

# Comandos de Repetição

*Aula 07*

**Marcos Silvano Almeida**

*[marcossilvano@professores.utfpr.edu.br](mailto:marcossilvano@professores.utfpr.edu.br)*

Departamento de Computação

UTFPR Campo Mourão

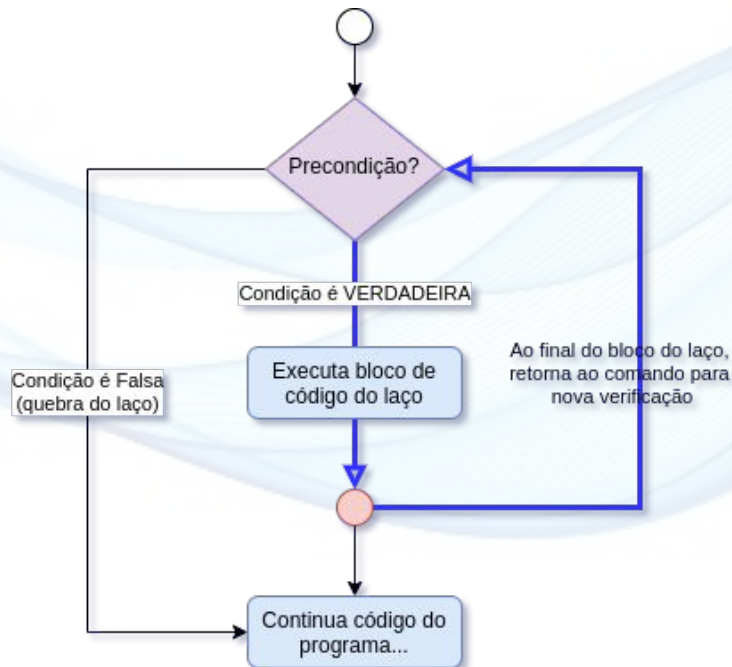
# Roteiro

- Comandos de seleção e exemplos
  - While
    - Laços de contagem
  - For
  - Quebra manual
  - Do-While
  - Escopos de variáveis

# Comandos de Repetição WHILE

# Comando WHILE

- A estrutura de repetição ou laço (loop) é utilizada para repetir um bloco de comandos enquanto uma condição for verdadeira
  - A condição é verificada antes da execução do laço (comandos WHILE e FOR)



# Comando WHILE

- O laço é executado enquanto a condição for satisfeita (verdadeira).
  - O laço pode não executar logo na primeira verificação, caso a condição seja **falsa** na entrada.
- Teste da condição é feito no início, antes da execução do bloco de código do comando **while**.

```
while (condição) {  
    linha1;  
    linha2;  
    linha3;  
    ...  
}
```

# Comando WHILE: exemplo

```
#include <stdio.h>

int main () {
    printf("\nCONTANDO 1..10\n\n");

    // enquanto a <= 10, executa loop
    int i = 1;
    while( i <= 10 ) {
        printf("%d ", i);
        i++;
    }

    printf("\n\n");
    return 0;
}
```

Observe que foi empregada uma variável para contar o número de iterações (repetições). Isto é chamado de LAÇO DE CONTAGEM.

# Comando WHILE: Laço de contagem

- Um laço de contagem possui uma quantidade predeterminada de repetições.
- Tipicamente, é composto por 3 partes:
  - Criação e inicialização da variável de contagem (contador)
    - `int i = 1;`
  - Definição do limite da contagem (condição de execução do loop)
    - `while( i <= 10 )`
  - Modificação do contador
    - `i++;`

```
// laço de contagem incremental
int i = 1;           // inicia contador
while( i <= 10 ) {   // condição de execução (limite da contagem)
    ...              // Laço é quebrado quando i=11
    ...
    i++;             // incrementa contador
}
```

# Comando WHILE: exemplo 2

- Somar uma quantidade indefinida de números informados pelo usuário

```
int main() {  
    printf("\nSOMA DE NUMEROS INTEIROS:\n\n");  
    printf("Entre com inteiros (0 para cancelar): \n");  
    int num = 1;                // permite entrar no loop  
    int sum = 0;  
    while(num != 0) {           // não sabemos de antemão a quantidade  
        printf("> ");           // de repetições que o laço realizará  
        scanf("%d", &num);  
  
        sum += num;             // acumula a soma  
    }  
    printf("\nSOMA: %d\n", sum);  
    return 0;  
}
```



# Comandos de Repetição FOR

# Comando FOR

- Laços de contagem são muito comuns em programas.
- Por praticidade, podemos usar o comando de repetição FOR, que compacta as 3 partes de um laço de contagem WHILE em uma só linha.
  - Em outras palavras, o comando FOR é uma outra forma de escrever um WHILE

```
// este laço de contagem for...
for(int i = 1; i <= 10; i++) {
    ...
}

// ...tem o mesmo comportamento que este while
int i = 1;
while( i <= 10 ) {
    ...
    i++;
}
```

# Comando FOR

- Forma geral:

```
for(inicia contador; condição de execução; modifica contador) {  
    ...  
}
```

- É comum que programadores C escrevam laços de contagem iniciando em **i = 0** e o executem enquanto **i < limite\_da\_contagem**:

```
// este loop for executa 10 vezes, com i = 0..9 (i=10 quebra o laço)  
for (int i = 0; i < 10; i++) {  
    printf("%d ", i+1);  
}
```

# Quebra manual de loop

- É possível utilizarmos o comando break para quebra manual do laço

```
int main() {  
    printf("Entre com 10 inteiros positivos: \n");  
    int num, sum = 0;  
    // executa loop 10 vezes ou realiza quebra prematura, se num negativo  
    for (int i = 0; i < 10; i++) {  
        printf("> ");  
        scanf("%d", &num);  
        if (num <= 0) {  
            printf("*AVISO* Numero negativo: entrada cancelada! \n");  
            break;  
        }  
        sum += num; // acumula a soma  
    }  
    printf("\nSOMA: %d\n", sum);  
    return 0;  
}
```

# Comando DO-WHILE

- Com DO-WHILE, condição é testada ao final
  - Útil quando desejamos executar o bloco do loop ao menos uma vez

```
int main() {  
    printf("\nSOMA DE NUMEROS:\n\n");  
    printf("Inteiros (0-sair): \n");  
    int num;  
    int sum = 0;  
    do {  
        printf("> ");  
        scanf("%d", &num);  
        sum += num;  
    } while (num != 0);  
  
    printf("\nSOMA: %d\n", sum);  
    return 0;  
}
```

```
int num = 1; // executa bloco  
int sum = 0;  
while (num != 0) {  
    printf("> ");  
    scanf("%d", &num);  
    sum += num;  
}
```

## Escopos de variáveis

# Escopos de variáveis

- Um escopo { } define a extensão de código em que a variável é acessível
  - Variáveis podem ser acessadas somente no escopo em que foram definidas ou em escopos descendentes, a partir de sua declaração

```
int main() {  
    int num = 0;                                // acessível em todo o escopo de main()  
    for (int i = 0; i < 10; i++) {                // i existe apenas dentro do escopo do for  
        printf("> ");  
        scanf("%d", &num);  
        if (num < 0) {                            // acessando i em escopo descendente  
            printf("*AVISO* Entrada cancelada no passo %d! \n", i+1);  
            break;  
        }  
    }  
    printf("Ultimo passo: %d\n", i);              // i não é acessível fora do escopo do FOR  
    printf("Ultimo numero informado: %d\n", num);  
    return 0;  
}
```

# Escopos de variáveis

- O escopo global permite o acesso universal à variável, ao custo de mantê-la na memória durante a execução inteira do programa (evitar)

```
int globalNum = 0; // variável GLOBAL (EVITE!)
int main() {
    int rep; // variável do escopo da main()
    printf("\nLOOPS:\n> ");
    scanf("%d", &rep);
    for (int i = 0; i < rep; i++) { // i existe no escopo do FOR
        if (i % 2 == 0) { // i é par?
            int multi = i * 10; // multi existe no escopo do IF
            printf("[%d] ", multi);
            globalNum += multi; // acessando variável global
        } else {
            printf("%d ", i);
        }
    }
}
```



# Referências

- Algoritmos e Programação
  - Marcela Gonçalves dos Santos
  - Disponível pelo Moodle
- Estruturas de Dados, Waldemar Celes e José Lucas Rangel
  - PUC-RIO - Curso de Engenharia
  - Disponível pelo Moodle
- Linguagem C, Silvio do Lago Pereira
  - USP - Instituto de Matemática e Estatística
  - Disponível pelo Moodle
- Curso Interativo da Linguagem C
  - <https://www.tutorialspoint.com/cprogramming>