



Programação de Computadores

- ✓ Estruturas de repetição

Conceitos abordados nesta aula

A proposta desta aula é apresentar as estruturas de repetição (looping).



Exemplo: Tabuada

1- Escreva um programa em Python que solicite ao usuário um número, calcule e mostre a tabuada desse número.



```
#Exemplo1
num = int(input('Digite um número inteiro: '))

print(f'{num} * 1 = {num*1}')
print(f'{num} * 2 = {num*2}')
print(f'{num} * 3 = {num*3}')
print(f'{num} * 4 = {num*4}')
print(f'{num} * 5 = {num*5}')
print(f'{num} * 6 = {num*6}')
print(f'{num} * 7 = {num*7}')
print(f'{num} * 8 = {num*8}')
print(f'{num} * 9 = {num*9}')
print(f'{num} * 10 = {num*10}')
```

Estruturas de repetição

- ✓ Também conhecidas como laços (loop, cycle, iteration, ...);
- ✓ Utilizadas para executar repetidamente uma instrução ou um bloco de instruções enquanto determinada condição for verdadeira.
- ✓ São três:



- ✓ para
- ✓ enquanto
- ✓ faça... enquanto



- ✓ for
- ✓ while

Estruturas de repetição

Para determinarmos qual é a estrutura mais adequada, devemos saber:

- ✔ o número de vezes que o trecho programa vai ser executado: laços contados
- ✔ ou a condição para que ela aconteça: laços condicionais

Laços contados: um contador irá auxiliar no laço. Neste laço, a repetição da estrutura repete-se até que o contador atinja o limite estipulado na condição.

Laços condicionais: o valor é desconhecido e devemos utilizar uma variável com valor predefinido em uma condição dentro do laço para finalizarmos a repetição.



Estruturas de repetição



Independente do tipo de laço, todos são constituídos de três partes:

- Inicialização(ões) da(s) variável(is) de controle.
- Condição(ões).
- Atualização da(s) variável(is) de controle.



Estrutura para (for)

É compacta, pois a inicialização, condição e atualização estão reunidos na declaração do laço.

Algoritmo

```
para(inicialização; condição; atualização)
{
    bloco de instruções
}
```



Python

```
for variável in lista:
    bloco de instruções
```

Instruções do bloco devem ser indentadas corretamente



O comando for na linguagem Python

Diferente de outras linguagens de programação, o comando for de Python itera sobre os itens de uma sequência (uma lista ou uma string), na ordem em que aparecem na sequência. Veja o exemplo a seguir:

```
#Exemplo simples do comando for  
animais = ['gato', 'cachorro', 'leão', 'camelo']  
for a in animais:  
    print(a)
```

```
gato  
cachorro  
leão  
camelo
```

```
sequencia = [0, 1, 2, 4, 4, 5]  
for num in sequencia:  
    print(num)
```

```
0  
1  
2  
4  
4  
5
```



O comando for na linguagem Python

Diferente de outras linguagens de programação, o comando for de Python itera sobre os itens de uma sequência (uma lista ou uma string), na ordem em que aparecem na sequência. Veja o exemplo a seguir:

```
▶ nome = 'aline'
  for letra in nome:
    print(letra)
```

```
↳ a
   l
   i
   n
   e
```



O comando for na linguagem Python - função range

- ✓ A função **range()** retorna uma série de números consecutivos. Por padrão, ela inicia no número 0 e é incrementada adicionando 1.
- ✓ O comando **range(4)**, por exemplo, retornará o seguinte valor : “0, 1, 2, 3”, pois ao chegar ao número 4, o loop será concluído.
- ✓ A sintaxe da função range() é:

```
range(início, parada, incremento)
```



- **início**: valor opcional e corresponde a partir de qual número o range será iniciado;
- **parada**: valor obrigatório e indica o número de parada do range;
- **incremento**: opcional e indica o valor que queremos adicionar entre um item e outro.

O comando for na linguagem Python - função range

A função range é útil para iterar sobre sequências numéricas, porque gera progressões aritméticas.

```
for i in range(10):  
    print(i, end=" ")
```

```
↳ 0 1 2 3 4 5 6 7 8 9
```

```
for i in range(3, 8):  
    print(i, end=" ")
```

```
↳ 3 4 5 6 7
```

```
for i in range(0, 21, 2):  
    print(i, end=" ")
```

```
↳ 0 2 4 6 8 10 12 14 16 18 20
```

```
for i in range(5, 20, 3):  
    print(i, end=" ")
```

```
↳ 5 8 11 14 17
```



Exemplo: Tabuada

1- Escreva um algoritmo que solicite ao usuário um número, calcule e mostre a tabuada desse número.

```
algoritmo
inicio
    inteiro numero, i
    escreva("Entre com o número:")
    leia(numero)
    para (i=0; i<=10; i++) {
        escreva ( numero * i )
    }
fim
```

Teste de mesa:

Por exemplo: numero = 3

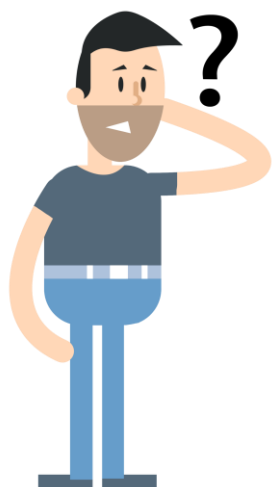


i	Saída
0	0
1	3
2	6
3	9
4	12
5	15
6	18
7	21
8	24
9	27
10	30

Exemplo: Tabuada

1- Escreva um algoritmo que solicite ao usuário um número, calcule e mostre a tabuada desse número.

```
main.py
1 num = int(input("Digite um número inteiro: "))
2 for i in range(11):
3     print("%d * %d = %d" %(num, i, num*i))
```



- ✓ Por que não precisamos iniciar o range()?
- ✓ Por que termina em 11??

Exemplo

2- Faça um programa em Python que solicite ao usuário 10 números reais, calcule e mostre a soma dos números digitados. Use a estrutura de repetição para (**for**).

```
main.py
1 soma = 0
2 for i in range(10):
3     num = float(input("Digite um número real "))
4     soma = soma + num
5
6 print ("O resultado da soma é: ", soma)
```

Acumulador



se
Liga
Ai

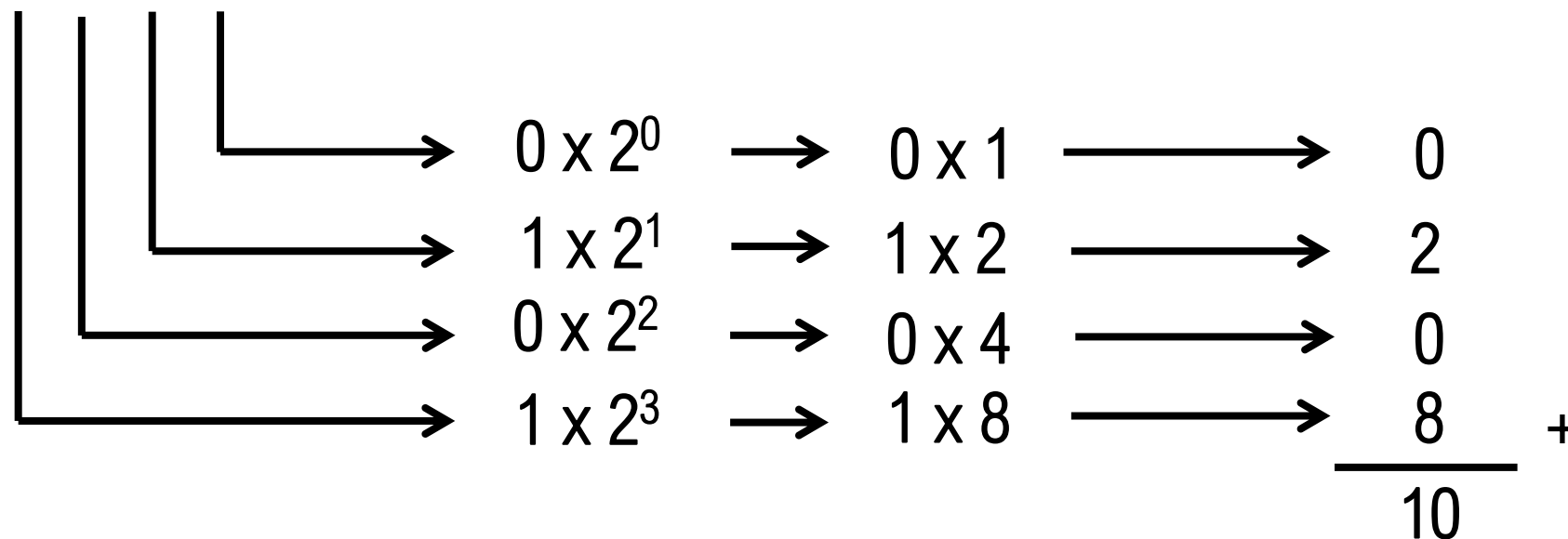
Observe que a variável soma funciona como um acumulador!!



Exemplo

3- Faça um programa em Python que solicite um número binário, faça a conversão e exiba o número digitado na base decimal. Use a estrutura de repetição para (**for**).

1010



$$1010_2 = 10_{10}$$



Exemplo

3- Faça um programa em Python que solicite um número binário, faça a conversão e exiba o número digitado na base decimal. Use a estrutura de repetição para (**for**).

bin2dec(v1).py > ...

```
1 num = input("Digite um número binário: ")
2 n = len(num) - 1
3 decimal = 0
4
5 for d in num:
6     decimal = decimal + int(d)*2**n
7     n = n - 1
8
9 print("O binário %s equivale a %d na base decimal." %(num,decimal))
```



Estrutura enquanto (while)

Estrutura utilizada tanto para **laços contados** quanto para os **laços condicionais**, possui a seguinte sintaxe:

Algoritmo

```
iniciar a variável de controle  
enquanto (condição for verdadeira)  
{  
    bloco de instruções  
    atualizar a variável de controle  
}
```

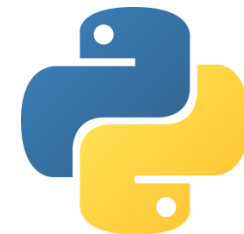


Instrução que modifica o estado de algum elemento utilizado na condição

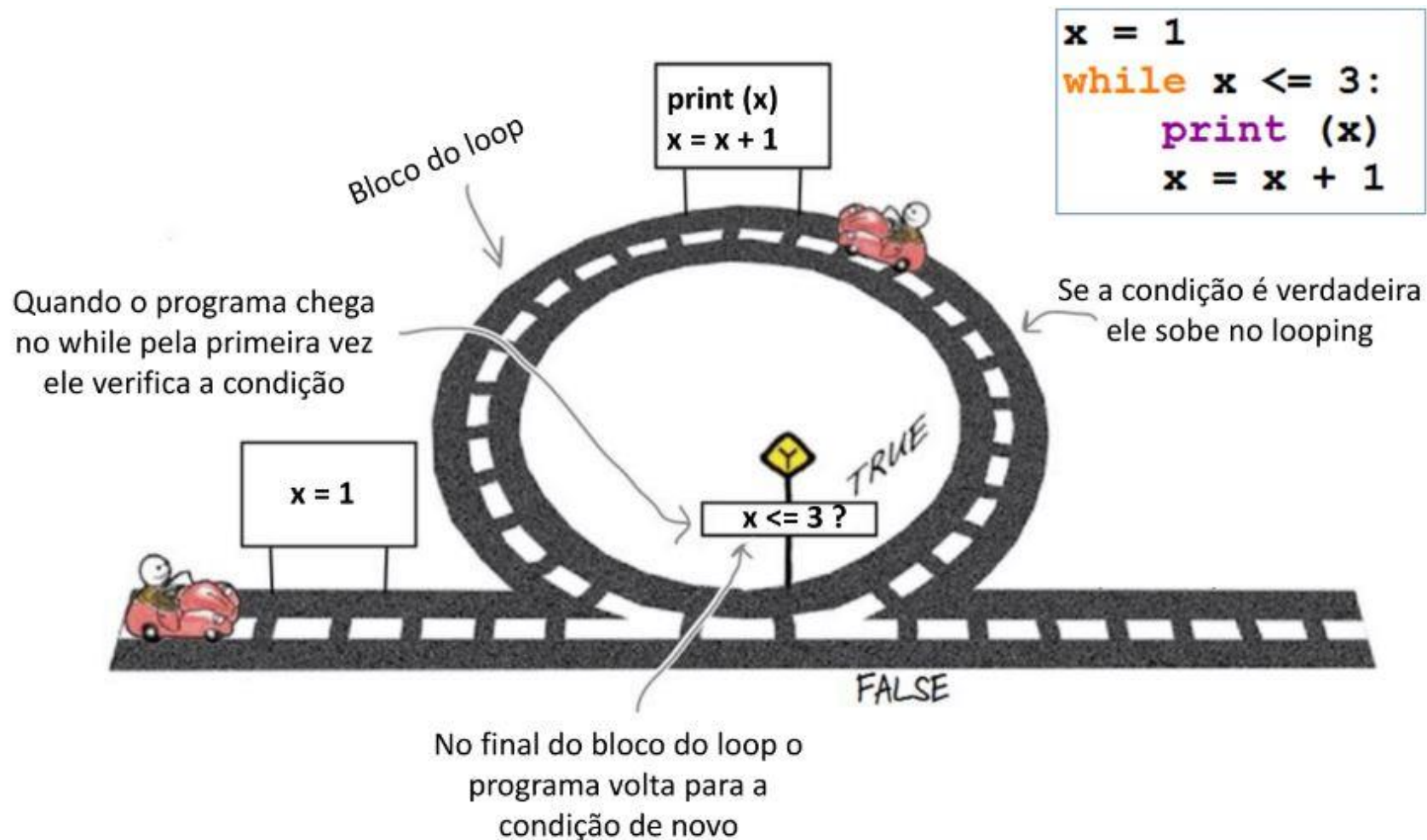
Python

```
iniciar a variável de controle  
while condição:  
    bloco de instruções  
    atualizar variável de controle
```

Instruções do bloco devem ser indentadas corretamente



Comandos para repetições na linguagem Python



Exemplo

4- Faça um programa em Python que calcula e mostra a média entre duas notas de 10 alunos. Use a estrutura de repetição **enquanto**

```
main.py
1 contador = 1
2 while contador <= 10:
3     nota1 = float(input("Digite a primeira nota: "))
4     nota2 = float(input("Digite a segunda nota: "))
5     media = (nota1 + nota2)/2
6     print("A média é %.2f" %media)
7     contador = contador +1
```



se
Liga
Ai

Observe que a variável contador controla o fluxo da estrutura de repetição!!



Pausa para reflexão

4- Faça um programa em Python que calcula e mostra a média entre duas notas de 10 alunos. Use a estrutura de repetição **enquanto**



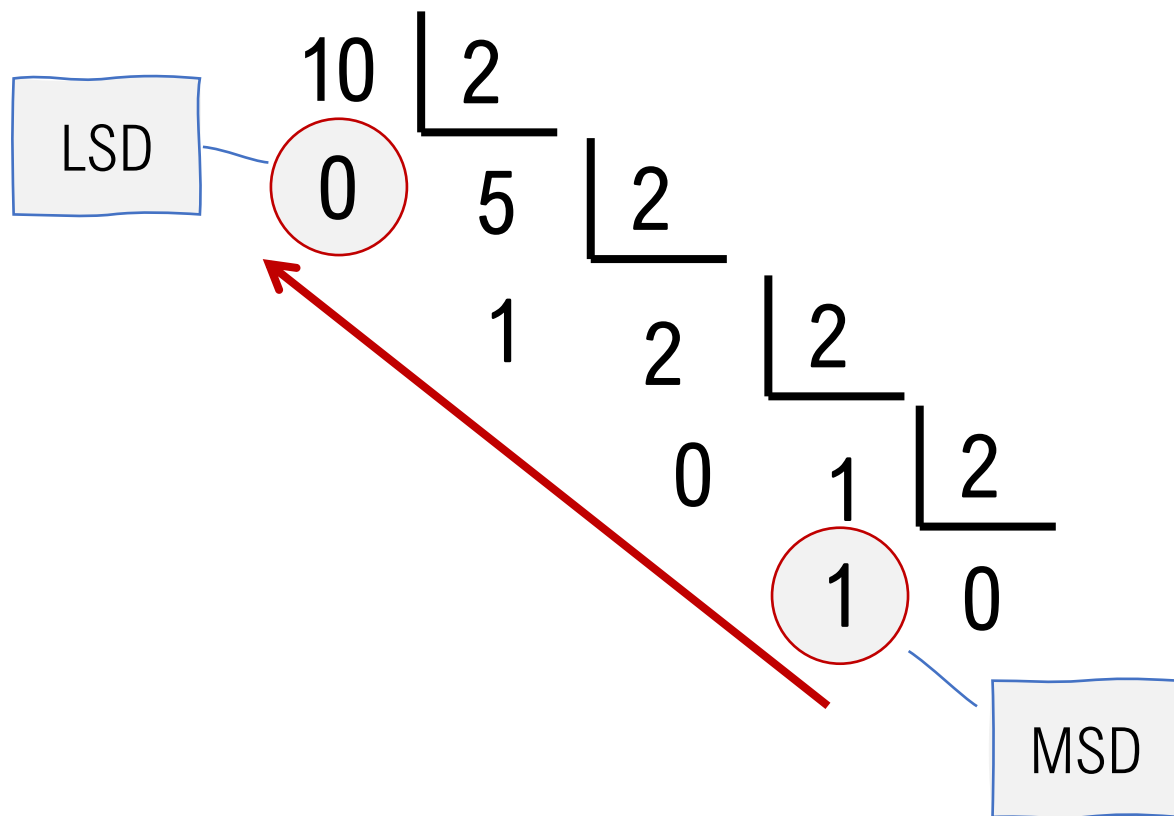
- ✓ Neste exemplo, seria mais fácil utilizar a estrutura para (for)???
- ✓ Por qual motivo???



Pausa para Refletir

Exemplo

5- Faça um programa em Python que solicite um número decimal, faça a conversão e exiba o número digitado na base binária. Use a estrutura de repetição enquanto (**while**).



$$10_{10} = 1010_2$$



Exemplo

5- Faça um programa em Python que solicite um número decimal, faça a conversão e exiba o número digitado na base binária. Use a estrutura de repetição enquanto (**while**).

```
dec2bin.py M x
Python-Projects > dec2bin.py
1  num = int(input("Digite um número inteiro: "))
2
3  binario = ""
4  while num!=0:
5      r = num%2
6      binario = str(r) + binario
7      num = num//2
8
9  print(binario)
```



Exemplo

6- Faça um programa em Python que calcule e mostre a média de uma quantidade indeterminada de números inteiros digitados pelo usuário. Use a estrutura de repetição **enquanto**.

```
main.py
1  contador = 0
2  soma = 0
3  resp = "s"
4
5  while resp == "s" or resp == "S":
6      num = float(input("Digite um número: "))
7      soma = soma + num
8      contador = contador + 1
9      resp = input("Deseja continuar (S/N)? ")
10
11  media = soma / contador
12  print("A média dos números digitados é %.2f" %media)
```

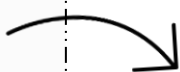
Observe a função da variável *contador* e compare com o exemplo anterior!!!




Controlando o loop na linguagem Python

Python

for variável **in** lista:
bloco de instruções
continue (opcional)
break (opcional)



while condição:
bloco de instruções
continue (opcional)
break (opcional)



- ✓ O comando **continue** pode ser usado para ignorar os comandos e executar a próxima iteração ou passo do laço mais interno.
- ✓ O comando **break** permite sair do ciclo mais interno a qualquer momento.

Controlando o loop na linguagem Python

- ✓ O comando **continue** pode ser usado para ignorar os comandos e executar a próxima iteração ou passo do laço mais interno.

```
▶ numeros = [0, 1, 2, 3, 4, 5, 6]
for num in numeros:
    if num%2==0:
        continue
    else:
        print(num)
```

1
3
5



Controlando o loop na linguagem Python

- ✓ O comando **break** permite sair do ciclo mais interno a qualquer momento.

```
▶ animais = ['gato', 'cachorro', 'leão', 'camelo', 'macaco']  
for animal in animais:  
    if animal == 'leão':  
        break  
    print(animal)
```

```
gato  
cachorro
```



Controlando o loop na linguagem Python

- ✓ O comando **break** permite sair do ciclo mais interno a qualquer momento.



Vamos retomar o **Exemplo 6**: Faça um programa em Python que calcule e mostre a média de uma quantidade indeterminada de números inteiros digitados pelo usuário. Use a estrutura de repetição **enquanto**.



Desafio

7- Adivinhe meu número: Crie um jogo onde o computador escolha um número inteiro aleatório entre 0 e 100.

- ✓ Leia a entrada do usuário para tentar acertar o número;
- ✓ Se errar informar ao usuário se o número é maior ou menor;
- ✓ Repetir até o usuário acertar.

Dica: faça a importação da biblioteca random e gere um número aleatório inteiro utilizando a função **randint**(início, fim)

```
from random import *  
num = randint(0,100)  
print(num)
```



DESAFIO

Desafio

```
algoritmo
    inicio
        inteiro num, x, i
        num = rand(20) \\gera um número aleatório entre 0 e 20
        i = 0

        faça{
            i = i + 1
            escreva("Digite um número entre 0 e 20: ")
            leia (x)
            se (x == num)
                escreva("Parabéns, você acertou em " + i + " tentativas")
            senao se (x < num)
                escreva("O número pensado é maior")
            senao
                escreva("O número pensado é menor")
        } enquanto ( x != num)
    fim
```



DESAFIO

desafio

```
main.py
1  from random import *
2  num = randint(0,10)
3  i = 0
4  controle = 0
5
6  while controle == 0:
7      i = i+1
8      x = int(input("Digite um número inteiro: "))
9      if num == x:
10         print("Parabéns, você acertou em %d tentativas" %i)
11         controle = 1
12     elif num > x:
13         print("O número pensado é maior!")
14     else:
15         print("O número pensado é menor!")
```



Alguma dúvida????



Exercícios de aplicação



Observações sobre exercícios

- ✓ Todos os exercícios devem ser resolvidos em Python.
- ✓ O código Python pode ser feito no IDLE, no Repl.it, ou na ferramenta que você ache mais adequada e deve ser salvo um arquivo por exercício com a extensão .py
- ✓ Após finalizar todos os exercícios da aula, coloque-os em uma pasta com o nome da aula, compacte a pasta e envie no Blackboard.



<https://youtu.be/BuxuUbgKwCg>



Exercícios

- 1- Faça um programa em Python que imprima os números pares entre 0 e 100.
- 2- Faça um programa em Python que imprima os números de 1 a 50 de 1 em 1 e de 52 a 100 de 2 em 2.
- 3- Faça um programa em Python que leia um valor n , inteiro e positivo, calcule e mostre a seguinte soma:

$$S = 1 + 1/2 + 1/3 + 1/4 + \dots + 1/n$$

- 4- Escreva um algoritmo que leia um grupo de valores reais e determine quantos valores são positivos e quantos são negativos. Determine, também, qual é o menor desses valores. Utilize o comando de repetição que desejar.

Exercícios

5- Uma pesquisa foi realizada em um grupo de alunos de uma escola. Escreva um programa em Python que leia o sexo e a altura de cada aluno, calcule e mostre a altura média das alunas e dos alunos separadamente. Utilize o comando de repetição que desejar.

Créditos

Esta aula foi elaborada com base no material produzido e cedido gentilmente pelos **Professores Alcides, Lédon, Amilton e Cristiane.**





That's all Folks!