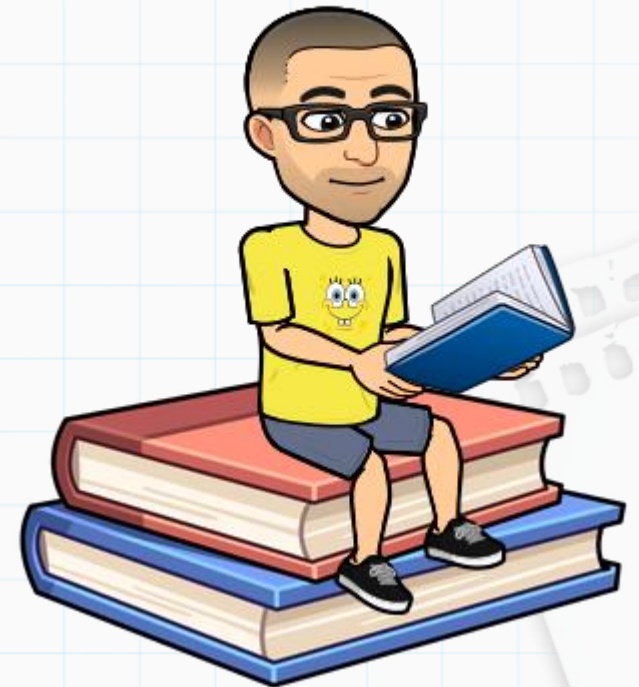


Programação De Computadores

Professor : Yuri Frota

www.ic.uff.br/~yuri/prog.html

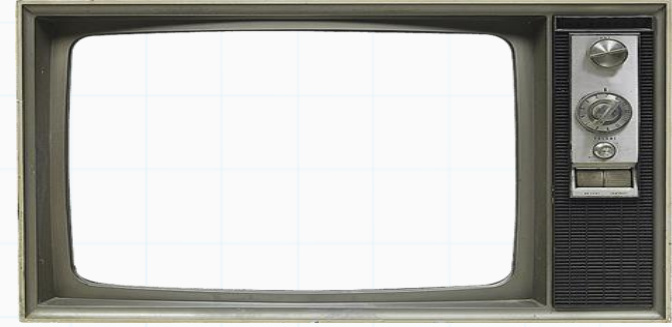
yuri@ic.uff.br



Matrizes

Suponha por exemplo que devemos armazenar as notas de cada aluno(a) de Prog1. Assumindo que um(a) aluno(a) é avaliado(a) com 3 notas (P1,P2 e VS), seria necessário um vetor de 3 posições para guardar as notas de um(a) aluno(a).

0	1	2
6.5	4	8
notas		



Matrizes

Suponha por exemplo que devemos armazenar as notas de cada aluno(a) de Prog1. Assumindo que um(a) aluno(a) é avaliado(a) com 3 notas (P1,P2 e VS), seria necessário um vetor de 3 posições para guardar as notas de um(a) aluno(a).

0	1	2
6.5	4	8

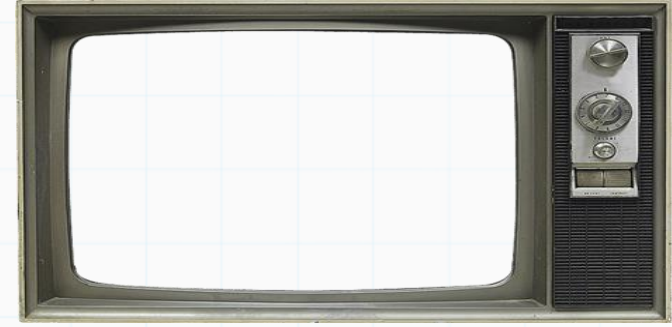
notas

Contudo, assumindo que uma turma tem 40 aluno(a)s, seria necessário uma matriz bidimensional para guardar as notas de todo(a)s o(a)s aluno(a)s de uma turma.

	notas		
	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

alunos

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
```



Matrizes

Suponha por exemplo que devemos armazenar as notas de cada aluno(a) de Prog1. Assumindo que um(a) aluno(a) é avaliado(a) com 3 notas (P1,P2 e VS), seria necessário um vetor de 3 posições para guardar as notas de um(a) aluno(a).

0	1	2
6.5	4	8

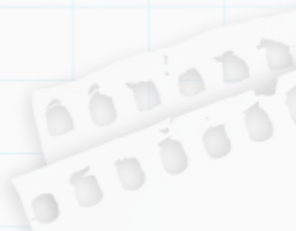
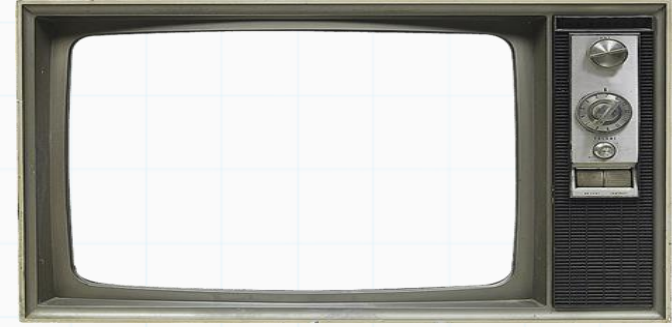
notas

Contudo, assumindo que uma turma tem 40 aluno(a)s, seria necessário uma matriz bidimensional para guardar as notas de todo(a)s o(a)s aluno(a)s de uma turma.

	notas		
	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

alunos

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
```



Matrizes

Acesso:

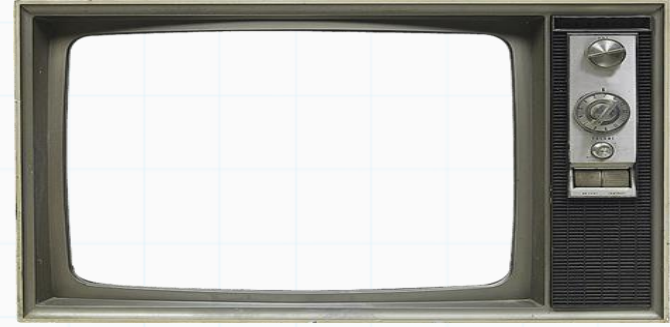
```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
2
3 print(turma[0][1])
4 print(turma[2][2])
```

Shell ×

Python 3.7.7 (bundled)

```
>>> %Run teste.py
```

```
4
10
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Acesso:

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
2 print(turma[0][5])
3
```

Shell x

>>>

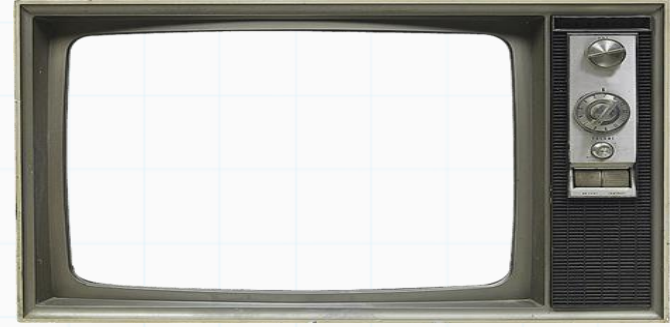
>>> %Run teste.py

Traceback (most recent call last):

File "C:\Users\Yuri\Desktop\teste.py", line 2, in <module>

print(turma[0][5])

IndexError: list index out of range



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

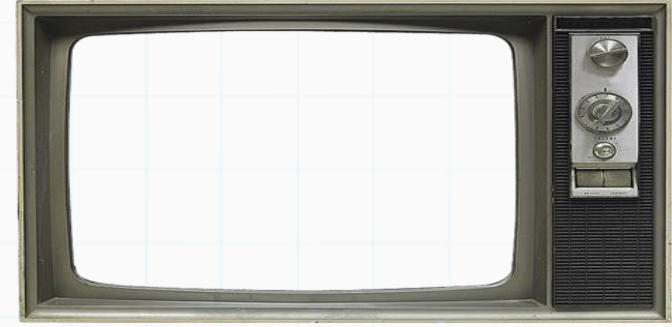
Acesso: Exemplo de calculo da média das notas

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
2
3 media=0
4 for i in range(4): # linhas
5     for j in range(3): # colunas
6         media = media + turma[i][j]
7 media = media / 12
8 print(media)
~
```

Shell x

```
>>> %Run teste.py
```

```
6.699999999999999
```



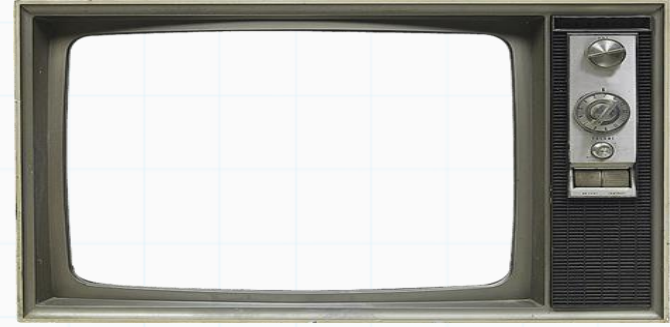
	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Inicialização:

- Direta

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Inicialização:

- Direta

```
1 turma = [[6.5, 4, 8], [7.5, 8.1, 9], [4.4, 7, 10], [3, 9.6, 3.3]]
```

- Indireta

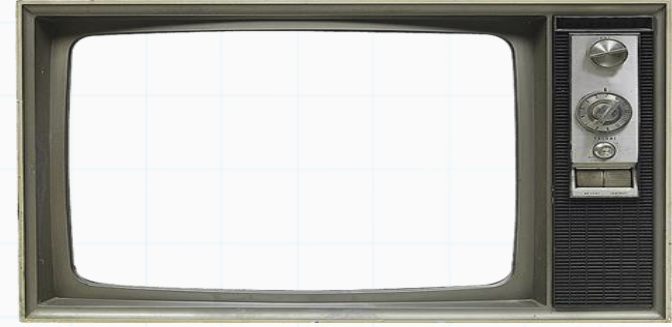
```
1 turma = []
2 for i in range(4):           # para cada linhas
3     linha = []               # cria linha vazia
4     for j in range(3):       # adiciona colunas na linha
5         elem = float(input("Nota "+str(j)+" do aluno "+str(i)+" "))
6         linha.append(elem)
7     turma.append(linha)      # adiciona linha na matriz
```

	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

```
>>> %Run teste.py
```

```
Nota 0 do aluno 0) 6.5
Nota 1 do aluno 0) 4
Nota 2 do aluno 0) 8
Nota 0 do aluno 1) 7.5
Nota 1 do aluno 1) 8.1
Nota 2 do aluno 1) 9
Nota 0 do aluno 2) 4.4
Nota 1 do aluno 2) 7
Nota 2 do aluno 2) 10
```

[código](#)

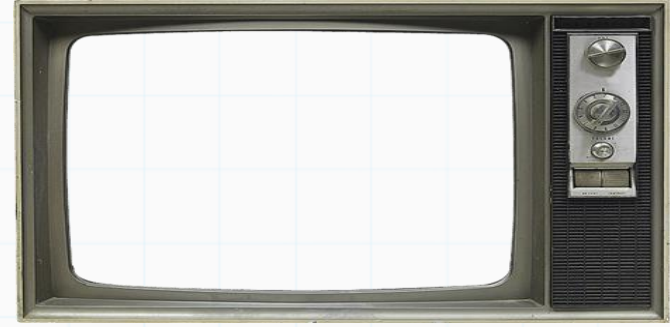


Matrizes

Inicialização:

- Indireta de zeros

```
1 turma = []  
2 for i in range(4):           # para cada linhas  
3     linha = []               # cria linha vazia  
4     for j in range(3):       # adiciona colunas na linha  
5         linha.append(0)  
6     turma.append(linha)      # adiciona linha na matriz
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

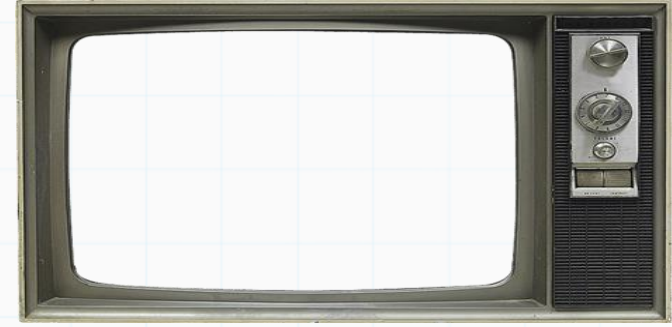
Inicialização:

- Indireta de zeros

```
1 turma = []
2 for i in range(4):           # para cada linhas
3     linha = []               # cria linha vazia
4     for j in range(3):       # adiciona colunas na linha
5         linha.append(0)
6     turma.append(linha)      # adiciona linha na matriz
```

- Ou

```
1 turma = []
2 for i in range(4):           # para cada linhas
3     linha = [0]*3             # cria linha vazia
4     turma.append(linha)      # adiciona linha na matriz
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Inicialização:

- Indireta de zeros

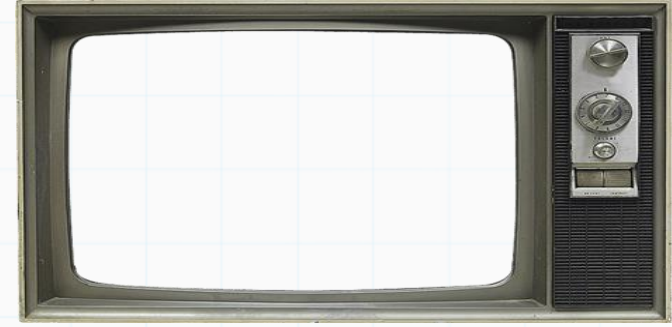
```
1 turma = []
2 for i in range(4):           # para cada linhas
3     linha = []               # cria linha vazia
4     for j in range(3):       # adiciona colunas na linha
5         linha.append(0)
6     turma.append(linha)      # adiciona linha na matriz
```

- Ou

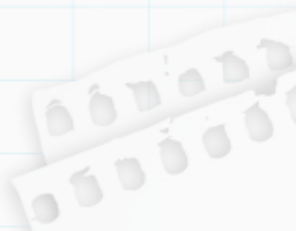
```
1 turma = []
2 for i in range(4):           # para cada linhas
3     linha = [0]*3             # cria linha vazia
4     turma.append(linha)      # adiciona linha na matriz
```

- Ou

```
1 turma = []
2 for i in range(4):           # para cada linhas
3     turma.append([0]*3)      # adiciona linha na matriz
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3



Matrizes

Impressão:

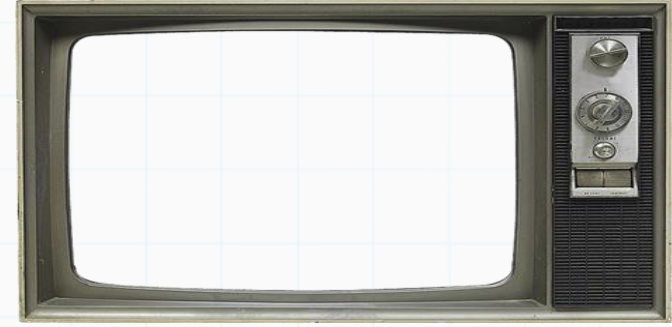
- em linha

```
1 turma = [[1,2,3],[4,5,6],[7,8,9]]  
2 print(turma)
```

Shell x

```
>>> %run teste.py
```

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Impressão:

- em linha

```
1 turma = [[1,2,3],[4,5,6],[7,8,9]]
2 print(turma)
```

Shell x

```
>>> %run teste.py
```

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

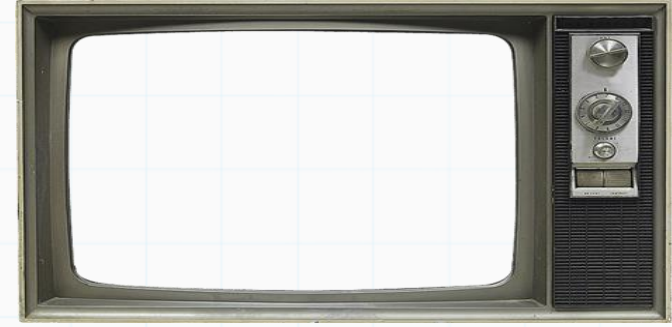
- no formato de matriz

```
1 turma = [[1,2,3],[4,5,6],[7,8,9]]
2
3 for i in range(3):
4     print(turma[i])
```

Shell x

```
>>> %run teste.py
```

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3



Matrizes

Observação:

- Se em listas (vetores) podemos iniciar assim:

```
1 aluno = [0]*4
2 print(aluno)
```

Shell ×

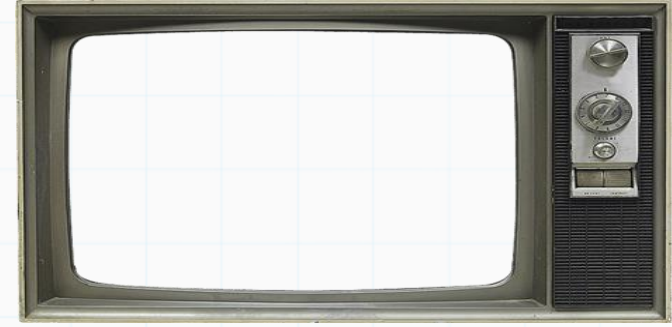
```
[0, 0, 0, 0]
```

- então matrizes podemos fazer isso ?

```
1 turma = [[0]*4]*3
2 print(turma)
```

Shell ×

```
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Observação:

- Se em listas (vetores) podemos iniciar assim:

```
1 aluno = [0]*4
2 print(aluno)
```

Shell ×

```
[0, 0, 0, 0]
```

- então matrizes podemos fazer isso ?

```
1 turma = [[0]*4]*3
2 print(turma)
```

Shell ×

```
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

```
1 turma = [[0]*4]*3
2 turma[0][0]=5
3 print(turma)
```

Shell ×

```
[[5, 0, 0, 0], [5, 0, 0, 0], [5, 0, 0, 0]]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

todas as linhas da matriz
apontam para o mesmo
espaço de memória

Matrizes

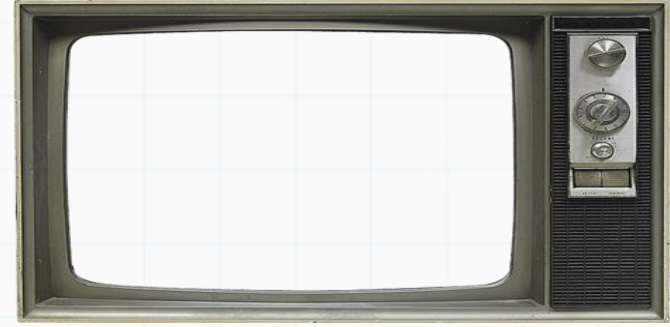
Cópia de Matrizes:

-Tem alocar espaço separado para as duas matrizes

```
1 turma = [[1,2,3],[4,5,6]]
2 turma2 = [[0,0,0],[0,0,0]]
3 for i in range(2):
4     for j in range(3):
5         turma2[i][j] = turma[i][j]
6 print(turma2)
```

Shell ×

```
[[1, 2, 3], [4, 5, 6]]
```



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3

Matrizes

Cópia de Matrizes:

-Tem alocar espaço separado para as duas matrizes

```
1 turma = [[1,2,3],[4,5,6]]
2 turma2 = [[0,0,0],[0,0,0]]
3 for i in range(2):
4     for j in range(3):
5         turma2[i][j] = turma[i][j]
6 print(turma2)
```

Shell ×

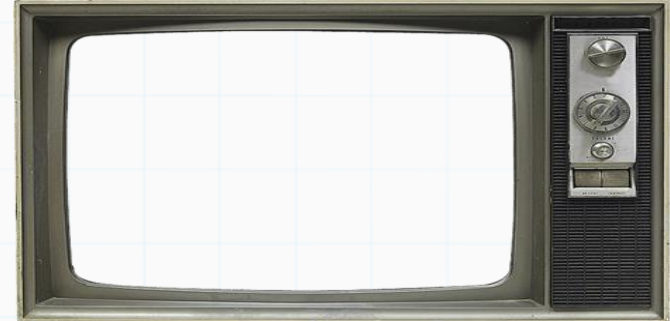
```
[[1, 2, 3], [4, 5, 6]]
```

```
1 turma = [[1,2,3],[4,5,6]]
2 turma2 = []
3 for i in range(2):
4     linha = [0]*3
5     for j in range(3):
6         linha[j] = turma[i][j]
7     turma2.append(linha)
8 print(turma2)
9
```

Shell ×

```
[[1, 2, 3], [4, 5, 6]]
```

- Ou



	0	1	2
0	6.5	4	8
1	7.5	8.1	9
2	4.4	7	10
3	3	9.6	3.3



Matrizes

Matrizes podem ser formada por tipos diferentes:



```
1 GoT = []
2 for i in range(5):
3     linha=[]
4     linha.append(input("nome: "))
5     linha.append(int(input("idade: ")))
6     GoT.append(linha)
7 print(GoT)
```

```
>>> %Run teste.py
```

nome: Sansa

idade: 20

nome: Bran

idade: 17

nome: Jon

idade: 24

nome: Daenerys

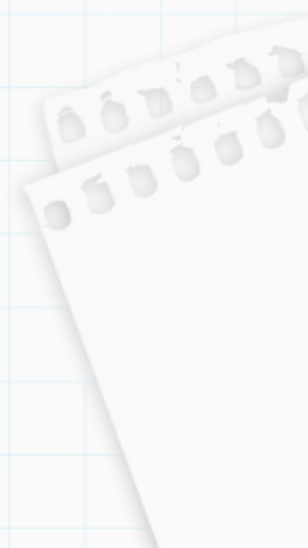
idade: 24

nome: Tyrion

idade: 39

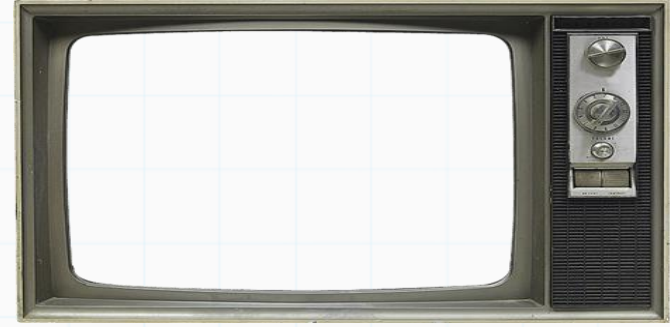
```
[['Sansa', 20], ['Bran', 17], ['Jon', 24], ['Daenerys', 24], ['Tyrion', 39]]
```

Sansa	20
Bran	17
Jon	24
Daenerys	24
Tyrion	39



Matrizes

Matrizes podem ter mais de 2 dimensões :



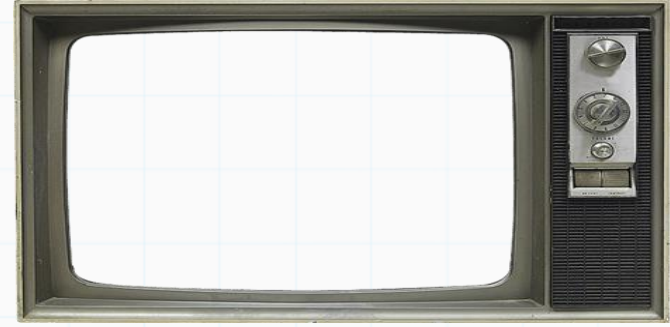
```
1 series_id = [ [['Sansa', 20], ['Bran', 17], ['Jon', 24], ['Daenerys', 24], ['Tyrion', 39]],  
2              [['Eleven', 14], ['Mike', 14], ['Dustin', 14], ['Lucas', 14], ['Will', 14]] ]
```

Sansa	20
Bran	17
Jon	24
Daenerys	24
Tyrion	39

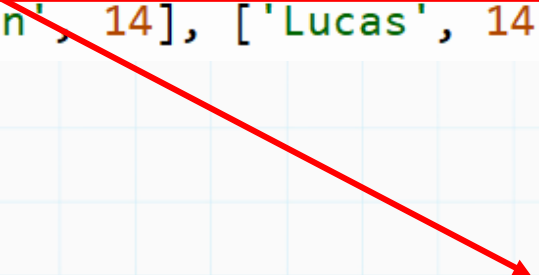
Eleven	14
Mike	14
Dustin	14
Lucas	14
Will	14

Matrizes

Matrizes podem ter mais de 2 dimensões :



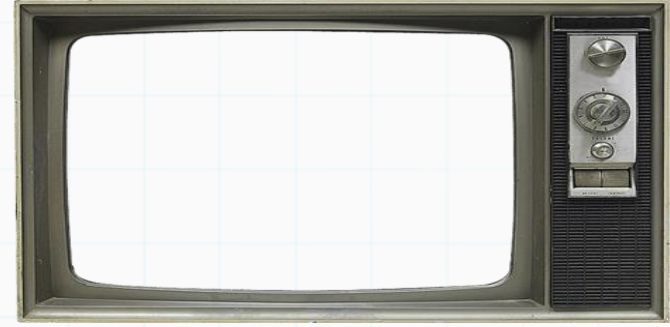
```
1 series_id = [ [['Sansa', 20], ['Bran', 17], ['Jon', 24], ['Daenerys', 24], ['Tyrion', 39]],  
2              [['Eleven', 14], ['Mike', 14], ['Dustin', 14], ['Lucas', 14], ['Will', 14]] ]
```



Sansa	20
Bran	17
Jon	24
Daenerys	24
Tyrion	39
Eleven	14
Mike	14
Dustin	14
Lucas	14
Will	14

Matrizes

Matrizes podem ter mais de 2 dimensões :



```
1 series_id = [ [['Sansa', 20], ['Bran', 17], ['Jon', 24], ['Daenerys', 24], ['Tyrion', 39]],  
2              [['Eleven', 14], ['Mike', 14], ['Dustin', 14], ['Lucas', 14], ['Will', 14]] ]
```

série, personagem, idade

```
4 print(series_id[0][1][1])
```

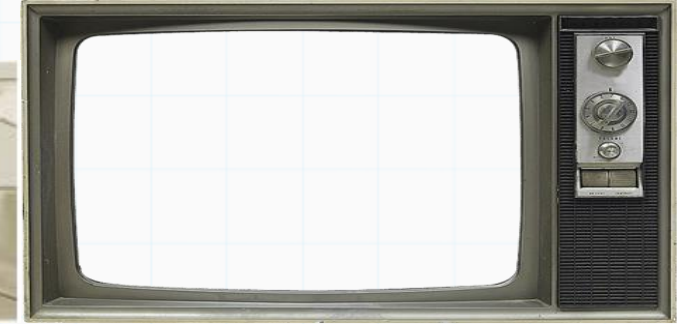
Shell x

```
>>> %Run teste.py
```

```
17
```

Sansa	20
Bran	17
Jon	24
Daenerys	24
Tyrion	39
Eleven	14
Mike	14
Dustin	14
Lucas	14
Will	14

Matrizes

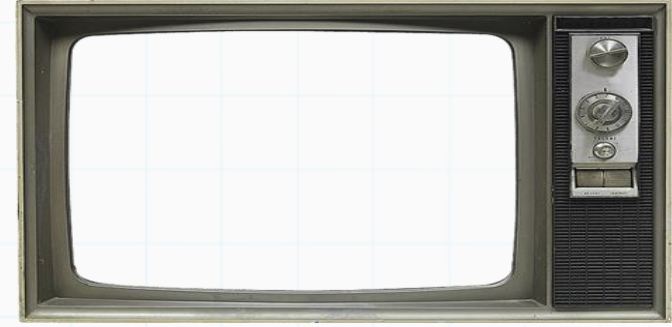


Fura Olho: O que será escrito ?

```
1 # Entrada 2, 3, 5
2 M = []
3 for i in range(3):
4     l=[0]*4
5     M.append(l)
6 for i in range(1,4):
7     M[1][i] = int(input())
8     if (M[1][i] >= 3):
9         ac=2
10        for j in range(1,M[1][i]+1,2):
11            ac=ac*j
12            M[2][i]=-ac
13    else:
14        ac=3
15        for j in range(2,M[1][i]+1,2):
16            ac=ac+j
17            M[2][i]=-ac
18    M[2][i] = M[2][i] * M[2][i]
19 for i in range(3):
20     print(M[i])
```

	0	1	2	3
0				
1				
2				

Até a próxima



Slides baseados no curso de Vanessa Braganholo