



# INTRODUÇÃO À PROGRAMAÇÃO


## Aula 10 - Estruturas de Repetição (Parte 2)

Prof. Carlos Alexandre Siqueira da Silva



Tecnologia em Análise e  
Desenvolvimento de Sistemas

Campus de Alegre



# Revisão da Aula Anterior

- Estruturas de repetição indeterminadas
- Comando `while`
- Comando `do - while`

# Estruturas de Repetição Determinadas

- O bloco de comandos é repetido um número fixo (conhecido) de vezes.
- A cada repetição, uma **variável contadora** é atualizada.
- A repetição termina quando a variável contadora alcança o limite determinado de repetições.

# Comando for

Sintaxe:

```
for(inicialização; condição; incremento){  
    Bloco de comandos  
}
```

O comando for pode ser dividido em 3 partes:

- **inicialização**: Valor inicial da variável contadora (valor inicial da repetição)
- **condição**: Expressão lógica a ser avaliada a cada repetição (valor final da repetição)
- **incremento**: Passo da repetição

# Comando `for` - Exemplo 1

Exemplo 1 – Exibir números de 1 a 5:

```
1 int cont;  
2 for(cont=1; cont<6 ; cont++) {  
3     printf("Contador: %d\n", cont);  
4 }
```

# Comando **for** - Exemplo 2

Exemplo 2 – Contagem regressiva de 10 até 0:

```
1 int cont;  
2 printf("Contagem Regressiva\n");  
3 for(cont=10; cont>=0 ; cont--) {  
4     printf("%d\n", cont);  
5 }
```

# Diferenças entre os comandos **while**, **do - while** e **for**

- **while**

- Repetição Indeterminada.
- O teste da condição é feito **antes da execução**.
- O bloco de comandos pode não ser executado se a condição for falsa desde o início.
- Utilizado quando a repetição depende de uma condição de entrada.

- **do - while**

- Repetição Indeterminada.
- O teste da condição é feito **após a execução**.
- O bloco de comandos **sempre** será executado ao menos uma vez.
- Utilizado quando a repetição precisa executar ao menos uma vez.

- **for**

- Repetição Determinada.
- O teste da condição é feito **antes da execução**.
- O bloco de comandos pode não ser executado se a condição for falsa desde o início.
- Utilizado quando há número fixo de repetições.

# Comando `break`

O comando `break` é um comando de controle de fluxo que **interrompe imediatamente a execução da repetição**, mesmo que a condição ainda seja verdadeira.

Ele é usado quando uma condição especial é atingida e não é necessário continuar o loop. Pode ser utilizado com qualquer estrutura de repetição (`while`, `do-while` ou `for`).



# Comando `continue`

O comando `continue` é um comando de controle de fluxo que **faz o programa pular para a próxima iteração da repetição**, sem executar o restante do bloco atual.

Ele é usado quando queremos ignorar casos específicos sem interromper o loop por completo. Pode ser utilizado com qualquer estrutura de repetição (`while`, `do-while` ou `for`).

# Comando **break** - Exemplo

Exemplo 1 – Exibir números de 1 a 5:

```
1 int cont;  
2 for(cont=1; cont<10 ; cont++) {  
3     printf("Contador: %d\n", cont);  
4     if(cont == 5) {  
5         break;  
6     }  
7 }
```

# Comando `continue` - Exemplo

Exemplo 1 – Exibir números de 1 a 10, pulando o 3:

```
1 int cont;  
2 for(cont=1; cont<=10 ; cont++) {  
3     if(cont == 3) {  
4         continue;  
5     }  
6     printf("Contador: %d\n", cont);  
7 }
```

# O que vem por aí...

Nas próximas duas semanas, teremos as Provas Presenciais, portanto não teremos conteúdo no AVA nem outras atividades avaliativas ou de fixação.

Após as provas presenciais, retornaremos com o conteúdo de **Vetores**.

Boas Provas!