.2) é 3.
log	log (X)	Obtém o logaritmo natural de X.
log10	log10 (X)	Obtém o logaritmo de base 10 de X.
modf	z = modf (X, & Y)	Decompõe o número real armazenado em X em duas partes: Y recebe a parte fracionária e z, a parte inteira do número.
pow	pow (X, Y)	Calcula a potência de X elevado a Y.
sin	sin (X)	Calcula o seno de X (X deve estar representado em radianos).
sqrt	sqrt (X)	Calcula a raiz quadrada de X.
tan	tan (X)	Calcula a tangente de X (X deve estar representado em radianos).

Comandos da linguagem de programação

- Estrutura Sequencial em C/C++
- Exemplo: Faça um programa que calcule e mostre a área de um triângulo.

```
1  #include <stdio.h>
2  int main()
3  {
4  float base, altura, area;
5  // Mostra mensagem antes da leitura da base
6  printf("\nDigite a base do triângulo\n");
7  // Recebe a base
8  scanf("%f%c",&base);
9  // Mostra mensagem antes da leitura da altura
10 printf("\nDigite a altura do triângulo\n");
11 // Recebe a altura
12 scanf("%f%c",&altura);
13 // Calcula a área
14 area = (base * altura)/2;
15 // Mostra a área
16 printf("\nA área do triângulo é %4.2f",area);
17 // Pára o programa a espera de um ENTER
18 getchar();
19 return 0;
20 }
```


Funções pré-definidas

- Faça um programa que calcule e mostre a área de um círculo. Sabe-se que: $\text{Área} = \pi * R^2$

```
1 //Faça um programa que calcule e mostre a área de um
2 //círculo. Sabe-se que: Área = pi * R 2
3 #include <stdio.h>
4 #include <math.h>
5 int main()
6 { float area, raio;
7   // Mostra mensagem antes da leitura do raio
8   printf("\nDigite o raio: ");
9   // Recebe o raio
10  scanf("%f%c",&raio);
11  // Calcula a área
12  area = 3.1415 * pow(raio,2);
13  // Mostra a área
14  printf("\nA área é: %4.3f",area);
15  // Pára o programa a espera de um ENTER
16  getchar();
17  return 0;
18 }
```

Referencias Bibliográficas

- ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de, **Fundamentos da Programação de Computadores**, Pearson Editora, 3ª edição.