



Estruturas de seleção e repetição






Prof. Lilian Berton

São José dos Campos, 2019

Divisão de inteiros e reais

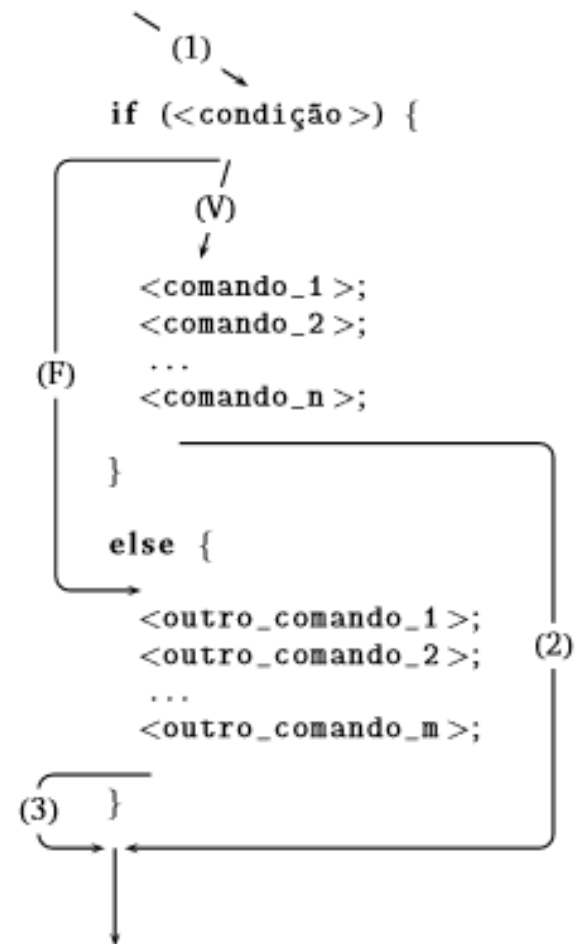
```
int i,j;  
float y;
```

```
i = 5 / 3; /* divisão inteira e o resultado é 1 (5 e 3 são inteiros) */  
y = 5 / 3; /* divisão inteira e o resultado é 1.0 (y é real) */  
y = 5.0 / 2; /* divisão tem como resultado 2.5 (o numerador é real) */  
y = 5 / 2.0; /* divisão tem como resultado 2.5 (o denominador é real) */
```

- Se as variáveis envolvidas na operação forem inteiras  O resultado é inteiro
- Se as variáveis envolvidas na operação forem reais  O resultado é real
- Se as variáveis envolvidas na operação forem inteiras  O resultado é real e reais.
- Se a variável que recebe o resultado for inteira  O valor armazenado é inteiro
- Se a variável que recebe o resultado for real  O valor armazenado é real

Comando if-else

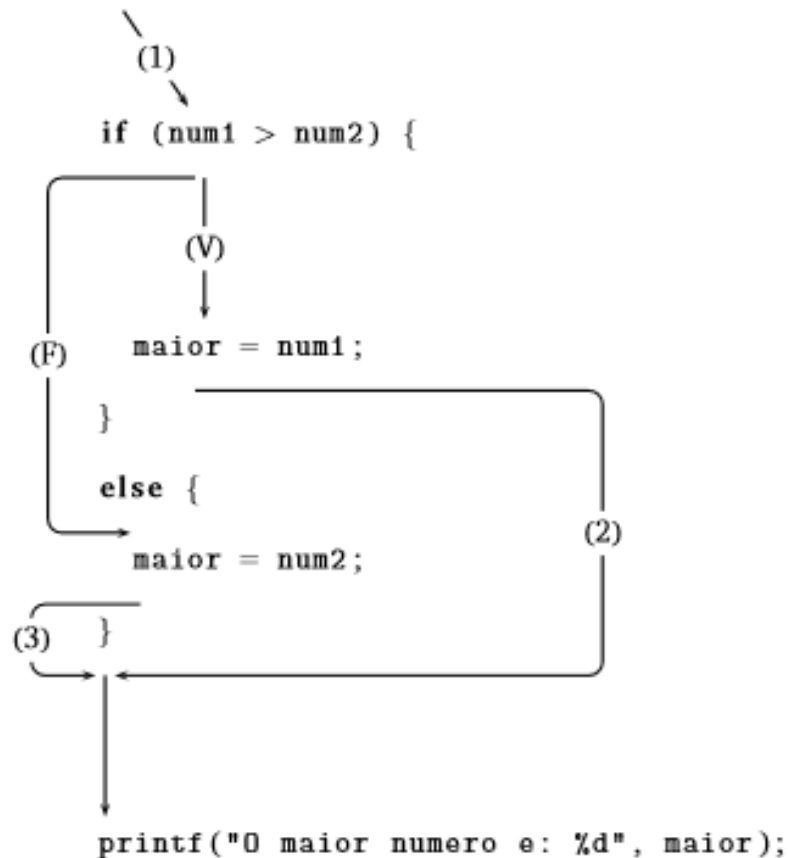
- Primeiramente, a execução do programa vem e **testa a <condição> do if** (seta marcada com (1)).
- **Se a <condição> é verdadeira, o fluxo do programa segue a seta marcada com (V) e executa a sequência de comandos dentro do if** e ignora a sequência de comandos dentro do else seguindo a seta marcada com (2) a instrução seguinte do comando if-else é executada.
- **Se a <condição> do if é falsa, o fluxo do programa ignora a sequência de comandos dentro do if e segue a seta marcada com (F) e executa a sequência de comandos dentro do else.**
- No final o fluxo segue a seta marcada com (3) executando a instrução seguinte ao comando if-else.



Exemplo 1 comando if-else

- Faça um programa que leia dois números e indique o maior deles.

```
printf("Entre com 2 numeros inteiros");  
scanf("%d %d", &num1, &num2);
```



Exemplo 2 comando if-else

```
#include <stdio.h>
```

```
int main (void) {  
    int N1, N2 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
    printf("Digite o segundo numero: ");  
    scanf("%d", &N2);  
  
    if (N1 == N2)  
        printf("Os numeros são iguais!");  
    else if (N1 > N2)  
        printf("O maior valor e = %d", N1);  
    else  
        printf("O maior valor e = %d", N2);  
  
    return (0);  
}
```

```
#include <stdio.h>
```

```
int main (void) {  
    int N1, N2 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
    printf("Digite o segundo numero: ");  
    scanf("%d", &N2);  
  
    if (N1 == N2)  
        printf("Os numeros são iguais!");  
    if (N1 > N2)  
        printf("O maior valor e = %d", N1);  
    if (N1 < N2)  
        printf("O maior valor e = %d", N2);  
  
    return (0);  
}
```

Exemplo 2 comando if-else

```
#include <stdio.h>
```

```
int main (void) {  
    int N1, N2 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
    printf("Digite o segundo numero: ");  
    scanf("%d", &N2);  
  
    if (N1 == N2)  
        printf("Os numeros são iguais!");  
    else if (N1 > N2)  
        printf("O maior valor e = %d", N1);  
    else  
        printf("O maior valor e = %d", N2);  
  
    return (0);  
}
```

Apenas uma das condições pode ser Verdadeira!

```
#include <stdio.h>
```

```
int main (void) {  
    int N1, N2 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
    printf("Digite o segundo numero: ");  
    scanf("%d", &N2);  
  
    if (N1 == N2)  
        printf("Os numeros são iguais!");  
    if (N1 > N2)  
        printf("O maior valor e = %d", N1);  
    if (N1 < N2)  
        printf("O maior valor e = %d", N2);  
  
    return (0);  
}
```

Não tem
diferença na
saída

Todas as condições podem ser verdadeiras!

Exemplo 3 comando if-else

```
#include <stdio.h>
```

```
int main (void) {  
    int N1 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
  
    if (N1 > 0)  
        printf("O numero eh positivo!");  
    else if (N1 < 0)  
        printf("O numero eh negativo!");  
    else if (N1 % 2 == 0)  
        printf("O numero eh par! ");  
    else  
        printf("O numero eh impar");  
  
    return (0);  
}
```

```
#include <stdio.h>
```

```
int main (void) {  
    int N1 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
  
    if (N1 > 0)  
        printf("O numero eh positivo!");  
    if (N1 < 0)  
        printf("O numero eh negativo!");  
    if (N1 % 2 == 0)  
        printf("O numero eh par! ");  
    else  
        printf("O numero eh impar");  
  
    return (0);  
}
```

Exemplo 3 comando if-else

```
#include <stdio.h>
```

```
int main (void) {  
    int N1 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
  
    if (N1 > 0)  
        printf("O numero eh positivo!");  
    else if (N1 < 0)  
        printf("O numero eh negativo!");  
    else if (N1 % 2 == 0)  
        printf("O numero eh par! ");  
    else  
        printf("O numero eh impar");  
  
    return (0);  
}
```

```
Digite o primeiro numero: 2  
O numero eh positivo!
```

```
#include <stdio.h>
```

```
int main (void) {  
    int N1 ;  
    printf("Digite o primeiro numero: ");  
    scanf("%d", &N1);  
  
    if (N1 > 0)  
        printf("O numero eh positivo!");  
    if (N1 < 0)  
        printf("O numero eh negativo!");  
    if (N1 % 2 == 0)  
        printf("O numero eh par! ");  
    else  
        printf("O numero eh impar");  
  
    return (0);  
}
```

Tem diferença
na saída

```
Digite o primeiro numero: 2  
O numero eh positivo  
O numero eh par
```


Switch case em C

- É uma forma de reduzir a complexidade de vários if ... else encadeados.
- É muito utilizado, principalmente para uso em estruturas de menu.
- O uso do **break** evita testar as demais alternativas de forma desnecessária quando uma opção verdadeira já foi encontrada.

```
switch (variável)
{
    case constante1:
        Instruções;
        break;

    case constante2:
        Instruções;
        break;

    default
        Instruções;
}
```

Exemplo uso switch case

```
#include <stdio.h>
#include <conio.h>
int main (void )
{
    int valor;

    printf ("Digite um valor de 1 a 7: ");
    scanf("%d", &valor);

    if (valor == 1)
        printf ("Domingo\n");
    else
        if (valor == 2)
            printf ("Segunda\n");
        else
            if (valor == 3)
                printf ("Terça\n");
            else
                if (valor == 4)
                    printf ("Quarta\n");
                else
                    if (valor == 5)
                        printf ("Quinta\n");
                    else
                        if (valor == 6)
                            printf ("Sexta\n");
                        else
                            if (valor == 7)
                                printf ("Sabado\n");
                            else
                                printf ("Valor invalido!\n");

    getch();
    return 0;
```

```
#include <stdio.h>
#include <conio.h>
int main (void )
{
    int valor;

    printf ("Digite um valor de 1 a 7: ");
    scanf("%d", &valor);

    switch ( valor )
    {
        case 1 :
            printf ("Domingo\n");
            break;

        case 2 :
            printf ("Segunda\n");
            break;

        case 3 :
            printf ("Terça\n");
            break;

        case 4 :
            printf ("Quarta\n");
            break;

        case 5 :
            printf ("Quinta\n");
            break;

        case 6 :
            printf ("Sexta\n");
            break;

        case 7 :
            printf ("Sabado\n");
            break;

        default :
            printf ("Valor invalido!\n");
    }

    getch();
    return 0;
}
```

Operador ternário

- A linguagem de programação C possui o operador ternário `?:`, que representa uma [expressão condicional](#). Sua sintaxe é:
- **<condição> ? <operação 1> : <operação 2>;**
- Essa expressão avalia para <operação 1> se a <condição> for verdadeira. Caso contrário, avalia para a <operação 2>.

```
#include<stdio.h>
```

```
int main() {
```

```
    int a = 0, i;
```

```
    scanf( "%d" , &i );
```

```
    ( i % 2 ==0 ) ? (a++) : (a--);
```

```
    printf("\n i=%d e eh %s\n", i, (a<0) ? ("impar") : ("par") );
```

```
}
```

Estrutura sequencial



- Uma instrução é executada somente após o término da execução da instrução anterior.
- A ordem das instruções é fundamental.

Ex: Robô

- Vamos imaginar que você tenha que ensinar um robô a desenhar um quadrado no chão.



- Quais os passos necessários para desenhar um quadrado?

Ex: Robô

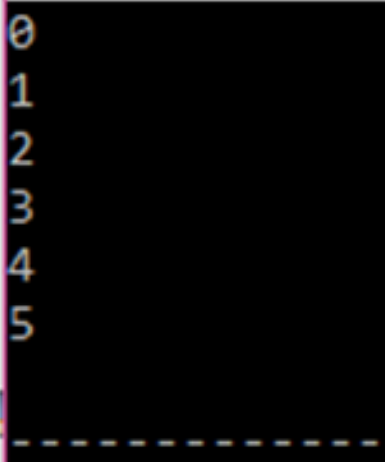
- Então, os passos necessários para desenhar um quadrado são:
 - Pegue a caneta;
 - Avance 3 passos;
 - Vire à direita 90 graus;
 - Avance 3 passos;
 - Vire à direita 90 graus;
 - Avance 3 passos;
 - Vire à direita 90 graus;
 - Avance 3 passos;
 - Solte a caneta.
-
- A ordem das instruções é fundamental. Se invertermos a ordem o resultado final não será o esperado.

Estrutura sequencial em C

```
#include <stdio.h>
```

```
int main( ) {  
    int contador = 0;  
    //imprime o numero 0 na tela  
    printf ("%d \n", contador);  
    contador++;  
    //imprime o numero 1 na tela  
    printf ("%d \n", contador);  
    contador++;  
    //imprime o numero 2 na tela  
    printf ("%d \n", contador);  
    contador++;  
    //imprime o numero 3 na tela  
    printf ("%d \n", contador);  
    contador++;  
    //imprime o numero 4 na tela  
    printf ("%d \n", contador);  
    contador++;  
    //imprime o numero 5 na tela  
    printf ("%d \n", contador);  
  
    return 0;  
}
```

Saída do programa



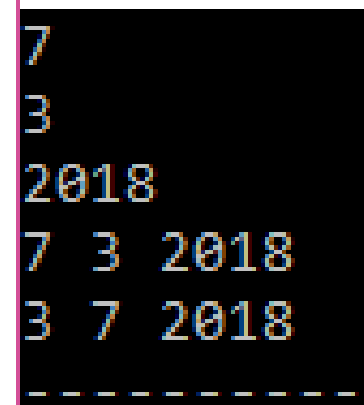
```
0  
1  
2  
3  
4  
5
```

Estrutura sequencial em C

```
#include <stdio.h>
```

```
int main( ) {  
    int dia, mes, ano;  
    scanf ( "%d %d %d", &dia, &mes,  
            &ano );  
  
    //impressão formato brasileiro de data  
    printf ( "%d %d %d", dia, mes, ano );  
    //impressão formato americano de data  
    printf ( "%d %d %d", mes, dia, ano );  
  
    return 0;  
}
```

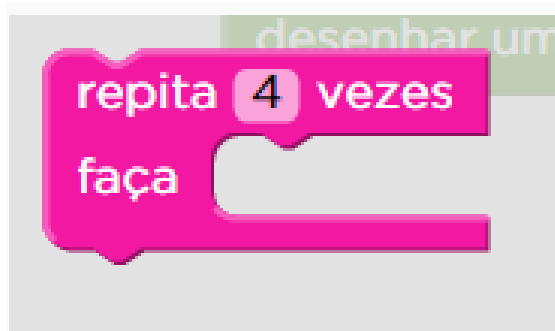
Saída do programa



A terminal window with a black background and yellow text. The output of the program is displayed on five lines: the first line shows '7', the second line shows '3', the third line shows '2018', the fourth line shows '7 3 2018', and the fifth line shows '3 7 2018'. A dashed line is visible at the bottom of the terminal window.

```
7  
3  
2018  
7 3 2018  
3 7 2018  
-----
```


Estrutura de repetição



Estrutura de repetição



Estrutura de repetição com condição

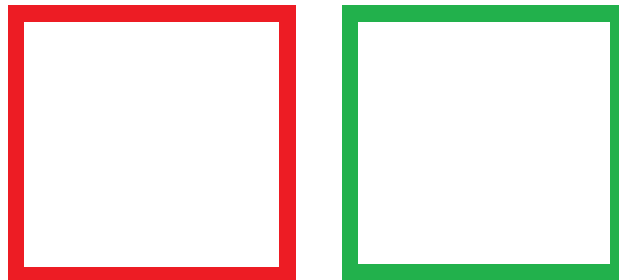
- Em muitos casos, as mesmas instruções são repetidas diversas vezes.
- Para tornar o programa mais eficiente, precisamos utilizar o menor número possível de instruções.

Desafio

- No exemplo do quadrado, podemos reescrever as instruções utilizando a estrutura de repetição.
- *Pegue a caneta*
- ***Repita 4 vezes:***
 - *Avance 3 passos*
 - *Vire à direita 90 graus*
- ***Fim Repita***
- *Solte a caneta*

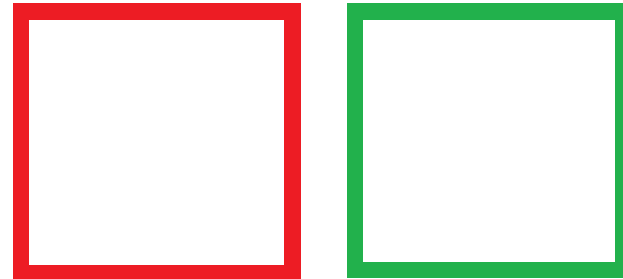
Desafio

- Usando a estrutura de repetição, como você escreveria o pseudocódigo para solucionar o problema: desenhar dois quadrados, um ao lado do outro, com um passo de distância entre eles?



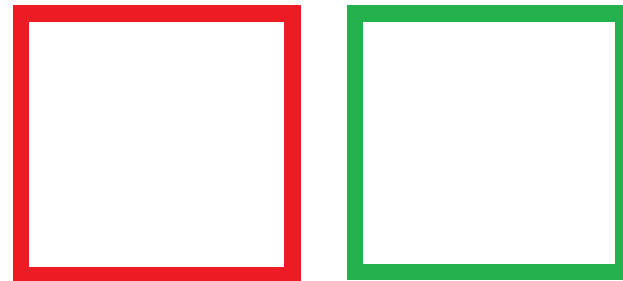
Ex. Robô

- No exemplo do quadrado, podemos reescrever as instruções utilizando a estrutura de repetição.
- *Pegue a caneta*
- ***Repita 4 vezes:***
 - *Avance 3 passos*
 - *Vire à direita 90 graus*
- ***Fim Repita***
- ***Vire à esquerda***
- ***Avance um passo***
- ***Repita 4 vezes:***
 - *Avance 3 passos*
 - *Vire à direita 90 graus*
- ***Fim Repita***
- *Solte a caneta*



Ex. Robô

- No exemplo do quadrado, podemos reescrever as instruções utilizando a estrutura de repetição.
- *Pegue a caneta*
- ***Repita 2 vezes:***
 - ***Repita 4 vezes:***
 - *Avance 3 passos*
 - *Vire à direita 90 graus*
 - ***Fim Repita***
 - ***Vire à esquerda***
 - ***Avance um passo***
- ***Fim Repita***
- *Solte a caneta*

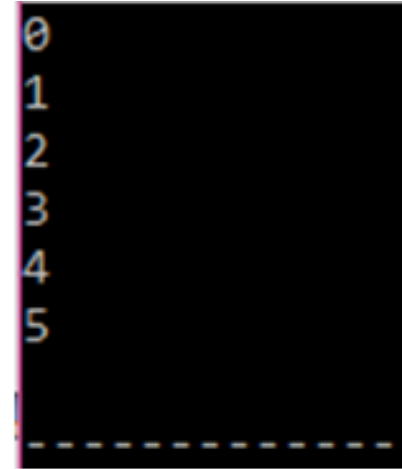


Estrutura de repetição: while

```
#include <stdio.h>
```

```
int main( ) {  
    int contador = 0;  
    while (contador < 6){  
  
        //imprime um numero na tela  
        printf ( "%d\n", contador );  
        contador++;  
    }  
  
    return 0;  
}
```

Saída do programa



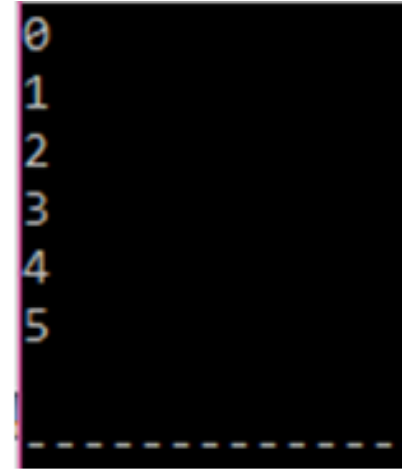
```
0  
1  
2  
3  
4  
5
```

Estrutura de repetição: do-while

```
#include <stdio.h>
```

```
int main( ) {  
    int contador = 0;  
    do {  
        //imprime um numero na tela  
        printf ( "%d\n", contador );  
        contador++;  
    } while ( contador < 6 );  
  
    return 0;  
}
```


Saída do programa



A terminal window with a black background and yellow text. It displays the output of the program: the numbers 0, 1, 2, 3, 4, and 5, each on a new line. The numbers are aligned to the left. At the bottom of the terminal, there is a dashed line.

While x do-while


```
#include <stdio.h>
```

```
int main() {  
    int contador = 6;   
    while (contador < 6){
```

```
        //imprime um numero na tela  
        printf ("%d", contador);  
        contador++;  
    }
```

```
    return 0;  
}
```

```
#include <stdio.h>
```

```
int main() {  
    int contador = 6;   
    do{
```

```
        //imprime um numero na tela  
        printf ("%d\n", contador);  
        contador++;  
    }while (contador < 6);
```

```
    return 0;  
}
```


While x do-while

```
#include <stdio.h>
```

```
int main() {
```

```
    int contador = 6;
```

```
    while (contador < 6){
```

```
        //imprime um numero na tela
```

```
        printf ("%d", contador);
```

```
        contador++;
```

```
    }
```

```
    return 0;
```

```
}
```

Não imprime nada!

```
#include <stdio.h>
```

```
int main() {
```

```
    int contador = 6;
```

```
    do{
```

```
        //imprime um numero na tela
```

```
        printf ("%d\n", contador);
```

```
        contador++;
```

```
    }while (contador < 6);
```

```
    return 0;
```

```
}
```

A terminal window with a black background. The number '6' is printed in a light blue color, followed by a black rectangular redaction box.

Estrutura de repetição: while

```
#include <stdio.h>
```

```
int main( ) {
```

```
    int contador = 0;
```

```
    while (contador < 6){
```

```
        //imprime um numero na tela
```

```
        printf ( "%d\n", contador );
```

```
    }
```

```
    return 0;
```

```
}
```

Saída do programa

?

Estrutura de repetição: while

```
#include <stdio.h>
```

```
int main( ) {  
    int contador = 0;  
    while (contador < 6){  
        //imprime um numero na tela  
        printf ( "%d\n", contador );  
    }  
  
    return 0;  
}
```

Saída do programa

Está em loop infinito!
Preciso uma condição
de parada!!

Observações sobre while

- Permite repetir instruções enquanto uma condição for verdadeira. Você precisa:
 - **Inicializar as variáveis de controle** do loop while antes do comando;
 - Certificar-se que a condição do while se **mantém verdadeira pelo número correto de iterações**;
 - E por fim garantir que **a condição se torne falsa** para terminar o looping.
 - **Existe diferença entre while e do-while na primeira verificação**, se passar a primeira então não tem diferença entre eles.

Exercícios

Assista as Aulas 7, 8 e 9:

<https://www.cursoemvideo.com/course/curso-de-algoritmos/>



Algoritmo

Curso de Algoritmo

★★★★★ (2016 RESENHAS)

120022 ALUNOS

Professores

GUSTAVO GUANABARA

HOME

HOME / CURSO / ALGORITMO / CURSO DE ALGORITMO

Curso de Algoritmo

120022 ALUNOS

Hoje em dia, algoritmos computacionais estão presentes em quase tudo na nossa vida. Além dos tradicionais computadores e notebooks, muitos estão totalmente acostumados com o uso de aplicativos para smartphones e tablets, TVs inteligentes podem executar programas personalizados e até mesmo outros aparelhos que usamos no nosso dia-a-dia.

O Curso de Algoritmo é a base necessária para quem quer aprender em linguagens famosas do mercado, como C, Java, PHP e muitas outras. Inscreva-se no curso agora mesmo e aprenda as técnicas básicas para a construção de programas para dispositivos eletrônicos.

EMENTA DO CURSO



Curso de Algoritmos – 01 –
Introdução a Algoritmos

GRÁTIS



00:14:00

Exercício 1

- Leia um número inteiro que representa um código de DDD para discagem interurbana. Em seguida, informe à qual cidade o DDD pertence, considerando a tabela abaixo. Se a entrada for qualquer outro DDD que não esteja presente na tabela acima, o programa deverá informar: DDD não cadastrado!

Obs: use o comando switch-case.

DDD	Destination
61	Brasilia
71	Salvador
11	Sao Paulo
21	Rio de Janeiro
32	Juiz de Fora
19	Campinas
27	Vitoria
31	Belo Horizonte

Exercício 2

- Faça um programa que encontre as raízes reais de uma equação do segundo grau na forma: $ax^2 + bx + c = 0$. Implemente o cálculo das raízes como uma função e considere os casos em que delta é igual a zero, maior que zero e menor que zero.

$$x = \frac{-b \pm \sqrt{\Delta}}{2.a}$$

$$\Delta = b^2 - 4.a.c$$

```
while (not edge) {  
    run();  
}
```

```
do {  
    run();  
} while (not edge);
```

