

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

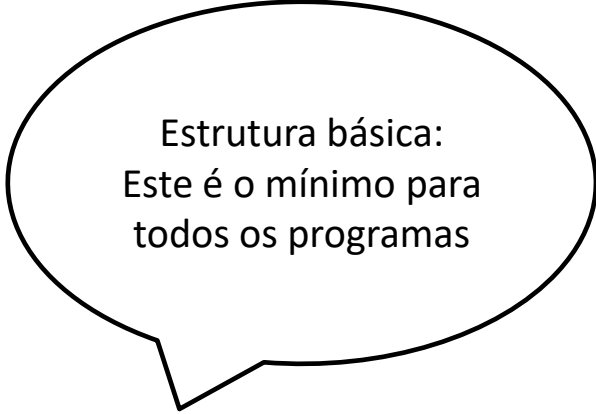
- Linguagem C/C++
- Segundo Schildt(1996), Dennis Ritchie inventou a linguagem C e foi o primeiro a implementá-la usando um computador DEC PDP-11, que utilizava o sistema operacional Unix.
- A linguagem C++ é uma extensão da linguagem C, e as instruções que fazem parte desta última representam um subconjunto 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

Lógica de Programação conceitos básicos

- 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

- **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     getch();
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 }
```

- Comandos da linguagem de programação
- 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.

• Comandos da linguagem de programação

- 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

• Comandos da linguagem de programação

- 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:

$x \leftarrow 4$

$x \leftarrow x + 2$

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

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

• Comandos da linguagem de programação

- 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.

- Comandos da linguagem de programação

- 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.

• Programação estruturada

- **Estrutura condicional em algoritmos** - A estrutura condicional em algoritmos pode ser simples ou composta.

Exemplo:

Estrutura condicional simples

SE condição *ENTÃO*
comando

- O comando só será executado se a condição for verdadeira. Uma condição é uma comparação que possui dois valores possíveis: verdadeiro ou falso.

Exemplo:

SE condição *ENTÃO*
INÍCIO

comando1
comando2
comando3

FIM

- Os comandos 1, 2 e 3 só serão executados se a condição for verdadeira. As palavras INÍCIO e FIM serão necessárias apenas quando dois ou mais comandos forem executados.

• Programação estruturada

- Estrutura condicional composta

- Exemplo1:

```
SE condição ENTÃO
    comando1
SENÃO
    comando2
```

Se a condição for verdadeira, será executado o comando1; caso contrário, será executado o comando2.

Exemplo2:

```
SE condição
    ENTÃO INÍCIO
        comando1
        comando2
    FIM
SENÃO INÍCIO
    comando3
    comando4
FIM
```

Se a condição for verdadeira, o comando1 e o comando2 serão executados; caso contrário, o comando3 e o comando4 serão executados.

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"

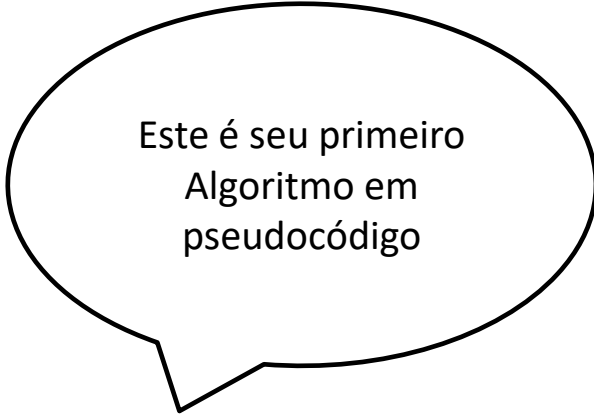
SENÃO INÍCIO

$D = N1/N2$

 ESCREVA "Divisão = ", D

 FIM

FIM_ALGORITMO



Este é seu primeiro
Algoritmo em
pseudocódigo

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

Operadores Aritméticos e Expressões Aritméticas

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: <code>int z = 5/2;</code> → a variável z receberá o valor 2. <code>float z = 5.0/2.0;</code> → 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

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).

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

Estruturas de Controle: Sequencial; Condicional; Repetição

- Estrutura condicional em C/C++
- A estrutura condicional é apresentada em três maneiras: simples, composta e case.

Exemplo:

Estrutura condicional simples

```
if(condição)
    comando;
```

Exemplo:

Estrutura condicional simples

```
if(condição)
{
    comando1;
    comando2;
    comando3;
}
```

Estruturas de Controle: Sequencial; Condicional; Repetição

- Estrutura condicional simples
- Exemplo: Faça um programa que receba um numero inteiro e verifique se este numero é maior que zero.

```
1  #include <stdio.h>
2  int main()
3  {
4      int x;
5      printf("Digite um numero inteiro: ");
6      scanf("%d%c", &x);
7      if (x>0)
8          printf("O numero maior que zero");
9      getchar();
10     return 0;
11 }
```

Estruturas de Controle: Sequencial; Condicional; Repetição

- Estrutura condicional em C/C++

Exemplo:

Estrutura condicional composta

```
if(condição)
    comando1;
else
    comando2;
```

Exemplo:

Estrutura condicional composta

```
if(condição)
{
    comando1;
    comando2;
}
else
{
    comando3;
    comando4;
}
```

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

Estruturas de Controle: Sequencial; Condicional; Repetição

- Estrutura condicional composta
- Exemplo: Faça um programa que receba um numero inteiro e verifique se é par ou ímpar.

```
1  #include <stdio.h>
2  int main()
3  {
4      int num;
5      //Mostra mensagem solicitando um número
6      printf( "\nDigite um número: ");
7      //Recebe o número
8      scanf("%d%c",&num);
9      if (num % 2 == 0)
10         printf( "\nO número , par");
11     else
12         printf( "\nO número , ímpar");
13     return 0;
14     getchar();
15 }
```

Estruturas de Controle: Sequencial; Condicional; Repetição

- Estrutura condicional em C/C++ e **Estrutura case**
- Em alguns programas, existem situações mutuamente exclusivas, isto é , se uma situação for executada, as demais não serão.

Exemplo:

```
switch(variável)
{
    case valor1:
        lista_de_comandos;
        break;
    case valor2:
        lista_de_comandos;
        break;
    default:
        lista_de_comandos;
        break;
```

```
15 switch(op)
16 {
17     case 1:
18         printf("\nDigite um valor para o primeiro número: ");
19         scanf("%f%c",&num1);
20         printf("\nDigite um valor para o segundo número: ");
21         scanf("%f%c",&num2);
22         soma = num1 + num2;
23         printf("\nA soma de %f e %f = %f",num1,num2,soma);
24         break;
25     case 2:
26         printf("\nDigite um valor: ");
27         scanf("%f%c",&num1);
28         raiz = sqrt(num1);
29         printf("\nA raiz quadrada de %f = %f",num1,raiz);
30         break;
31     default:
32         printf("\nOpção inválida !");
33 }
```

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

Estruturas de Controle: Sequencial; Condicional; Repetição

Exemplo:
Estrutura case

```
1  #include<stdio.h>
2  #include <math.h>
3  int main()
4  {
5      float num1,num2,soma, raiz;
6      int op;
7      //Mostra um menu de opcoes
8      printf("\n1- Somar dois nEmeros");
9      printf("\n2- Raiz quadrada de um nEmero");
10     //Mostra mensagem solicitando a opcao do usuario
11     printf("\nDigite sua opcao:");
12     //Recebe a opcao do usu rio
13     scanf("%d%c",&op);
14     //Avalia o valor da vari vel op para decidir qual CASE ser executado
15     switch(op)
16     {
17         case 1:
18             printf("\nDigite um valor para o primeiro nEmero: ");
19             scanf("%f%c",&num1);
20             printf("\nDigite um valor para o segundo nEmero: ");
21             scanf("%f%c",&num2);
22             soma = num1 + num2;
23             printf("\nA soma de %f e %f = %f", num1,num2,soma);
24             break;
25         case 2:
26             printf("\nDigite um valor: ");
27             scanf("%f%c",&num1);
28             raiz = sqrt(num1);
29             printf("\nA raiz quadrada de %f = %f", num1,raiz);
30             break;
31         default:
32             printf("\nOpcao invalida !");
33     }
34     //Para o programa a espera de um ENTER
35     getchar();
36 }
```

Operadores Relacionais

- Exemplos
- if(x == 3)
- printf("Número igual a 3");
- if (num1 > num2)
- printf("\nO maior numero : %f",num1);
- if (x > 0)
- printf("O numero digitado e positivo");

```
1  #include <stdio.h>
2  int main()
3  {
4      float num1, num2;
5      //Mostra mensagem solicitando o primeiro nEmero
6      printf("\nDigite o primeiro numero: ");
7      //Recebe o valor do primeiro nEmero
8      scanf("%f%c",&num1);
9      //Mostra mensagem solicitando o segundo nEmero
10     printf("\nDigite o segundo numero: ");
11     //Recebe o valor do segundo nEmero
12     scanf("%f%c",&num2);
13     //Determina e mostra o maior nEmero
14     if (num1 > num2)
15         printf("\nO maior numero : %f",num1);
16     if (num2 > num1)
17         printf("\nO maior numero : %f",num2);
18     if (num1 == num2)
19         printf("\nOs numeros sao iguais");
20     //Para o programa a espera de um ENTER
21     getchar();
22 }
```


Operadores Lógicos e Expressões Lógicas

TABELA E (&&)	TABELA OU ()	TABELA NÃO (!)
$V \text{ e } V = V$	$V \text{ ou } V = V$	Não $V = F$
$V \text{ e } F = F$	$V \text{ ou } F = V$	Não $F = V$
$F \text{ e } V = F$	$F \text{ ou } V = V$	
$F \text{ e } F = F$	$F \text{ ou } F = F$	

Operadores Lógicos e Expressões Lógicas

p	q	p && q	p q
falso	falso	falso	falso
falso	verdadeiro	falso	verdadeiro
verdadeiro	falso	falso	verdadeiro
verdadeiro	verdadeiro	verdadeiro	verdadeiro

Exemplo 1: $72 > 30$ && $32 \leq 10$
Verdadeiro && Falso
Falso

Exemplo 2: $9 == 3$ || $10 \leq 10$
Falso || Verdadeiro
Verdadeiro

Operadores Lógicos e Expressões Lógicas

- Exemplos
- `if (x > 5 && x < 10)`
- `printf("\n Numero entre 5 e 10 ");`
- `if ((x == 5 && y == 2) || (y == 3))`
- `printf(" x é igual a 5 e y é igual a 2, ou y é igual a 3 ");`
- `if (x == 5 && (y == 2 || y == 3))`
- `printf(" x é igual a 5, e y é igual a 2 ou y é igual a 3 ");`

Referencias Bibliográficas

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