

Aluno: Gilberand Barros Marçalma

Cada questão da prova vale 2,0 pontos. Você pode resolver quantas quiser dentro do tempo de 80 minutos, todas em Python.

Assunto da prova: laços de repetição, vetores e matrizes.

1. Faça um programa de computador que imprime todos os números que iniciam pelo dígito 3 ou terminam pelo dígito 3, de 1 a 1000. Você vai imprimir, portanto: 3, 13, 23, 33 ... 300, 301, ... até 993.

30

2. Faça um programa de computador que imprime todos os divisores de 4 até todos os divisores de 44. Em outros termos, você deve imprimir os divisores de 4 (1, 2, 4), depois os divisores de 5 (1, 5), depois os divisores de 6 (1, 2, 3, 6), e assim por diante até imprimir os divisores de 44.

1) `for i in range(1, 1001):`
 `if "3" in str(i) or "3" in i[:len(i)-1]:`
 `print(i)`

20 → `if "3" in str(i) or "3" in i[:len(i)-1]:` → Não existe índice em ~~vetores~~ inteiros,
→ só em vetores ou strings.
Pra isso → converter i para string.

2) `divisores = []`
`for i in range(4, 45):`
 `for j in range(1, i+1):`
 `if i % j == 0:`
 `divisores.append(j)`
 ← `print(divisores)`

Indentação errada. Você quer imprimir o vetor divisores apenas ao final, depois do primeiro for

3. Faça um programa que cria um vetor com 100 inteiros, que devem ser os números pares, primeiro o 846, depois 848 e assim por diante até o centésimo inteiro par.

4. Crie a função `encontrarProduto`, que tem como parâmetro um vetor `v`, com 100 inteiros e um inteiro `alvo`. A função deve retornar um valor lógico. Para retornar verdadeiro (True), o inteiro `alvo` deve ser encontrado como o produto de dois inteiros do vetor `v`. Caso contrário, a função deve retornar falso (False).

```
def encontrarProduto(v:List, alvo:int) -> bool:
```

3) `int_pares = []`

```
for i in range(846, 947):
    if i % 2 == 0:
        int_pares.append(i)
```

846 a 946 tem 101 inteiros pares. :-)

4) `def encontrarProduto(v:List, alvo:int) -> bool:`

```
for i in v: ~~~~~ range(len(v))
```

05 → ~~for j in range(len(v)-1):~~
→ ~~if v[i] * v[j] == alvo:~~

```
for j in range(len(v)-1): ~~~~~ não está ok
    if v[i] * v[j] == alvo:
        return True
    else:
        return False.
```

→ esse return False faz você retornar false logo na primeira vez que a multiplicação não der certo.

Aluno: Guilherme B. Moreira

5. Uma matriz booleana 19X19 representa o tabuleiro do jogo Go. Cada posição da matriz está preenchida com uma peça branca ou preta. A matriz possui True para representar uma peça preta ou False para representar uma peça branca. Vence o jogo a peça que houver em maior número na matriz. Sabendo disso, crie uma função **vencedorDoGo**, que recebe como parâmetro uma matriz booleana 19x19 chamada **tabuleiro** e deve retornar True se a maioria das peças for preta ou False se a maioria das peças for branca.

```
def vencedorDoGo(tabuleiro) -> bool:
```

6. Em um programa de processamento de imagens, uma matriz 1000x1000 de inteiros representa cada um dos pixels de uma imagem. Isso significa que cada posição da matriz representa a cor de um pixel, onde a primeira linha é a primeira linha da imagem e assim sucessivamente, até a última linha da imagem. Sua tarefa é procurar um padrão 8x8 nessa imagem, o que também é representado por uma matriz de inteiros chamada **padrao**, que é 8x8. Deve retornar quantas vezes você encontra o padrão, onde vai retornar zero se não encontrar nenhuma vez. Implemente a função **procurarPadrao**, que deve receber como parâmetro uma matriz **imagem** e outra matriz **padrao**. A função deve retornar quantas vezes o **padrao** está presente integralmente na **imagem**, na mesma disposição linha por linha.

```
def procurarPadrao(imagem, padrao) -> int:
```

5) # T rue = preto, falso = branco. Força, Padawan! Sorte é para os despreparados.

def vencedorDoGo(tabuleiro) -> bool:
 n-pretos=0 *
 n-brancos=0
 for i in tabuleiro:
 for j in tabuleiro:
 if tabuleiro[i][j] == True:
 n-pretos+=1
 else:
 n-brancos+=1
 if n-pretos > n-brancos:
 return True
 else:
 return False