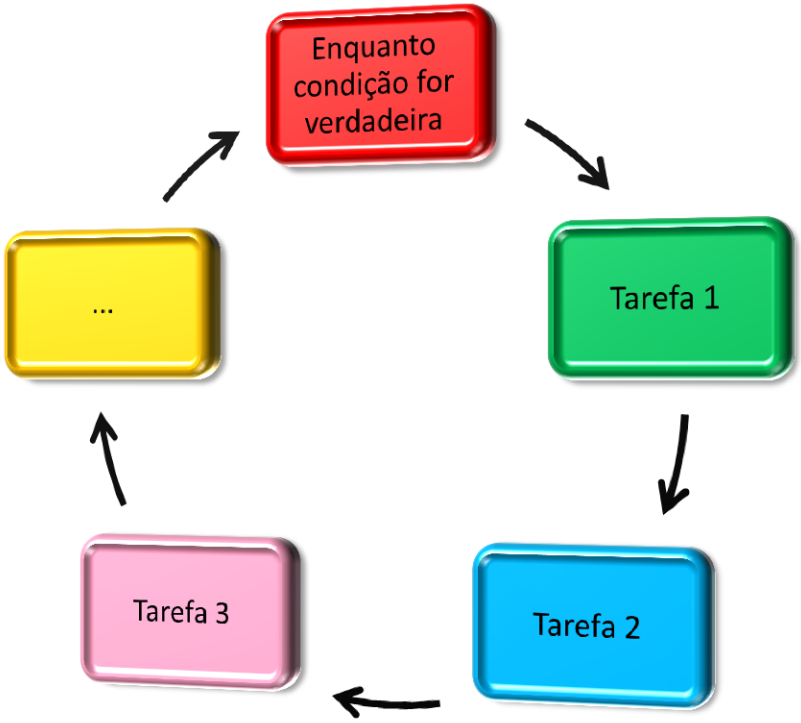
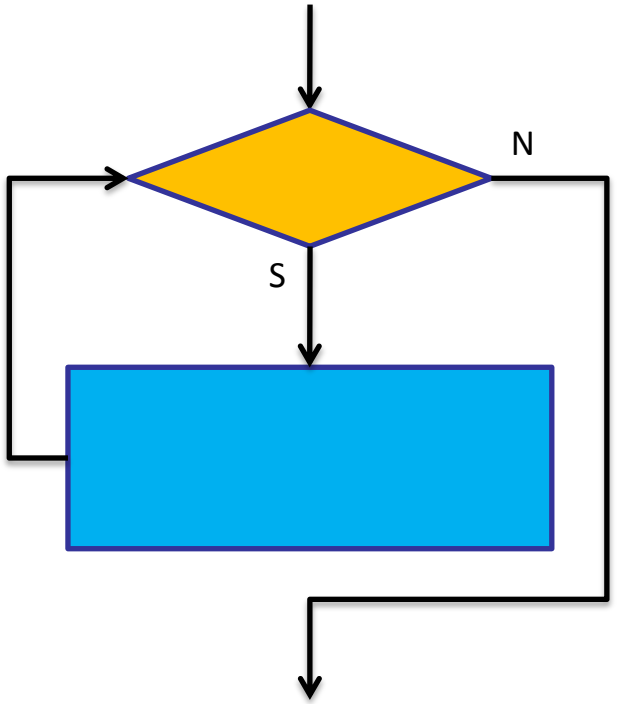
Lógica de Programação

Fabrício Curvello Gomes

Rodrigo Dacome Lima



Programação Com Laços



Laço com Teste Lógico no Início

Laço com Teste Lógico no Início

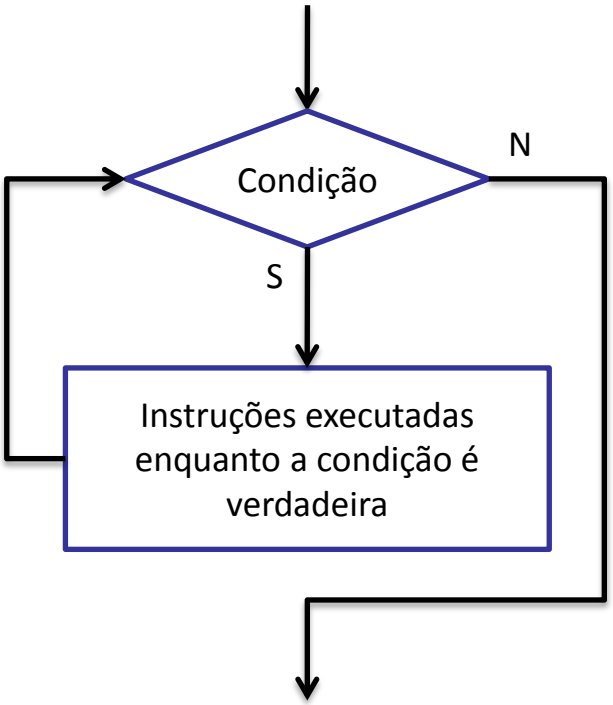
A estrutura da instrução **while (...) { ... }** tem o funcionamento controlado por decisão, e pode executar um determinado conjunto de instruções enquanto a condição verificada for Verdadeira.

No momento em que essa condição se torna Falsa, o processamento da rotina é desviado para fora do laço.

Se a condição for Falsa logo de início, as instruções do laço são ignoradas.

Laço com Teste Lógico no Início(Cont.)

Diagrama de Blocos:

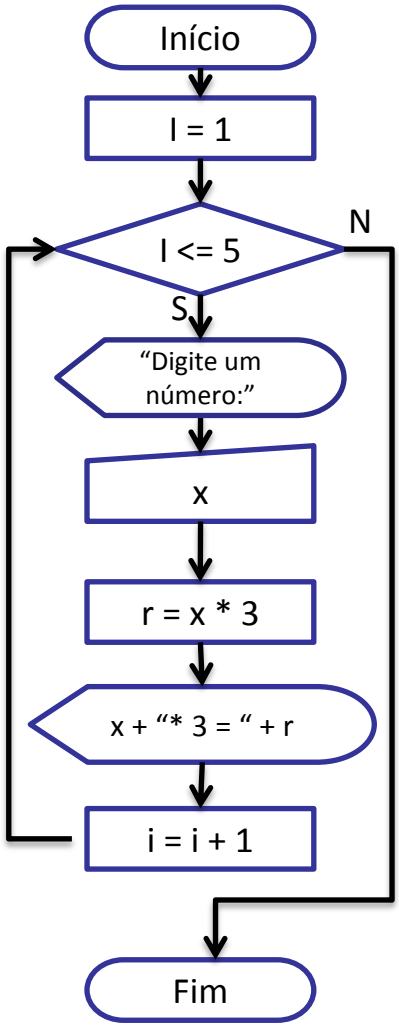


Código:

```
while (<condição>) {
 <Instruções para condição verdadeira>
}
```

Laço com Teste Lógico no Início

Exemplo: Algoritmo para pedir a leitura de um valor para a variável **x**, multiplicar este valor por 3, colocar o valor obtido na variável **r**, e apresentar o valor de **r**, repetindo a sequência cinco vezes.

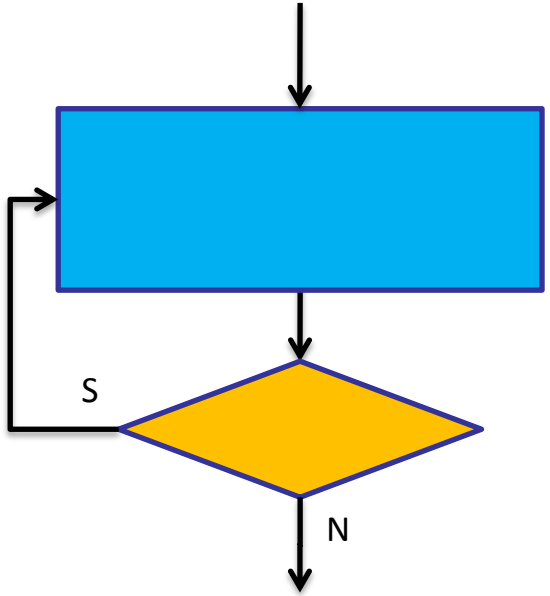



```
#include <iostream>
#include <locale>
using namespace std;

int main(){
    setlocale(LC_ALL,"Portuguese");
    int r, x, i;
    i = 1;
    while (i <= 5) {
      cout << "Digite um número:";
      cin >> x;
      r = x * 3;
      cout << x << " * 3 = " << r << endl;
      i = i + 1;
    }
    return 0;
}
```

Aqui a variável I recebe o valor 1, para que após o comando *while*, ela vire um contador.

Aqui acontece a função de **contador**, que irá encerrar o "looping" da função quando **i** se tornar maior que 5



Laço com Teste Lógico no Fim

Laço com Teste Lógico no Fim

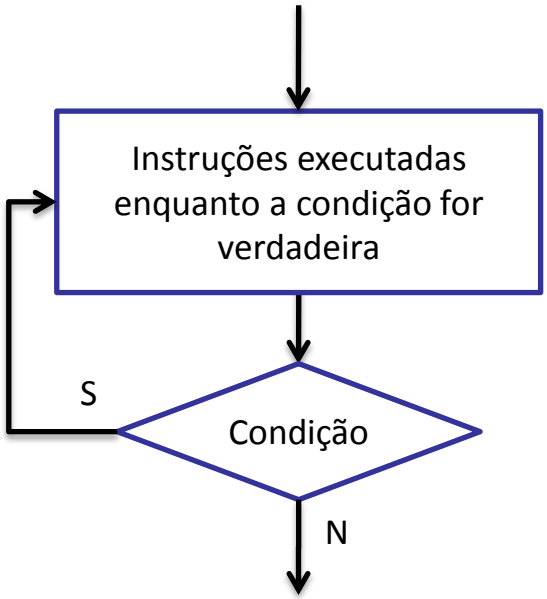
A estrutura da instrução **do { ... } while (...)** também tem o funcionamento controlado por decisão, e pode executar um determinado conjunto de instruções enquanto a condição verificada for Verdadeira.

No momento em que essa condição se torna Falsa, o processamento da rotina é desviado para fora do laço.

O teste é feito ao final do laço, e mesmo que a condição dê falso logo no primeiro teste, pelo menos uma vez o laço será executado.

Laço com Teste Lógico no Fim(Cont.)

Diagrama de Blocos:

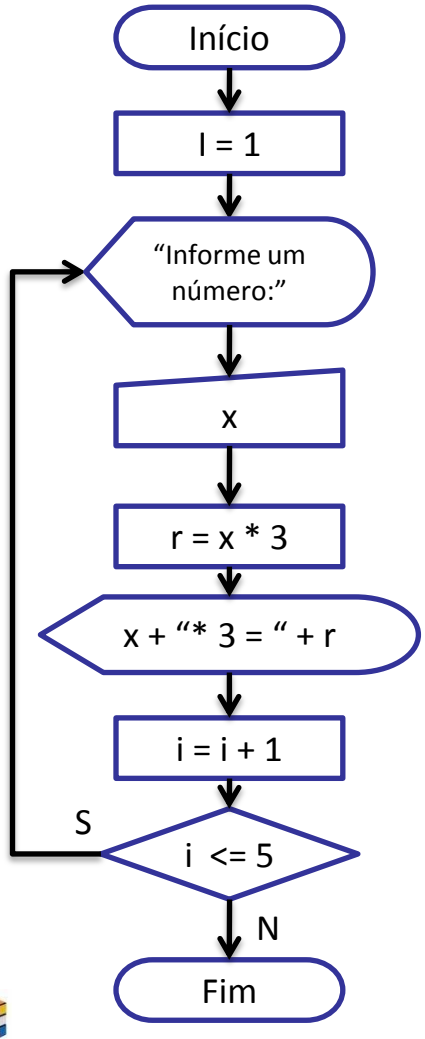


Código:

```
do {
     <Instruções executadas enquanto
      a condição for verdadeira>
} while (<condição>);
```

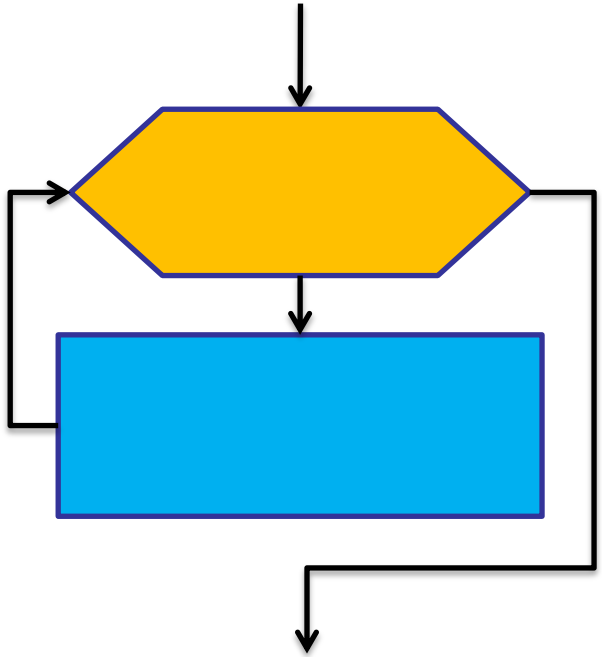
Laço com Teste Lógico no Fim (Cont.)

Vamos utilizar o mesmo exemplo aplicado anteriormente: Algoritmo para pedir a leitura de um valor para a variável **x**, multiplicar este valor por 3, colocar o valor obtido na variável **r**, e apresentar o valor de **r**, repetindo a sequência cinco vezes.




```
#include <iostream>
#include <locale>
using namespace std;

int main(){
    setlocale(LC_ALL,"Portuguese");
    int r, x, i;
    i = 1;
    do{
      cout << "Digite um número:";
      cin >> x;
      r = x * 3;
      cout << x << " * 3 = " << r << endl;
      i = i + 1;
    }while (i <= 5);
    return 0;
}
```



Laço com Variável de Controle

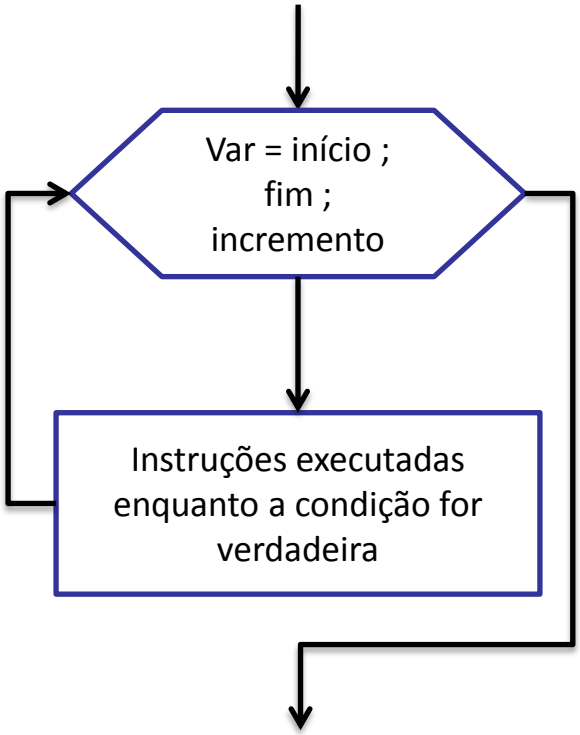
Laço com Variável de Controle

Os laços que possuem um número finito de execuções podem ser processados pela estrutura para, sendo conseguida com a utilização das instruções **for (... ; ... ; ...) { ... }**.

As instruções ***for*** têm o funcionamento controlado por uma variável denominada variável de controle. Sendo assim, pode executar um conjunto de instruções um determinado número de vezes.

Laço com Variável de Controle (Cont.)

Diagrama de Blocos:

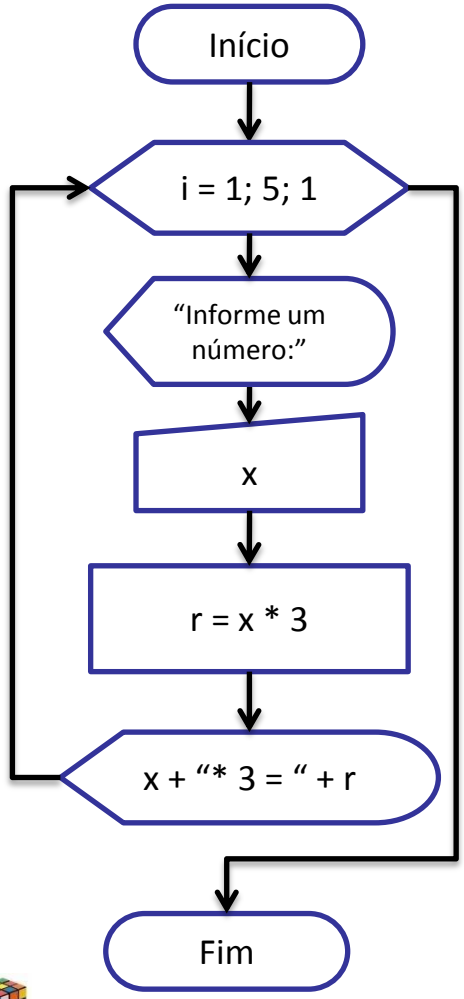


Código:

```
for (<variável> = <início>; <fim>; <incremento>) {
   <instruções>
}
```

Laço com Variável de Controle (Cont.)

Vamos utilizar mais uma vez o mesmo exemplo aplicado anteriormente: Algoritmo para pedir a leitura de um valor para a variável **x**, multiplicar este valor por 3, colocar o valor obtido na variável **r**, e apresentar o valor de **r**, repetindo a sequência cinco vezes.




```
#include <iostream>
#include <locale>
using namespace std;

int main(){
    setlocale(LC_ALL,"Portuguese");
    int r, x, i;
    for (i = 0; i < 5; i++) {
      cout << "Digite um número:";
      cin >> x;
      r = x * 3;
      cout << x << " * 3 = " << r << endl;
    }
    return 0;
}
```

i++ equivale à **i = i + 1**

Dúvidas?

Bibliografia

Estudo Dirigido de Algoritmos
José Auguto N. G. Manzano e Jayr Figueiredo de Oliveira
Ed. Érica

Fundamentos de Computação e Orientação a Objetos Usando JAVA
Francisco A. C. Pinheiro
Ed. LTC