Laços de Repetição

Existem problemas que são repetitivos por natureza. Por exemplo, escrever um algoritmo para calcular a média de um aluno é algo fácil, mas se quisermos calcular a média da turma inteira? A solução mais simples seria executar o algoritmo tantas vezes que o cálculo fosse necessário, embora esta tarefa seja um tanto trabalhosa. Mas se ainda, nesta conta ao final o professor quisesse que fosse mostrada a média mais alta e mais baixa da turma?
Esse e outros problemas podem ser resolvidos com a utilização de laços de repetição. Um laço de repetição, como sugere o próprio nome, é um comando onde uma quantidades de comandos se repete até que uma determinada condição seja verdadeira.
O Portugol contém 3 tipos de laços de repetição: pré -testado, pós-testado e laço com variável de controle.

Nesta seção, serão abordados os seguintes tópicos:

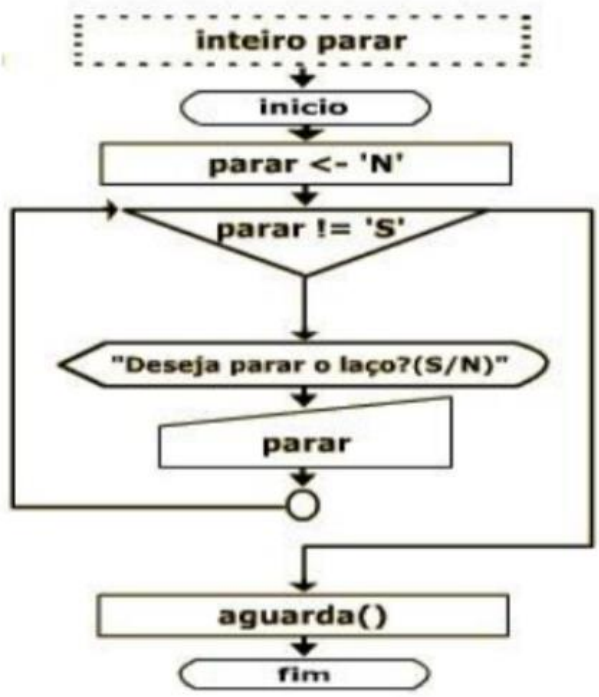
- Enquanto
- Faça-Enquanto
- Para

Laço Enquanto (Pré-Testado)

Se fosse necessário a elaboração de um jogo, como por exemplo um jogo da velha, e enquanto houvessem lugares disponíveis no tabuleiro, este jogo devesse continuar, com o faríamos para que o algoritmo tivesse este comportamento? É simples. O comando enquanto poderia fazer esse teste lógico. A função do comando enquanto é: executar uma lista de comandos enquanto uma*determinada condição for verdadeira.
A sintaxe é respectivamente a palavra reserva da enquanto, a condição a ser testada entre parênteses, e entre chaves a lista de instruções que se deseja executar.

```
logico condicao = verdadeiro
enquanto (condicao)
{
   //Executa a as instruções dentro do laço enquanto a
 condicao for verdadeira
}
```

A figura abaixo ilustra um algoritmo que verifica uma variável do tipo carácter. Enquanto a variável for diferente da letra 'S' o comando enquanto será executado, assim com o as instruções dentro dele. No momento em que o usuário atribuir S a variável, o comando enquanto terminará e o programa chega ao seu final.



O exemplo a seguir ilustra em Portugol o fluxograma acima.

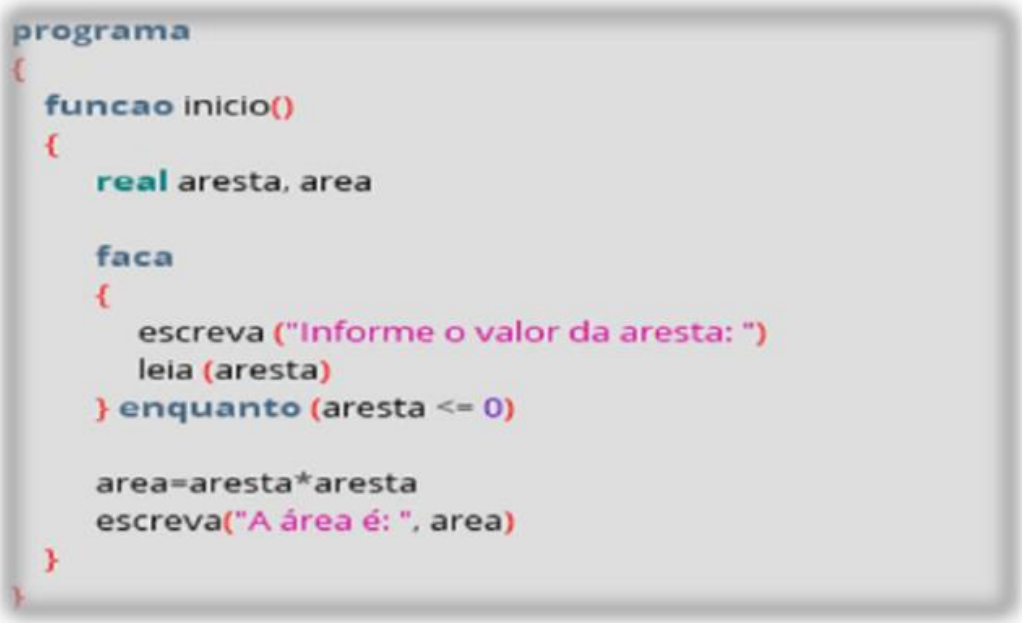
```
programa
{
  funcao inicio()
  {
     caracter parar
     parar = 'N'

     enquanto (parar != 'S')
     {
        escreva ("deseja parar o laço? (S/N)")
        leia (parar)
     }
  }
}
```

Laço Faça-Enquanto (Pós-Testado)

Em algumas situações, faz -se necessário verificar se uma condição é verdadeira ou não após uma entrada de dados do usuário. Para situações como essa, podemos usar o laço de repetição faca-enquanto. Este teste é bem parecido com o enquanto. A diferença está no fato de que o teste lógico é realizado no final, e com isso as instruções do laço sempre serão realizadas pelo menos uma vez. O teste verifica se elas devem ser repetidas ou não.
A sintaxe é respectivamente a palavra reservada faca, entre chaves as instruções a serem executadas , a palavra reservada enquanto e entre parênteses a condição a ser testada.

```
logico condicao = verdadeiro
faca
{
   //Executa os comandos pelo menos uma vez, e continua
 executando enquanto a condição for verdadeira
} enquanto (condicao)
```

A figura abaixo ilustra um algoritmo que calcula a área d e um quadrado. Note que para o cálculo da área é necessário que o valor digitado pelo usuário para aresta seja maior que 0. Caso o usuário informe um valor menor ou igual a 0 para a aresta, o programa repete o comando pedindo para que o usuário entre novamente com um valor para a aresta. Caso seja um valor válido, o programa continua sua execução normalmente e ao f im exibe a área do quadrado.



```
programa
{
  funcao inicio()
  {
     real aresta, area

     faca
     {
        escreva ("Informe o valor da aresta: ")
        leia (aresta)
     } enquanto (aresta <= 0)

     area=aresta*aresta
     escreva("A área é: ", area)
  }
}
```

Laço Para (Com Variáv el de Controle)

E se houver u m problema em que sejam necessárias um número determinado de repetições? Por exemplo, se quiséssemos pedir ao usuário que digitasse 10 valores. Poderíamos utilizar a instrução Leia repetidas vezes. Porém se ao invés d e 10 valores precisássemos de 100, essa tarefa se tornaria muito extensa. Para resolver problemas como esse, podemos usar um laço de*repetição com variável de controle. No portugol, ele é conhecido como para.

O laço de repetição com variável de controle facilita a construção de algoritmos com número definido de repetições, pois possui um contador (variável de controle) embutido no comando como incremento automático. Desta forma, um erro muito com um que se comete ao esquecer de fazer o incremento do contador é evitado. Toda vez que temos um problema cuja solução necessita de um número determinado de repetições utilizamos um contador. O contador deve ser inicializado antes do laço e deve ser incrementado dentro do laço.

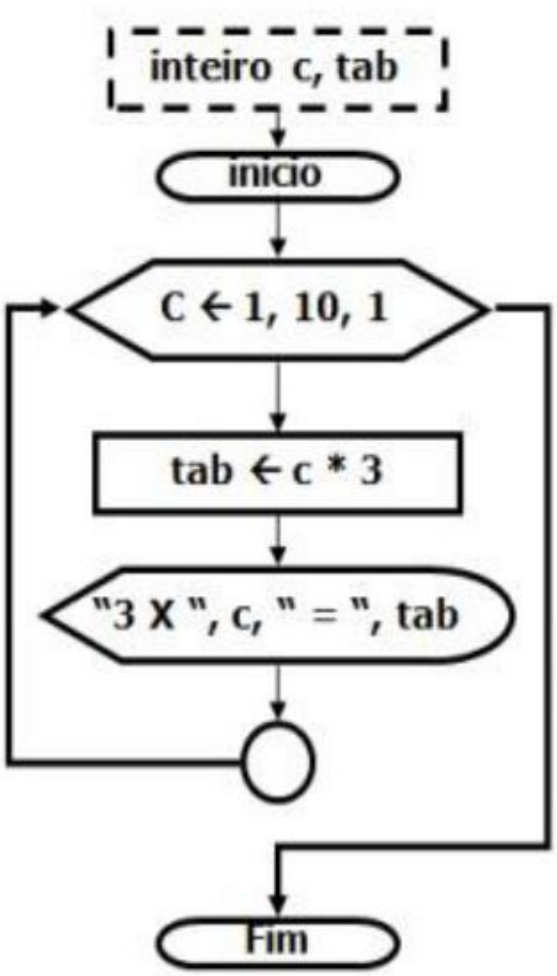
O laço com variável de controle possui três partes. A inicialização da variável contadora, a definição d o valor final do contador e a definição do incremento.

Estas três partes são escritas juntas, no início do laço.

A sintaxe é respectivamente a palavra reservada para, abre parênteses, a declaração de uma variável de controle, ponto e virgula, a condição a ser testada, ponto e virgula, uma alteração*na variável de controle a ser feita a cada iteração, fecha parênteses, e entre chaves as instruções do programa

```
para (inteiro i = 0; i < 8; i++)
{
    //Codigo a ser executado enquanto a condição for satisfeita.
}
```

A f figura abaixo ilustra um algoritmo que exibe na tela a tabuada de 3. Note que conforme a sintaxe mostrada anteriormente , a primeira instrução do laço é inicializar o contador c=1. O segundo comando especifica a condição para que o laço continue a ser executado, ou seja, enquanto o contador c for menor ou igual a1 0. Por ultimo, a terceira instrução demonstra que o contador c será acrescentado em 1 em seu valor a cada iteração do comando. O laço será executado 10 vezes e mostrará a tabuada de 3.




```
programa
{
  funcao inicio()
  {
     inteiro tab

     para (inteiro c=1; c<=10; c++)
     {
        tab=c*3
        escreva ("3 x ", c, " = ", tab, "\n")
     }
  }
}
```