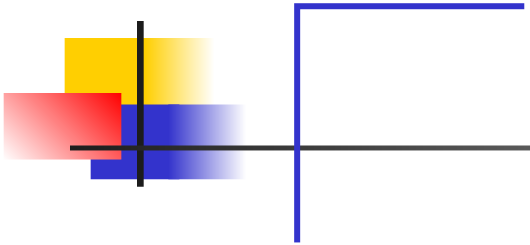


# Estrutura de Repetição

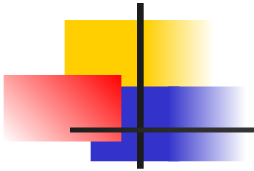
## Comando *for*

# Estrutura de Repetição ou Laços de Repetição

- A estrutura de repetição consiste em definir um grupo de instruções ou uma única instrução que se repete em um algoritmo.

- O desenvolvimento de laços de repetição na Linguagem C é possível com uso de um dos comandos:
  **for, while** e **do while.**

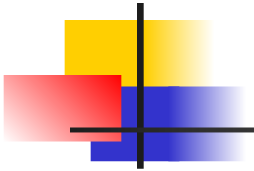
- Neste material, são apresentados os conceitos relacionados ao comando de repetição **for**.



# Estrutura de Repetição: **for**

- Comando **`for`** permite implementar laços de repetição, ou seja, definir trechos do código de um programa que se repetem;
- facilita a implementação de laços com número de execuções definido (quando conhecemos quantas vezes desejamos que a repetição do comando aconteça);
- o comando **`for`** possui 3 partes:

```
for(valor inicial; condição de execução do laço; incremento ou decremento){
        bloco de instruções;
}
```



# Estrutura de Repetição: **for**

- Neste comando, cada uma das partes significa o seguinte:

**`valor inicial`** contém o valor inicial da variável contadora (a variável que conta quantas repetições são realizadas),

**`condição de execução do laço`** contém a condição que será observada para que o laço de repetição aconteça. As instruções serão repetidas enquanto a condição for verdadeira. Quando a condição for falsa, a execução do laço é interrompida e é executada a próxima instrução após o comando `for`.

**`incremento ou decremento`** define como a variável contadora será atualizada após cada iteração (repetição) do laço. Pode ser incrementada ou decrementada.

# Estrutura de Repetição

- **Exemplo 1:** Repetir a escrita de um texto na tela

```c
#include <stdio.h>

int main(){
        int contar;

        for(contar = 1; contar <= 15; contar++){
                printf("========================\n");
        }

        return 0;
}
```

# Estrutura de Repetição

- **Exemplo 1:** Repetir a escrita de um texto na tela, mostrando a variável contadora

```c
#include <stdio.h>

int main(){
        int contar;

        for(contar = 1; contar <= 15; contar++){
                printf("%d ======================\n", contar);
        }

        return 0;
}
```

# Estrutura de Repetição

- **Exemplo 2:** Mostrar na tela os números de 1 até um número informado pelo usuário

```c
#include <stdio.h>

int main(){
        int n, a;

        printf("Digite um número maior que zero: ");
        scanf("%d", &n);

        if (n > 0){
                for(a = 1; a <= n; a++){
                        printf("%d\t", a);
                }
        } else{
                printf("O número deve ser maior que zero!");
        }

        return 0;
}
```

# Estrutura de Repetição

- **Exemplo 3:** Mostrar na tela os números de um número informado pelo usuário até 1

```c
#include <stdio.h>

int main(){
        int n, a;

        printf("Digite um número maior que zero: ");
        scanf("%d", &n);

        if (n > 0){
                for(a = n; a >= 1; a--){
                        printf("%d\t", a);
                }
        } else{
                printf("O número deve ser maior que zero!");
        }

        return 0;
}
```

# Estrutura de Repetição

- **Exemplo 4:** Ler 10 números digitados pelo usuário e mostrar se cada um é par ou ímpar

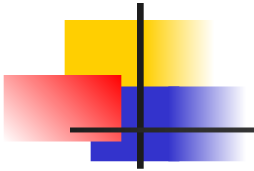
```c
#include <stdio.h>

int main(){
        int n, x;

        for(x = 1; x <= 10; x++){
                printf("Digite um número: ");
                scanf("%d", &n);

                if (n%2==0){
                        printf("Par\n");
                } else{
                        printf("Impar\n");
                }
        }

        return 0;
}
```



# Comando **for**

O comando `for` funciona assim:

- na primeira iteração (repetição), a variável contadora assume o `valor inicial` e se a `condição` for verdadeira, o bloco de instruções é executado. Se a `condição` for falsa, o algoritmo não executa a iteração do laço e continua a execução a partir da primeira instrução após o final `for`.

- Quando o bloco de instruções chegar ao fim, o controle retorna para a linha do `for`. A `condição` é testada e se for verdadeira, o bloco de instruções volta a ser executado. Se for falsa, o algoritmo continua a execução a partir da primeira instrução após o final `for`.

Na primeira iteração, na linha do `for` são executadas a atribuição inicial e a verificação da condição. Nas demais iterações, ao passar pela linha do `for`, no retorno do final do laço, são executadas a operação (incremento ou decremento) e a verificação da condição.