Repetição

Repetição com contador

```
DO varControle = valorInicial, valorFinal, [passo]
   instruções
END DO
```

- VarControle → variável que servirá como contador
- ValorInicial → valor com que a variável varControle iniciará
- ValorFinal →valor máximo valido para varControle
- Passo → passo da contagem (não obrigatório) assume 1 quando não explicitado

Exemplo

```
program contadores
  integer :: i, j, k
  integer ::   limite=10
  do i=1,limite,1
      write(*,*) "contador=", i
  end do
end program contadores
```

Exemplo

```
program contadores
  integer :: i, j, k
  integer , parameter ::   limite=10
  do i=1,limite,1
      write(*,*) "contador=", i
  end do
end program contadores
```

Exercícios

- Remova o passo do exemplo anterior
- Configure o passo para ser zero
- Configure o passo para ser -1

Exercício

- Faça um programa que leia 10 números digitados pelo usuário e imprima estes números na tela

Exercício

- Faça um programa que leia um número, que representa uma quantidade de palavras a serem digitadas. O programa deve então ler estas palavras e imprimi-las na tela

Exercício

- Faça um programa que calcule a multiplicação de números informados pelo usuário. A quantidade de números também deve ser informada

Exercício

- Faça um programa que calcule o fatorial de um número informado pelo usuário

Exercício

- Faça um programa que leia dois números e imprima todos os números pares entre estes números em ordem decrescente

Exercício

- Faça um programa que calcule a média das notas de uma turma. O número de alunos da turma deve ser lido pelo programa

Exercício - Possível?

- Faça um programa que leia e imprima números digitados pelo usuário. O programa deve parar quando o o usuário digitar um número negativo

Repetições sem contador

```
DO
  instruções
END DO
```

Mau exemplo

```
program doInfinito
  integer :: i
  do
    write(*,*)"digite um número"
    read (*,*)i
    write (*,*)"o número digitado foi ",i
  end do
end program doInfinito
```

Como resolver este problema

Uma das instruções dentro do laço deve determinar a parada

Comando Exit → determina a parada do laço mais interno

```
program doInfinito
  integer :: i
  do
    write(*,*)"digite um número"
    read (*,*)i
    if (i <0) exit
    write (*,*)"o número digitado foi ",i
  end do
end program doInfinito
```

Lidando com laços (loops) aninhados

```
program doisLoops
  integer :: i, soma, numAlunos
  do
      soma=0
      numAlunos=0
      write(*,*) "digite as notas dos alunos ou -1
para finalizar o calculo da média"
```

Lidando com laços (loops) aninhados

```
do
  write(*,*)"digite uma nota"
  read (*,*)i
  if (i==-1) exit
  soma=soma+i
  numAlunos = numAlunos+1
end do
```

Lidando com laços (loops) aninhados

```
      write(*,*)"a média desta turma é ", (soma/numAlunos)

      write(*,*)"digite -1 para sair do programa ou qualquer outro valor para calcular a média de outra turma"

      read (*,*)i

      if (i==-1) exit

   end do

end program doisLoops
```

Exercício

- Faça um programa que determine o menor número que uma variável inteira pode representar

```
program menorInteiro
  integer :: i
  i=-1
  do
     i=i-1
     if(i>=0) exit
  end do
  i=i+1
  write(*,*) "o menor inteiro é ",i
end program menorInteiro
```

Exercício

Faça um programa que encontra o máximo divisor comum entre dois números seguindo o critério de Euclides

- 1. Calcule o resto c da divisão de a por b
- 2. Se o resto c for zero, b é o m.d.c.
- 3. Senão, substitua a por b e b por c execute novamente o passo 1

```
PROGRAM  GreatestCommonDivisor
  INTEGER   :: a, b, c
  WRITE(*,*) “Digite dois inteiros--> “
  READ(*,*)  a, b
  IF (a < b) THEN       !  a deve ser maior que b
    c = a                    ! Caso isto não seja verdade eles devem ser trocados
    a = b
    b = c
  END IF
  DO
    c = MOD(a, b)      !    calcula C sendo o resto da divisão
    IF (c == 0) EXIT   !    Se C ==0 b é o MDC
    a = b              !   caso contrário a recebe b
    b = c              !    e b recebe c
  END DO               !    retorna para a divisão
  WRITE(*,*) “o mdc é”', b
END PROGRAM  GreatestCommonDivisor
```

Do Cycle

```
DO informações de controle
   instruções-1
   CYCLE
   instruções-2
END DO
```

Cycle

- O comando cycle, assim como o comando exit, tem a função de interromper o laço mais interno
- O comando exit interrompe o laço de maneira definitiva
- O comando Cycle por sua vez faz com que o laço inicie sua próxima iteração

```
INTEGER :: i
DO i = 1, 5
   IF (i == 3) THEN
      CYCLE
   ELSE
      WRITE(*,*)  i
   END IF
END DO
```

Exercício

- Faça um programa que leia dois números. O programa deve imprimir todos os números primos entre os dois números lidos no primeiro passo

Exercício p1

- O dia da semana para uma data qualquer pode ser aproximado da seguinte maneira:

- Sejam “rD” o resto da divisão, “qD” o quociente, “m” o número do mês (março=1,..., dezembro=10, janeiro=11,fevereiro=12), “d” o dia do mês, “s” os dois primeiros algarismos do ano e “a” os dois últimos

- dia=rD((qD((2.6 m – 0.1) ,1) + d + a + qD(a,4) + qD(s,4) -2s),7)

Exercício p2

- Faça um programa para calcular o dia da semana seguindo esta equação. Dia assume valores de 0 até 6, onde 0 é domingo
- Obs.: para os meses de janeiro e fevereiro “a” deve ser subtraído de um. Caso “dia” seja negativo, deve-se somar 7 ao seu resultado