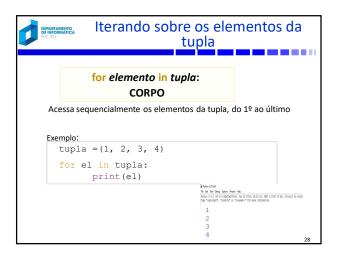
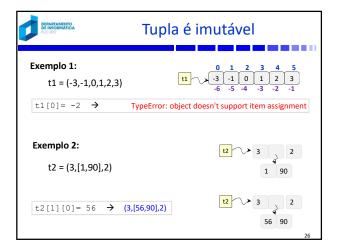
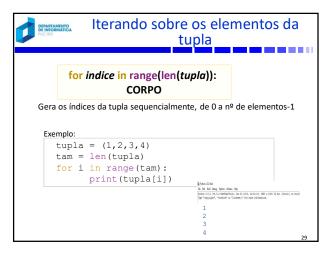
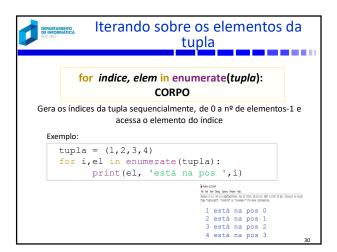


```
Acesso aos elementos da tupla:
                           Indexação
Exemplo 1:
                                    -3 -1 0 1 2 3
   tX = (-3, -1, 0, 1, 2, 3)
tam = len(tX)
print(tX[2])
                         0
print(tX[9-8])
                         -1
print(tX[-2])
                        3
print(tX[tam-1])
                    \rightarrow
print(tX[-1])
                         3
                         IndexError: tuple index out of range
print(tX[tam])
```









```
        tupla1+tupla2

        Cria uma nova tupla com os elementos da tupla1 seguidos dos elementos da tupla2

        Exemplo:

        T1 = (1, 2, 3, 4)

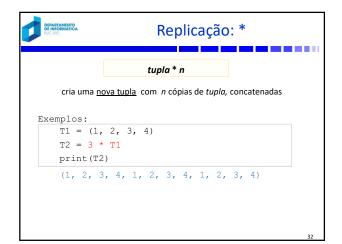
        T2 = (-2, -1, 0, 1, 2, 3, 4, 5)

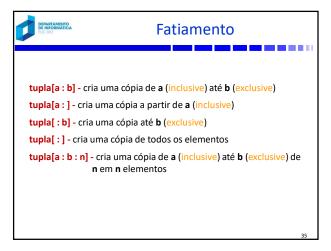
        T3 = T1 + T2

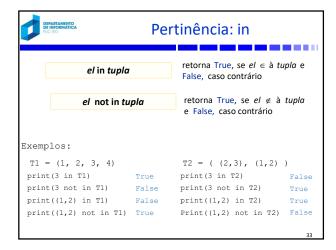
        print (T3)

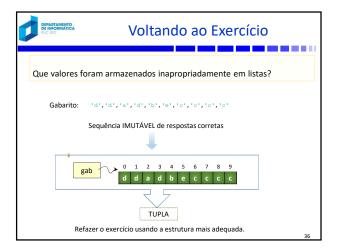
        (1, 2, 3, 4, -2, -1, 0, 1, 2, 3, 4, 5)
```

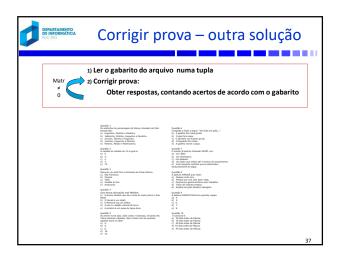


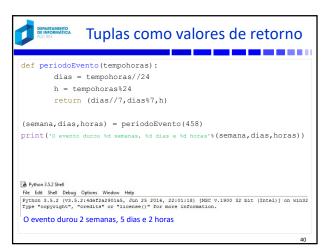


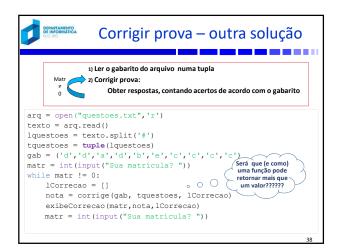


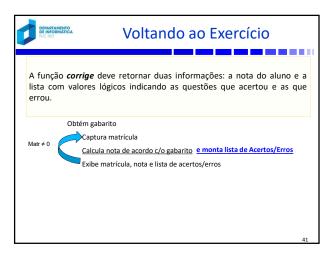


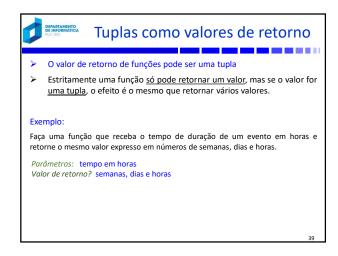


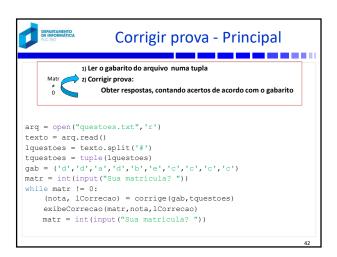




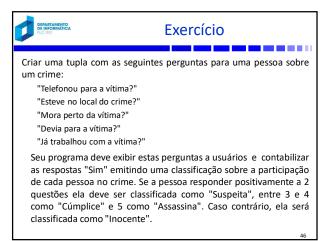






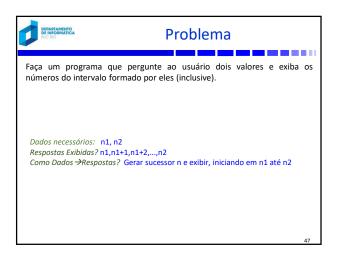


```
Revisão:
                           Corrigir prova
def corrige(gabarito,tquestoes):
   nota = (
   for i,el in enumerate(tquestoes):
       print(el)
       resp = input("Digite a resposta da questão %d: "%(i+1))
       if resp == gabarito[i]:
           nota+=1
           lista.append(True)
       else:
          lista.append(False)
   return (nota, lista)
def exibeCorrecao(matr,nota,Lista):
   print("%d - Sua nota é: %d\n"%(matr,nota))
   print(Lista)
```



Revisão:
Corrigir prova — Nova versão

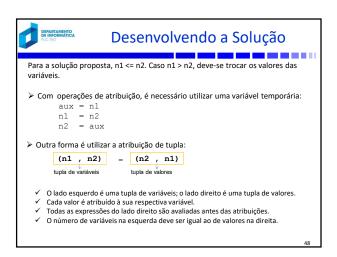
Alterar o programa para que a tupla retornada pela função corrige contenha a nota e as respostas do aluno.



Revisão:
Corrigir prova

def corrige(gabarito,tquestoes):
 nota = 0
 lista = []
 for i,el in enumerate(tquestoes):
 print(el)
 resp = input("Digite a resposta da questão %d: "%(i+1))
 if resp == gabarito[i]:
 nota+=1
 lista.append(resp)
 return (nota, lista)

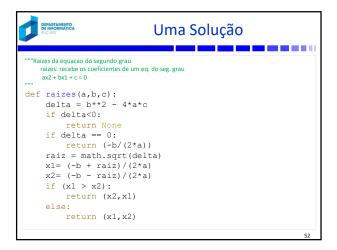
def exibeCorrecao(matr,nota,Lista):
 print("%d - Sua nota é: %d\n"%(matr,nota)
 print(Lista)

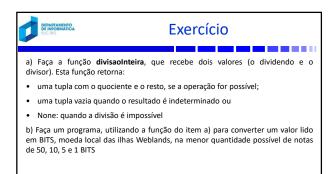


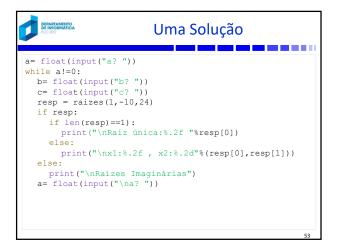
```
DEMARMANDA

no. BODAL MARINA

no. BODAL MARINA
```







Desenvolva a função **raízes**, que calcula e retorna as raízes **x1** e **x2** de uma equação do segundo grau, do tipo ax² + bx + c = 0. **a, b e c** representam os coeficientes da equação.

Observações:

a) Se as raízes forem reais e distintas, a função deve retornar os dois valores e **x1** deve ser associado ao menor valor e **x2** ao maior valor.

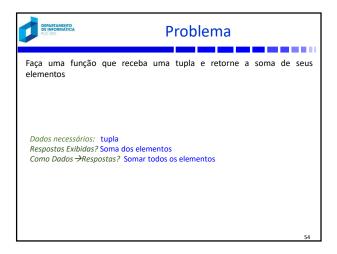
b) Se as raízes forem reais e iguais, a função deve retornar apenas um valor

c) Se não existirem raízes reais, a função deve retornar None.

Faça um programa completo que pergunte um número indeterminado de coeficientes **a, b** e **c** ao usuário (o programa deverá terminar quando o valor do coeficiente de **a** for **zero**) e, para cada grupo de coeficientes, chame a

função *raízes* e as exiba, caso elas sejam reais, ou a mensagem **Raízes** 

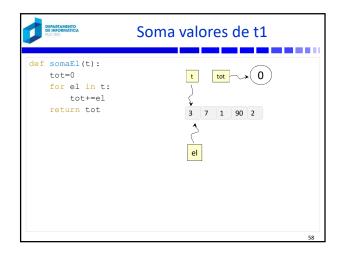
Imaginárias, caso elas sejam imaginárias.

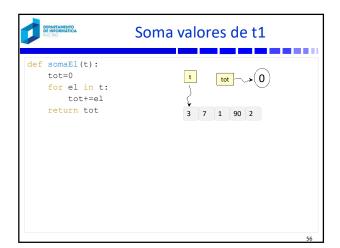


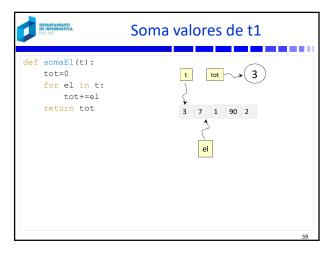
```
Soma valores de uma tupla

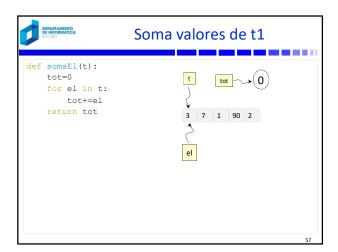
def somaEl(t):
    tot=0
    for el in t:
        tot+=el #tot=tot+el
    return tot

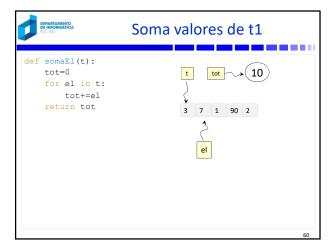
t1 = (3,7,1,90,2)
t2 = (3,(1,2,98),10)
print(somaEl(t1))
print(somaEl(t2))
```

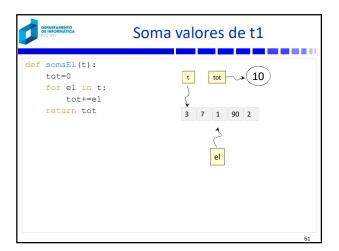


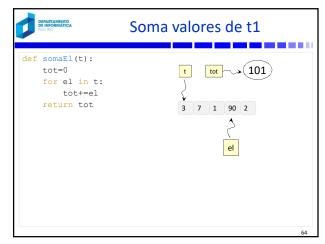


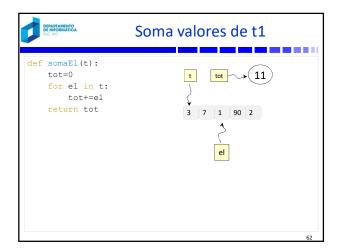


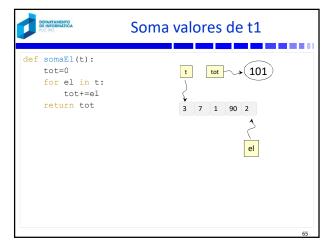


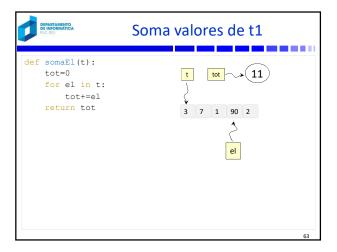


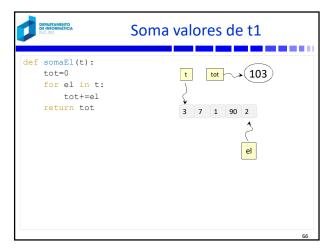


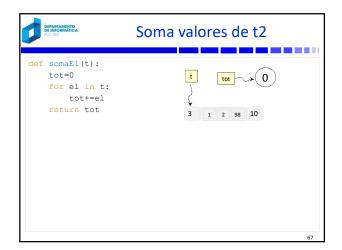


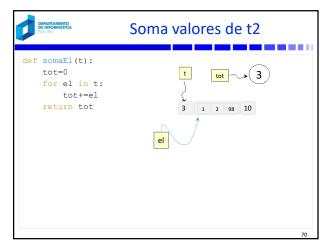


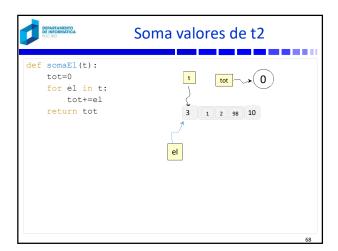


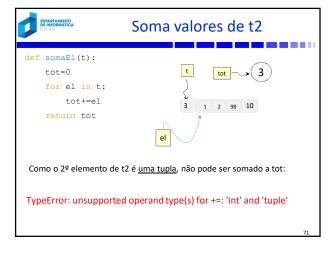


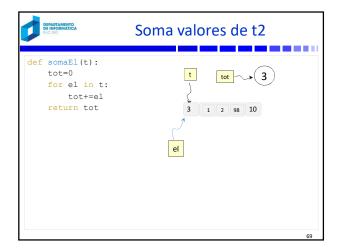


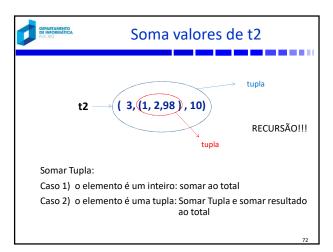












```
Soma elementos de uma tupla

def somaEl(t):
    tot=0
    for el in t:
        if isinstance(el,int):
            tot+=el
        else:
            tot+=somaEl(el)
    return tot

t1=(3,7,1,90,2)
t2=(3,(1,2,98),10)
print(somaEl(t1))
print(somaEl(t2))
```

```
| Uma Solução
| def combina(11, 12):
| if len(11)==0:
| return list()
| lresp = [(11[0],12[0])]+ combina(11[1:], 12[1:])
| return lresp
| print(combina([1,2,3], ['a','b','c']))
```

