Algoritmos e Lógica de Programação

80 horas // 4 h/semana

Estrutura de Repetição (for)

Aula 07 – parte 1

Prof. Piva

Para começar...

- Os comandos de repetição dão mais movimento ao nosso programa. Permitem que uma ação seja executada mais de uma vez sem que tenhamos que executar novamente o programa. Podemos testar entradas de dados e pedir que o usuário repita a entrada até que um valor válido seja digitado.

- Imagine a seguinte situação... Você tem que desenvolver um algoritmo que recebe como entrada 10 números inteiros (idade das pessoas que trabalham com você). Sua tarefa é calcular a idade média desses seus colegas de trabalho. Como seria esse algoritmo?

Algoritmo para calcular a idade média – 10 pessoas

```
algoritmo “idade média"
var
  idade1, idade2, idade3, idade4, ... idade10: inteiro
  media: real
inicio
      escreva("Digite a idade 1: ")
      leia(idade1)
      escreva("Digite a idade 2: ")
      leia(idade2)
      ...
      escreva("Digite a idade 10: ")
      leia(idade10)
      media <- (idade1+idade2+...+idade10)/10
      escreval ()
      escreva (“A media de idade e: “, media)
fimalgoritmo
```

Ufaaaa!!!

- Teria uma forma mais **EFICIENTE** de fazer isso?

Simmmm!!!

- Usando comandos que controlam repetições (ou comandos de repetição).

- Vamos ver um deles hoje:
- Comando **PARA (for)**

Comando PARA (for)

Este comando é útil quando se deseja repetir um número fixo de vezes determinado conjunto de comandos.

Possui a **mesma** lógica de funcionamento nas principais linguagens de programação

Para isso deverá ter uma **variável de controle**, **um valor inicial**, **um valor final** e o **valor do incremento** – passo - que essa variável receberá para sair do valor inicial até atingir o valor final.

Comando PARA (for)

sintaxe:

para *<variável>* **de** *<valor-inicial>* **ate** *<valor-limite>* [**passo** *<incremento>*]
    faca
        <seqüência-de-comandos>
fimpara

análise do comando:

- **para**, **de**, **ate**, **passo**, **faca** e **fimpara** são palavras-chave do comando;
- *variável* : é a variável de controle do comando e deve ser do tipo inteiro;
- *valor-inicial*: é o valor de início da variável de controle. Pode ser uma constante ou uma expressão aritmética, desde que o valor seja do tipo inteiro. A atribuição desse valor à variável de controle é feita apenas uma vez antes de iniciar a primeira repetição do comando;

Comando PARA (for)

análise do comando (continuação):

- *valor-limite*: é o valor de parada do comando. Isto é, a sequência de comandos será executada até que a variável de controle atinja esse valor;

- *incremento*: é o valor do passo que a variável de controle deve "caminhar" para que atinja o valor limite. É opcional. E, quando presente, é necessário apresentar o valor do incremento que será acrescentado à variável de controle após cada repetição da sequência de comandos. Quando não está presente não deverá especificado o valor do incremento e compilador entenderá que o valor do incremento é o padrão, ou seja, igual a 1 (um).

Comando PARA (for)

Exemplo:

```
para x de 1 ate 3 passo 1 faca
    escreval("X = ", x)
```

Como o passo usado, no exemplo, é igual a 1 ele pode ser omitido. Nesse caso a linha do comando ficaria:

```
para x de 1 ate 3 faca
```

Observação: ***de*** pode ser substituído por dois-pontos e igual (:=)

```
para x := 1 ate 3 faca
```

Voltando ao nosso problema inicial

- Idade média de 10 pessoas...
- Como poderíamos modificar o algoritmo para que fique mais OTIMIZADO?

Algoritmo idade média – 10 pessoas (comando **PARA**)

```
algoritmo “idade média"
var
  idade, soma, i: inteiro
  media: real
inicio
      soma <- 0   // inicializo a variável soma com zero.
      para i := 1 ate 10 faca
            escreva(“Digite a idade “, i, “: ")
            leia(idade)
            soma <- soma + idade
      fimpara
      media <- soma / 10
      escreval ()
      escreva (“A media de idade e: “, media)
fimalgoritmo
```

Outro exemplo

Elaborar um algoritmo que imprime na tela os dez primeiros múltiplos de um número inteiro qualquer fornecido pelo usuário (lido). No final, imprima também a soma destes dez números.

Exemplo da saída:

Valor lido: 3

Lista de Múltiplos: 3 6 9 12 15 18 21 24 27 30

Soma = 165

Outro Exemplo

```
algoritmo "Múltiplos de número lido"
var
  numero, soma, multiplo, i: inteiro
inicio
escreva("Digite um número: ")
leia(numero)
escreval ("Valor Lido: ", numero)
escreva ("Lista de Múltiplos: ")
soma<-0
para i de 1 ate 10 passo 1 faca
   multiplo <- i*numero
   escreva(multiplo, " ")
   soma<- soma+multiplo
fimpara
escreval()
escreval("Soma = ", soma)
fimalgoritmo
```

Estrutura Repetição Algoritmo
Determinada

Forma Geral 1:

PARA <<VAR.DE TIPO INTEIRO>> ← <<VALOR INICIAL>> ATE <<VALOR FINAL>> FAÇA
    <<COMANDO1>>;

Forma Geral 2:

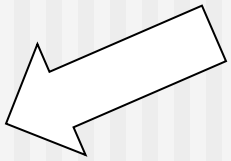
PARA <<VAR. DE TIPO INTEIRO>> ← <<VALOR INICIAL>> ATE <<VALOR FINAL>> FAÇA
    ÍNICIO
        <<COMANDO1>>;
        <<COMANDON>>
    FIM;

Estrutura Repetição em Python
Determinada

Forma Geral em linguagem Python:

```
for i in <<interável>>:
        <<COMANDO1>>
        ...
        <<COMANDON>>
```

INTERÁVEL...
Qualquer elemento que contenha um conjunto de elementos que podem ser selecionados um a um, por meio de interações



Exemplos:
range(1,10)
range(1,10,2)
str
list ...

Estrutura Repetição
Determinada

Exemplo:
Mostrar uma sequência numérica de 1 a 10, com a frase "o número é= n"

```
for i in range(1,11):
      print("O numero eh = ", i)
```

Estrutura Repetição
Determinada

Exemplo:2: Pedir para o usuário digitar 10 números inteiros. Ao final, exibir a média desses números.

```
soma = 0
media = 0
for i in range(1, 11)
      n = int(input("Digite um número inteiro = "))
      soma = soma + n
media = soma/10
print("A media eh = ", media);
```

Estrutura Repetição
Determinada

Faça um programa em Python que escreve os 100 primeiros números pares.

```
par = 0
for i in range(1, 101):
        print(par)
        par = par + 2
```

Comandos: break e continue

```
for i in range(5):
    if i == 0:
        print('\ni = 0, Então: ', i)
    elif i == 1:
        print('\ni = 1, Então: continue’)
        continue
    elif 1 < i < 3:
        print('\nA variável i, é: ', i)
    elif i == 3:
        print('\ni = 3, Então: break’)
        break
    else:
        print('\ni > 3, Então: ', i)
```

Qual a Saída?

Comando else (do for)

- Python fornece a cláusulas break e else para os laços.
- Else - Será executada quando a condição do laço for falsa.
- Break – sai do laço

```
for elemento in lista :
    if elemento == parada :
            break
    print elemento
else :
    print " Laço chegou ao fim “
```

- No exemplo acima, a mensagem **“Laço chegou ao fim”** só é impressa caso não exista um elemento que seja igual a **“parada”.**

EXERCÍCIOS

Algoritmos de repetição

EXERCÍCIO 1

Faça um algoritmo que leia um valor N inteiro e positivo, calcule e mostre o valor E, conforme a fórmula a seguir.

E = (2 ** 1) + (2 ** 2) + (2 ** 3) + ... + (2 ** N)

EXERCÍCIO 2

Faça um algoritmo que calcule a soma dos primeiros 50 números pares. Este algoritmo não recebe valores do teclado.
Os primeiros números pares são 2, 4, 6...

EXERCÍCIO 3

Faça um algoritmo que leia o valor do peso e da altura de 20 pessoas. Ao final, o algoritmo deve mostrar:

- **O peso médio**
- **A altura média**
- **O maior e o menor IMC**

Obs: IMC (Índice de Massa Corporal) – calculado a partir da fórmula:

$$\mathrm{IMC} = \frac{\mathrm{massa}}{(\mathrm{altura} \cdot \mathrm{altura})}$$

EXERCÍCIO 4

Construa um algoritmo que calcule a média aritmética de um conjunto de números pares que forem fornecidos pelo usuário. O valor de finalização será a entrada do número 0. Observe que nada impede que o usuário forneça quantos números ímpares quiser, com a ressalva de que eles não poderão ser acumulados.

EXERCÍCIO 5

Elabore um algoritmo que simule uma contagem regressiva de 10 minutos, ou seja, mostre 10:00 e então 9:59, 9:58, ..., 9:00; 8:59, 8:58, até 0:00.

EXERCÍCIO 6

Faça um programa que receba um número inteiro x. Calcule e mostre o fatorial desse número (x!).

EXERCÍCIO 7

Faça um programa que calcule os 10 primeiros números da sequencia de Fibonacci

EXERCÍCIO 8

Faça um programa que receba um número inteiro maior que 1. Ele deve verificar se o número fornecido é primo ou não, e mostrar a mensagem correspondente.

Lembre-se: um número primo só é divisível por 1 ou por ele mesmo.

Algoritmos e Lógica de Programação

80 horas // 4 h/semana

Estrutura de Repetição

Aula 07 – parte 2

Prof. Piva

Para começar...

- Existem situações onde não sabemos, ao certo, **quantas vezes teremos que repetir** a sequencia de entradas ou comandos.
- Quando isso ocorre, o comando PARA, passa a ser não mais uma boa opção.

Felizmente, temos outro comando de repetição !!

- Comando:
  - **ENQUANTO**

Para começar...

- Os duentes de Papai Noel estão com um problema... Periodicamente (não se sabe se é por semana, ou por mês, ou de quinze em quinze dias... O Sr. Noel pede a informação do número médio de cartas recebidas por dia. Portanto, você tem que ajudá-los com um algoritmo que possa receber uma certa quantidade de valores inteiros (que corresponde a quantidade de cartas recebidas por dia pelo Papai Noel). O algoritmo deve realizar o cálculo, quando o último valor digitado for igual a -1.
- A saída deve informar a quantidade de dias e o número médio de cartas recebidas por dia no período.

Comando while...

- As vezes, temos a necessidade de realizar o teste de REPETIÇÃO antes de executarmos os comandos...
- Nesse caso, podemos utilizar o comando ENQUANTO (while)

Comando ENQUANTO (while)

O comando cuja palavra-chave é **enquanto** (*while*), tem o mesmo comportamento nas várias linguagens, com apenas algumas diferenças na escrita do comando.

De modo geral, é um comando que repete um comando ou um conjunto de comandos **enquanto** uma **condição** (expressão lógica) **for verdadeira**.

Quando essa **condição** se tornar **falsa** o controle passa para o próximo comando que se segue imediatamente ao final do comando repetitivo enquanto.

Comando ENQUANTO (while)

sintaxe:

enquanto *<expressão-lógica>* **faca**
<sequência-de-comandos>
fimenquanto

análise do comando:

- o bloco do comando se inicia com a palavra-chave **enquanto** e termina com o **fimenquanto**;
- o comando utiliza uma outra palavra-chave: **faça**, sem a cedilha, após a *expressão-lógica*;
- a expressão-lógica não precisa estar entre parênteses;

Comando ENQUANTO (while)

Exemplo:

```
i<- 1
enquanto (i<=3) faca
   escreval("I = ", i)
   i<- i+1
fimenquanto
```

Exemplo

Elaborar um algoritmo que imprime na tela os dez primeiros múltiplos de um número inteiro qualquer fornecido pelo usuário (lido). No final, imprima também a soma destes dez números.

Exemplo da saída:

Valor lido: 3

Lista de Múltiplos: 3 6 9 12 15 18 21 24 27 30

Soma = 165

LEMBRE-SE QUE NÓS JÁ FIZEMOS ESSE EXEMPLO COM O COMANDO FOR

Estrutura Repetição
INDETERMINADA COM VALIDAÇÃO INICIAL

É usada para repetir N vezes uma ou mais instruções.

Tendo como vantagem o fato de não ser necessário o conhecimento prévio do número de repetições.

Estrutura Repetição
INDETERMINADA COM VALIDAÇÃO INICIAL

Forma Geral 1:

ENQUANTO <<CONDIÇÃO>> FACA
    <<COMANDO1>>;

Forma Geral 2:

ENQUANTO <<CONDIÇÃO>> FACA
    ÍNICIO
        <<COMANDO1>>;
        <<COMANDON>>
    FIM;

Comando While

```
while <condição>:
    <comandos>
```

Exemplo: Contagem

```
int i=0;
while (i < 10):
        print(i)
        i+=1
```

- O loop se repete, enquanto a condição for verdadeira

Comando While

```
a = 0
b = 2
while a <= b:
        print('\n' , a, ' <= ', b, ' ‘)
        a += 1
```

Comando While

```
i = 0
while True:
        print(i)
        i += 1
```

Comando While

Da mesma forma que o comando FOR com o comando WHILE existe a possibilidade de utilizar BREAK e CONTINUE.

```
from random import randint
while True:
    x = randint(0, 10)
    print(x)
    if x == 5:
        break
```

EXERCÍCIOS

Comandos de Repetição (parte 2)

EXERCÍCIO 1

Faça um algoritmo que leia dois valores b e N inteiros e positivos, calcule e mostre o valor E, conforme a fórmula a seguir.

E = (b ** 1) + (b ** 2) + (b ** 3) + ... + (b ** N)

EXERCÍCIO 2

Faça um algoritmo que calcule a área de um triângulo. Este algoritmo não pode permitir a entrada de dados inválidos, ou seja, medidas menores ou iguais a zero.

EXERCÍCIO 3

Faça um algoritmo que leia um número não determinado de pares de valores [m,n], todos inteiros e positivos, um par de cada vez e que calcule e mostre a soma de todos os números inteiros entre m e n (inclusive). Na digitação dos pares m,n deve-se validar que m é maior que n.

EXERCÍCIO 4

Faça um algoritmo que:

-Leia um número indeterminado de números que representam, cada um, a idade de um indivíduo.

-Para finalizar, o usuário deverá digitar 0, que não entrará nos cálculos.

- Calcule e mostre a idade média e o número total de pessoas deste grupo de indivíduos.

EXERCÍCIO 5

Faça um algoritmo que leia o valor do peso e da altura de 20 pessoas. Ao final, o algoritmo deve mostrar:

- **O peso médio**
- **A altura média**
- **O maior e o menor IMC**

Obs: IMC (Índice de Massa Corporal) – calculado a partir da fórmula:

$$\mathrm{IMC} = \frac{\mathrm{massa}}{(\mathrm{altura} \cdot \mathrm{altura})}$$

EXERCÍCIO 6

Faça um programa que receba um número inteiro x. Calcule e mostre o fatorial desse número (x!).

EXERCÍCIO 7

Faça um programa que calcule os 10 primeiros números da sequencia de Fibonacci

EXERCÍCIO 8

Faça um programa que receba um número inteiro maior que 1. Ele deve verificar se o número fornecido é primo ou não, e mostrar a mensagem correspondente.

Lembre-se: um número primo só é divisível por 1 ou por ele mesmo.