

O que mais precisamos aprender

- Git e GitHub
- Laços de repetição (enquanto, para até);
- Estrutura de seleção múltipla (escolha caso)
- Subrotinas (Funções);
 - Bibliotecas.
 - Recursividade;
- Estruturas de dados (Vetores, Matrizes, Filas e Pilhas);
- Estatística Básica;
- Regra de três;
- Introdução a armazenamento de dados ;

Laços de repetição

- Podemos colocar condições dentro da estrutura **enquanto**

programa

{

funcao inicio() {

inteiro contador = 10

enquanto (contador > 0)

{

limpa()

escreva ("**Detonação em:** ", contador)

contador = contador - 1

aguarde(1000) // Aguarda 1000 milisegundos (1 segundo)

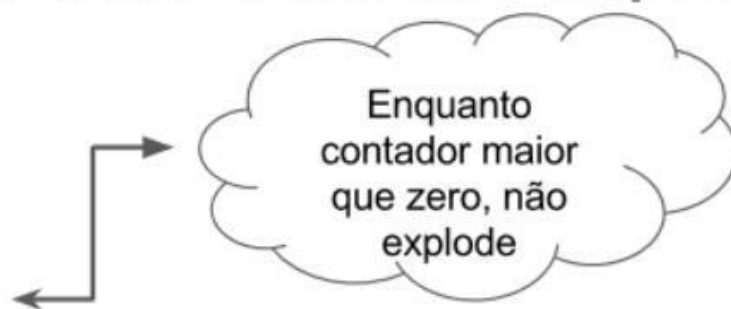
}

limpa()

escreva ("**Booom!\n**")

}

}



Laços de repetição

- Além do enquanto, temos o para... até ... faça;
- Imagine que queremos saber a tabuada de um número. Quais são os requisitos?
 - Escolher um número;
 - Multiplicar o número escolhido por 1 até 10;
- Então para 1 até 10 multiplique o número escolhido.

Laços de repetição

```
programa{  
    funcao inicio(){  
        inteiro numero, resultado, contador  
        escreva("Informe um número para ver sua tabuada: ")  
        leia(numero)  
        limpa()  
        para(contador = 1; contador <= 10; contador++)  
        {  
            resultado = numero * contador  
            escreva (numero, " X ", contador, " = ", resultado , "\n")  
        }  
    }  
}
```

Laços de repetição

```
programa{  
    funcao inicio(){  
        inteiro numero, resultado, contador  
        escreva("Informe um número para ver sua tabuada: ")  
        leia(numero)  
        limpa()  
        para (contador = 1; contador <= 10; contador++){  
            resultado = numero * contador  
            escreva (numero, " X ", contador, " = ", resultado , "\n")  
        }  
    }  
}
```

Laços de repetição

- Se uma ação se repete em um algoritmo, em vez de escrevê-la várias vezes, podemos resumir anotando uma só vez e solicitando que ela se repita, usando umas das estruturas de repetição;
- Podemos pedir que uma ação (ou um conjunto de ações) seja executada um número definido ou indefinido de vezes, ou enquanto um estado permanecer ou até que um estado seja atingido;
- Fora do Portugal, essas estruturas são denominadas do inglês, while (enquanto), do...while (faça...enquanto), e for (para).

Laços de Repetição

Mais alguns exemplos - Repetição de código

```
programa {  
    funcao inicio(){  
        inteiro i  
        para(i=0;i<20;i++)  
            escreva("i")  
        escreva("\n")  
        escreva("Numeros entre 1 e 5\n")  
        para(i=0;i<20;i++)  
            escreva("i")  
        escreva("\n")  
        para(i=1; i<=5; i++)  
            escreva(i, "\n")  
        para(i=0;i<20;i++)  
            escreva("i")  
        escreva("\n")  
    }  
}
```

Note o código repetido. Se
tivermos que consertar,
teremos que fazer o mesmo
ajuste várias vezes



Saída:

Numeros entre 1 e 5

1
2
3
4
5

Laços de Repetição

Mais alguns exemplos - Repetição de código

```
programa {  
    funcao inicio(){  
        inteiro i  
        para(i=0;i<20;i++){  
            escreva("*****")  
            escreva("\n")  
            escreva("Numeros entre 1 e 5\n")  
            para(i=0;i<20;i++){  
                escreva("*****")  
                escreva("\n")  
                para(i=1; i<=5; i++){  
                    escreva(i, "\n")  
                }  
            }  
        }  
    }  
}
```

Note o código repetido. Se tivermos que consertar, teremos que fazer o mesmo ajuste várias vezes



Saída:

Numeros entre 1 e 5

1
2
3
4
5

Estrutura de seleção múltipla - escolha caso

programa

```
{
  funcao inicio()
  {
    inteiro valor=1
    escolha (valor)
    {
      caso 0: //testa se o valor é igual a 0
        escreva ("o valor é igual a 0")
      pare

      caso 1: //testa se o valor é igual a 1
        escreva ("o valor é igual a 1")
      pare

      caso 2: //testa se o valor é igual a 2
        escreva ("o valor é igual a 2")
      pare

      caso contrario:
        escreva ("o valor não é igual a 0, 1 ou 2")
    }
  }
}
```

programa

```
{
  funcao inicio()
  {
    caracter texto
    escreva("Digite s para sim ou n para não:")
    leia(texto)
    escolha(texto)
    {
      caso 's':
        escreva("Você optou pelo sim!")
      pare

      caso 'n':
        escreva("Você optou pelo não!")
      pare

      caso contrario:
        escreva("Opção inválida!")
    }
  }
}
```

Funções

- Definição : Sequência de instruções executadas somente quando chamadas por um programa em execução:
 - Devem executar uma tarefa específica
 - Um programa pode conter diversas funções, além da função principal início(), que é **obrigatória**;
 - As funções executam somente quando chamadas à partir da função início();
 - Após a execução, o fluxo retorna ao ponto imediatamente após o da chamada da função;
 - Uma função pode (ou não) retornar um valor ao bloco que a chamou;
 - Uma função pode (ou não) necessitar de um ou mais argumentos ao ser chamada;