

# Estruturas de Repetição

Simone de Oliveira Santos

Universidade Federal do Ceará

22 de maio de 2020

# Súmario

- 1 Introdução
- 2 While
- 3 Do/while
- 4 For

# Sumário

1 Introdução

2 While

3 Do/while

4 For

# Introdução

- Estruturas de repetição são usadas para executar instruções repetidamente e de forma controlada
- Evita a repetição escrita de código

**EXEMPLO:** Leia e mostre a soma de 3 números inteiros:

```
int main(){
    int x1, x2, x3, soma;
    printf("Escreva um numero: ");
    scanf("%d",&x1);
    printf("Escreva um numero: ");
    scanf("%d",&x2);
    printf("Escreva um numero: ");
    scanf("%d",&x3);
    soma = x1 + x2 + x3;
    printf("A soma dos numeros e %d.", soma);
}
```

```
int main(){
    int x, soma; //uma variável para receber os valores lidos
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma = x; //soma passa a ser o valor do 1º valor
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma = soma + x; //adiciona o 2º valor à soma
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma = soma + x; //adiciona o 3º valor à soma
    printf("A soma dos numeros e %d.", soma);
}
```

```
int main(){ //usando operadores reduzidos
    int x, soma = 0;
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma += x;
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma += x;
    printf("Escreva um numero: ");
    scanf("%d",&x);
    soma += x;
    printf("A soma dos numeros e %d.", soma);
}
```

# Introdução

## Estruturas de Repetição em C

- While
- Do/while
- For



# Sumário

- 1 Introdução
- 2 While
- 3 Do/while
- 4 For

# While

## Estruturas do While

```
while(<condição>){  
    <instrução1>;  
    <instrução2>;  
    ...  
}
```

# Leia e mostre a soma de 3 números inteiros:

```
int main(){
    int x, soma = 0, cont = 0;
    while(cont < 3){
        printf("Escreva um numero: ");
        scanf("%d",&x);
        soma += x;
        cont++;
    }
    printf("A soma dos numeros e %d.", soma);
}
```

# while

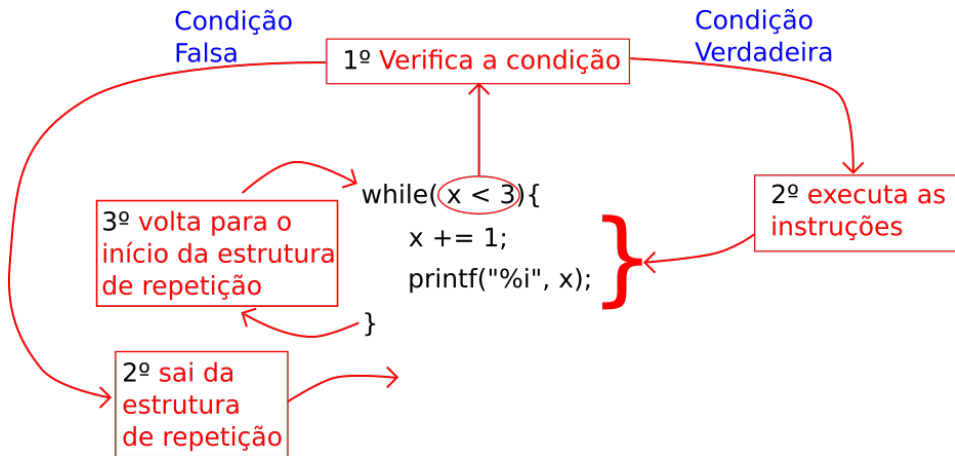
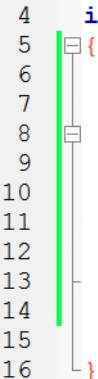


Figura: Etapas de execução da estrutura **while**.

# while

- O bloco de comandos do while é repetido enquanto a condição é **verdadeira**
- A variável **cont** é chamada de **contador**, e serve para controlar o número de iterações da estrutura de repetição
- **Iteração** é cada vez que as instruções do bloco de comandos da estrutura de repetição são executadas
- A condição da estrutura de repetição deve ser bem definida para evitar a repetição infinita da estrutura (*loop*)

# while



```
4  int main()  
5  {  
6      int cont = 0;  
7  
8      while(1) {  
9  
10         cont++;  
11  
12         printf("%d \n", cont);  
13     }  
14  
15     return 0;  
16 }
```

Figura: Exemplo de *loop*. A condição do while será sempre verdadeira (valor 1). Lembrando que em C, qualquer valor diferente de 0 é considerado verdadeiro em uma condição.

**EXEMPLO:** Leia e mostre a soma de N números inteiros digitados pelo usuário.

```
4  int main()  
5  {  
6      int soma = 0;  
7      int x = 0;  
8  
9      while(x != -1) {  
10  
11          soma += x;  
12          printf("Escreva um numero: \n");  
13          scanf("%d", &x);  
14      }  
15  
16      printf("\nA soma dos numeros e: %d", soma);  
17  
18      return 0;  
19 }
```

# while

- Enquanto o usuário não digitar o valor **-1**, o programa lerá o valor digitado pelo usuário, e somará o valor digitado com a soma dos números até agora digitados pelo usuário
- O valor **-1** utilizado para interromper a execução da estrutura **while** é chamado de **sentinela** ou *flag*
- O valor da sentinela deve ser um valor não válido ou não necessário para o algoritmo da estrutura de repetição



# Comando break

- Interrompe a execução da estrutura de repetição
- A execução do programa passa para a primeira instrução após o bloco de repetição

# Comando break

Qual a saída desse programa?

```
4  int main()  
5  {  
6      int cont = 0;  
7  
8      while(cont < 6) {  
9  
10         cont++;  
11  
12         if(cont == 3)  
13             break;  
14  
15         printf("%d \n", cont);  
16     }  
17  
18     return 0;  
19 }
```

# Comando continue

- Interrompe a execução de uma interação
- A execução passa para a próxima iteração

# Comando continue

Qual a saída desse programa?

```
4  int main()  
5  {  
6      int cont = 0;  
7  
8      while(cont < 6){  
9  
10         cont++;  
11  
12         if(cont == 3)  
13             continue;  
14  
15         printf("%d \n", cont);  
16     }  
17  
18     return 0;  
19 }
```

# Sumário

- 1 Introdução
- 2 While
- 3 Do/while
- 4 For

# Do/while

## Estruturas do Do/While

```
do{  
    <instrução1>;  
    <instrução2>;  
    ...  
}while(<condição>;
```

# Do/while

- O bloco de comandos do `do/while` é repetido enquanto a condição é **verdadeira**
- Executa o bloco de comandos pelo menos uma vez

# Do/while

**EXEMPLO:** Escreva os números de 1 à 5

```
int main(){  
    int cont = 1;  
    do{  
        printf("%i ", cont);  
        cont++;  
    }while(cont <= 5);  
  
    return 0;  
}
```



# Do/while

**EXEMPLO:** Escreva os números de 1 à 5

```
int main(){
    int cont = 1;
    do{
        printf("%i ", cont);
    }while(++cont <= 5);

    return 0;
}
```

# While e Do/while

## PRÁTICA:

- 1 Escreva na tela os números de 12 a 18
- 2 Escreva na tela os números de 1034 a 1332
- 3 Escreva os números pares de 100 a 180
- 4 Escreva os números de 33 a 2 na tela

# Sumário

- 1 Introdução
- 2 While
- 3 Do/while
- 4 For

# For

## Estruturas do For

```
for(<iniciaContador>; <condição>; <atualizaContador>){  
    <instrução1>;  
    <instrução2>;  
    ...  
}
```

# For

- Usado para repetição controlada, ou seja, quando deseja-se controlar mais precisamente o número de repetições do *loop*
- Controle realizado através de uma variável contador
- Com o **for** define-se:
  - O valor inicial do contador
  - Uma condição para execução do **for**
  - O quanto o valor do contador será mudado
- Definições feitas no cabeçalho do **for**

# For

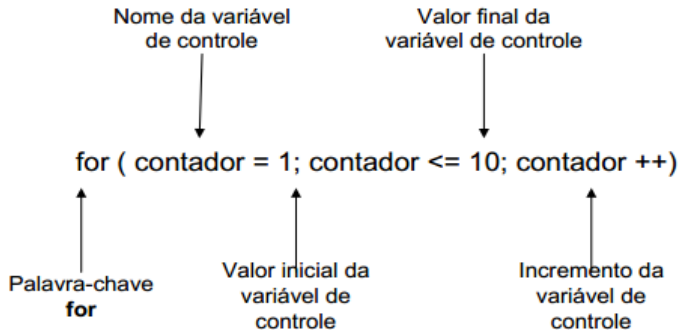


Figura: Cabeçalho da estrutura **for** com a descrição de cada elemento.

# For

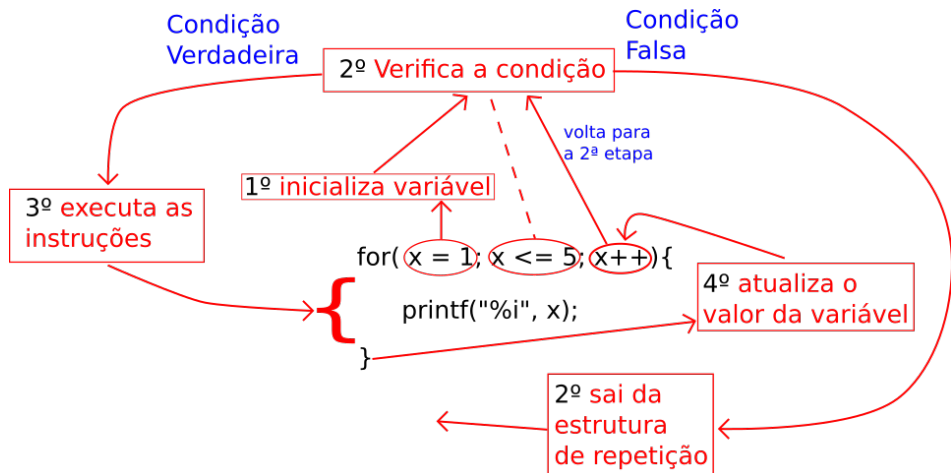


Figura: Etapas de execução da estrutura `for`.

# For

**EXEMPLO:** Escrever os números de 1 à 5

```
4  int main()  
5  {  
6      int cont;  
7  
8      for(cont = 1; cont <= 5; cont++){  
9  
10         printf("%i ", cont);  
11     }  
12  
13     return 0;  
14 }
```

Saída: 1 2 3 4 5



# For

- É possível declarar o contador dentro do cabeçalho do **for**
- Pode ser necessário habilitar a opção `-std = c99` para o compilador

```
4  int main()  
5  {  
6  
7  for(int i = 1; i <= 5; i++){  
8  
9      printf("%d ", i);  
10 }  
11  
12 return 0;  
13 }
```

# For

## PRÁTICA:

- 1 Escreva na tela os números de 20 a 27
- 2 Escreva na tela os números ímpares de 999 a 1402
- 3 Escreva os números múltiplos de 7 de 100 a 210
- 4 Escreva os números divisíveis por 3 entre 500 e 782

# Exemplos Práticos

- 1 Calcule e mostre a soma dos números entre 50 e 150
- 2 Calcule e mostre a soma dos números ímpares entre 50 e 150

# Exemplos Práticos

- 3. Escreva um programa que mostre um menu de opções a seguir, receba opções do usuário para executar uma operação, até que o usuário tecle o número 3.

## **Menu de opções:**

- 1 - Somar dois números
- 2 - Multiplique dois número
- 3 - Sair do programa