

# **Aula 4**

## **Estrutura de Repetição**

**Responsável**

Prof. Armando Toda ([armando.toda@usp.br](mailto:armando.toda@usp.br))

# **Revisão Estrutura condicional**

# Representação de Algoritmos

- Declaração de variáveis:
  - DECLARE
- Leitura/Escrita
  - Leitura de dados: LEIA
  - Escrita de dados: IMPRIMA
- **Estrutura Condicional**
  - **Simples: SE-ENTAO**
  - **Composta: SE-ENTAO-SENAO**
- Estrutura de Repetição
  - PARA
  - ENQUANTO
  - REPITA

## Revisão: Estrutura condicional

- As condições de decisão devem ser compostas por comparações entre 2 elementos
  - SE  $a < b$  ENTÃO ....
  - SE  $y > 2$  ENTÃO ....
- As condições de decisão podem ser compostas por múltiplas comparações
  - SE  $a < b$  OU  $b < c$  ENTÃO
  - SE  $x < 2$  E  $x > 0$  ENTÃO (  $0 < x < 2$  --- não vale)
- **Crie casos de teste para verificar se sua solução é correta**
- **Quanto melhor você entender o problema mais bonita será a sua solução**

Dado três inteiros crie um algoritmo para retornar o menor deles

## Resposta

LEIA n1, n2, n3

SE (n1 <= n2) ENTAO

SE (n1 <= n3) ENTAO

IMPRIME n1

SENAO

IMPRIME n3

FIMSE

SENAO

SE (n2 <= n3) ENTAO

IMPRIME n2

SENAO

IMPRIME n3

FIMSE

FIMSE

E como fazer ???????

Escrevemos 10  
linhas de código  
para encontrar o  
menor valor dentre  
3 números

E como fazer ??????

Mas e se você quiser  
achar o menor valor  
dentre 10, 100 ou  
1000 números?

O que fazer?



E como fazer ??????

# Como decompor o problema?



# Representação de Algoritmos

- Declaração de variáveis:
  - DECLARE
- Leitura/Escrita
  - Leitura de dados: LEIA
  - Escrita de dados: IMPRIMA
- Estrutura Condicional
  - Simples: SE-ENTAO
  - Composta: SE-ENTAO-SENAO
- **Estrutura de Repetição**
  - **PARA**
  - **ENQUANTO**
  - **REPITA**

## Estrutura de Repetição

- Uma estrutura de repetição é utilizada quando um comando ou um bloco de comandos deve ser repetido.
- A quantidade de repetições pode ser fixa ou pode depender de uma determinada condição.
- O teste da condição pode ocorrer no início ou no final da estrutura de repetição.

- Três tipos de estruturas serão consideradas na elaboração de Algoritmos:
  - Estrutura **PARA**
  - Estrutura **ENQUANTO**
  - Estrutura **REPITA**

## Estrutura de Repetição: PARA

**PARA**  $i \leftarrow \text{valorInicial}$  **ATÉ**  $\text{valorFinal}$  **PASSO**  $\times$  **FAÇA**

Instrução\_1

Instrução\_2

....

Instrução\_n

**FIMPARA**

## Estrutura de Repetição: PARA

- Normalmente utilizada quando é conhecido o número de repetições.
- A variável  $i$  é utilizada como controle, variando do `valor_inicial` até `valor_final`.
- O valor do incremento pode ser determinado  
PARA  $i \leftarrow \text{valor\_inicial}$   
ATÉ `valor_final` PASSO  $\times$  FAÇA

## Estrutura de Repetição: PARA

- Exemplos:

PARA  $i \leftarrow 1$  ATÉ 10 PASSO 1 FAÇA  
    ESCREVA  $i$

FIMPARA

**Saida: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10**

PARA  $i \leftarrow 10$  ATÉ 5 PASSO -1 FAÇA  
    ESCREVA  $i$

FIMPARA

**Saida: 10, 9, 8, 7, 6, 5**

## Exercício

- Faça um algoritmo que exiba na tela uma contagem de 1 até 30, exibindo apenas os múltiplos de 3.

Entrada

-

Saída

3 - 6 - 9 - 12 - 15 - 18 - 21 - 24 - 27 - 30



## Exercício

INICIO

inteiro i

PARA i  $\leftarrow$  1 ATÉ 30 PASSO 1 FAÇA

SE (i % 3 == 0) ENTÃO

MOSTRE i

FIMSE

FIMPARA

FIM

## Exercício

```
#include <stdio.h>

int main(){
    int i;
    for(i=0; i<=30; i++){
        if(i%3==0){
            printf("%d ",i);
        }//FIMSE
    }//FIMPARA
}//FIM
```

## Exercício

- Faça um algoritmo que receba o início e o fim do laço de repetição e imprima todos os números PARES dentro desse intervalo e mostre o total de números ÍMPARES dentro do mesmo intervalo.

Entrada

2, 10

Saída

2 - 4 - 6 - 8 - 10

Total de ímpares: 4

## Exercício

INICIO

inteiro inicio, fim, impar;

impar = 0;

LEIA(inicio, fim)

PARA inteiro i = inicio ATÉ fim PASSO 1 FAÇA

SE  $(i \% 2 == 0)$  ENTÃO

MOSTRE(i)

SENÃO

impar+=1

FIMSE

FIMPARA

MOSTRE(impar)

FIM

## Exercício

```
#include <stdio.h>

int main(){
    int inicio, fim, impar;
    impar = 0;
    scanf("%d %d", &inicio, &fim);
    for(int i=inicio; i<=fim; i++){
        if(i%2==0){
            printf("%d ",i);
        }else impar+=1;
    }
    printf("\nTotal de ímpares: %d", impar);
}
```

## Estrutura de Repetição: PARA

- Dinâmica:

```
PARA i ← X ATÉ Y PASSO 1 FAÇA  
    PASSE_PARA_O_PROXIMO(aluno)  
    SE(aluno_for_alergico == TRUE) ENTAO  
        PASSE_PARA_O_PROXIMO(aluno)  
    FIMSE  
FIMPARA  
    PODE_FICAR_COM_A_PAÇOCA(aluno)
```

# Representação de Algoritmos

- Declaração de variáveis:
  - DECLARE
- Leitura/Escrita
  - Leitura de dados: LEIA
  - Escrita de dados: IMPRIMA
- Estrutura Condicional
  - Simples: SE-ENTAO
  - Composta: SE-ENTAO-SENAO
- **Estrutura de Repetição**
  - **PARA**
  - **ENQUANTO**
  - **REPITA**

## Estrutura de Repetição: ENQUANTO

**ENQUANTO** *condição Verdadeira* **FAÇA**

Instrução\_1

Instrução\_2

...

Instrução\_n

**FIMENQUANTO**



## Estrutura de Repetição: ENQUANTO

```
while (condição Verdadeira) {  
    Instrução_1  
    Instrução_2  
    ...  
    Instrução_n  
}
```

## Estrutura de Repetição: ENQUANTO

Quantas repetições esse programa irá fazer?

$x = 1;$

$y = 5;$

ENQUANTO  $(x < y)$  FAÇA

$x = x + 2;$

$y = y + 1;$

FIMENQUANTO

## Estrutura de Repetição: ENQUANTO

**ENQUANTO** *música\_tocar* **FAÇA**

PASSE\_A\_PAÇOCA(aluno\_proximo)

**SE** (aluno\_segurar\_paçoca\_mais\_de\_2\_segundos ==  
TRUE) **ENTÃO**

ALUNO\_PERDE\_PAÇOCA

**FIMSE**

**FIMENQUANTO**

PODE\_FICAR\_COM\_A\_PAÇOCA(aluno)

## Exercício

- **Usando laços de repetição**, leia um número inteiro  $N$ . Escreva a soma de todos os números pares de 2 até  $N$ .

Entrada

10

Saída

30

## Exercício

INICIO

inteiro  $n$ ,  $i$ , soma

leia  $n$

$i \leftarrow 2$ ; soma  $\leftarrow 0$

**ENQUANTO** ( $i \leq n$ ) **FAÇA**

**SE** ( $i \% 2 == 0$ ) **ENTÃO**

        soma  $\leftarrow$  soma +  $i$

**FIMSE**

$i = n + 1$

**FIMENQUANTO**

**FIM**

## Exercício

```
#include <stdio.h>
int main(){
    int i, N, soma;
    i = 0; soma = 0;
    scanf("%d", &N);
    while(i <= N){
        if(i%2==0){
            soma+=i;
        }
        i++;
    }
    printf("\nSoma dos pares: %d", soma);
}
```

## Estrutura de Repetição: ENQUANTO

- Normalmente utilizada quando não se sabe exatamente o número de repetições.
- Também pode ser utilizada quando o número de repetições é conhecido.
- A repetição é executada enquanto a condição for **verdadeira**.

# Estrutura de Repetição: REPITA

## REPITA

Instrução\_1

Instrução\_2

...

Instrução\_n

**ATÉ** *condição*



# Estrutura de Repetição: REPITA

```
do{  
    Instrução_1  
    Instrução_2  
    ...  
    Instrução_n  
}while (condição);
```

```
i = 1;
media = 0;
do{
    printf("Nota prova: ");
    scanf("%f",&nota);

    media = (nota + media)/i;
    printf("Media do aluno = %f\n",media);
    i++;
    printf("Digite 1 para adicionar nota ou 0 para sair\n");
    scanf("%d", &resp);
} while (resp == 1);
```

## Estrutura de Repetição: REPITA

- Normalmente utilizada quando não se sabe exatamente o número de repetições.
- Também pode ser utilizada quando o número de repetições é conhecido.
- A repetição é executada ATÉ que a condição se torne verdadeira.
- A diferença entre a estrutura REPITA e ENQUANTO é que as instruções em REPITA serão executadas ao menos uma vez.

# Exercícios

## Exercício

Seja a seguinte série:

1, 4, 9, 16, 25, 36, ...

Escreva um algoritmo que gere esta série até o N-ésimo termo. Este N-ésimo termo é digitado pelo usuário.

## Exercício 2

Num frigorífico existem 90 bois. Cada boi traz preso em seu pescoço um cartão contendo seu número de identificação (1 até 90) e seu peso. Faça um algoritmo que escreva o número e o peso do boi mais gordo e do boi mais magro (supondo os bois não tem pesos iguais).

**[opcional]** E se o número de identificação for qualquer número? Como você modificaria o algoritmo acima?