

Programação em C

Laços de repetição

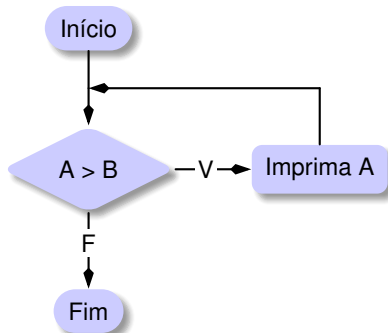
Agostinho Brito

2021

- 1 Laços de repetição - conceito
- 2 A estrutura de repetição for
- 3 A estrutura de repetição while
- 4 A estrutura de repetição do-while
- 5 Laços aninhados
- 6 Laços infinitos
- 7 Declarações de controle de laço

Laços de repetição - conceito

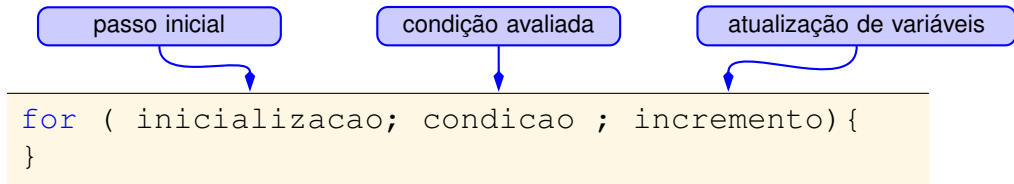
- Algoritmos que demanda repetição são bastante comuns em programação.
- A necessidade surge quando um grupo de declarações precisa ser executado sequencialmente por uma determinada quantidade de vezes, enquanto uma condição lógica é verdadeira, ou até que uma condição lógica seja encontrada.



A estrutura de repetição for

Laços de repetição - estrutura de controle `for`

- A estrutura de repetição `for` é usada quando um determinado trecho de algoritmo precisa ser repetido por uma quantidade pré-estabelecida de vezes.
- Sua estrutura possui a seguinte sintaxe:



- O passo inicial é executado apenas uma vez.
- A condição é avaliada. Se for verdadeira, o laço continua.
- Após o corpo do laço ser executado, o incremento é avaliado.
- O laço continua até que a condição seja falsa.

```
int i;  
for(i=0; i<10; i++){  
    printf("i = %d\n", i);  
}
```

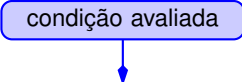


Praticando for...

A estrutura de repetição while

- A estrutura de repetição `while` é usada quando um determinado trecho de algoritmo precisa ser repetido enquanto uma determinada condição for verdadeira.
- Sua estrutura possui a seguinte sintaxe:

condição avaliada



```
while ( condicao ) {  
    }  
}
```

- **PRIMEIRO**, a condição é avaliada. Se for verdadeira, o laço continua.
- O laço continua até que a condição seja falsa.

```
int i=3;
while(i>0) {
    printf("i = %d\n", i);
    printf("digite o novo i: ");
    scanf("%d", &i);
}
```



Praticando while...

A estrutura de repetição do-while

- A estrutura de repetição `while` é usada quando um determinado trecho de algoritmo precisa ser repetido enquanto uma determinada condição for verdadeira.
- Sua estrutura possui a seguinte sintaxe:

```
do{  
} while ( condicao );
```

↑
condição avaliada

- **PRIMEIRO**, o bloco é executado.
- Se a condição for verdadeira, o laço continua.
- O laço continua até que a condição seja falsa.
- Atenção para o **;** no final da estrutura.

```
int i = 0;
do {
    printf("i = %d\n", i);
    printf("digite o novo i: ");
    scanf("%d", &i);
} while (i > 0);
```



Praticando do-while...

Laços aninhados

Laços aninhados

- Todas as estruturas de laço podem ser aninhadas umas com as outras, assim como com as estruturas de decisão.
- Laços aninhados são muito comuns em programação, especialmente quando há necessidade de manipular estruturas matriciais ou varrer estruturas de dados complexas.

```
int i, j;
for (i = 0; i < 10; i++) {
    for (j = 0; j < 10; j++) {
        printf("%2d ", i+j);
    }
    printf("\n");
}
```

Laços infinitos


```
for(;;) {  
    /* declaracoes */  
}
```

```
while(1) {  
    /* declaracoes */  
}
```

```
do {  
    /* declaracoes */  
} while(1);
```

- Laços infinitos só finalizam com **<CTRL>+C**.
- São muito comuns quando um processo deve repetir indefinidamente o mesmo trecho de código durante todo seu ciclo de vida.

Declarações de controle de laço

Declarações de controle de laço

- As declarações de controle de laço permitem ao programador mudar o fluxo normal do algoritmo, permitindo que saltos sejam feitos.
- A declaração `break` finaliza a estrutura de controle e segue para a linha imediatamente posterior ao laço.
- A declaração `continue` pula o restante do bloco e volta ao seu início.
- A declaração `goto` salta da linha atual para uma linha determinada pelo programador, que pode ser inclusive fora da estrutura atual.



Praticando controle de laço...



Obrigado