

ALGORITMIA E ESTRUTURAS DE DADOS

ESTRUTURAS DE DADOS NÃO PRIMITIVAS | LISTAS MULTIDIMENSIONAIS

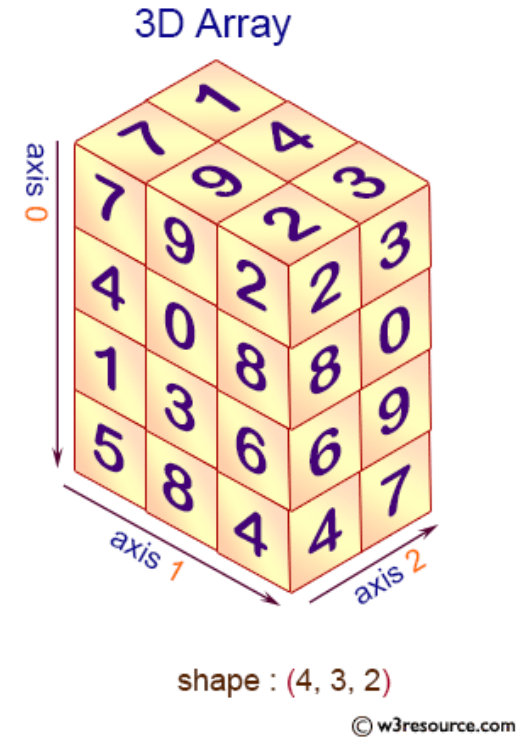
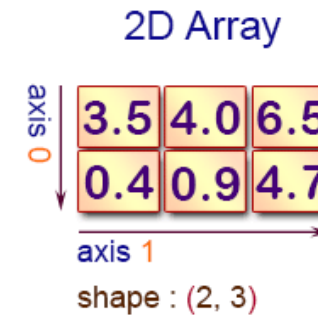
LICENCIATURA EM
TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB
#ESMAD #P.PORTO

1. Arrays | Listas Bidimensionais

- ☐ Conceito
- ☐ Índices
- ☐ Percorrer uma lista

2. Arrays | Listas Tridimensionais

- ☐ Conceito
- ☐ Índices
- ☐ Percorrer uma lista



❖ Listas | bidimensionais

- ❑ Uma lista geralmente contém dados numa ordem linear, ou numa única dimensão.
- ❑ No entanto, há muitos sistemas no “mundo real” que são multidimensionais. Para visualizar esses dados precisamos de uma estrutura de dados multidimensional
- ❑ Uma lista bidimensional poder servir para representar uma imagem digital, um jogo de tabuleiro, etc.
- ❑ Uma lista bidimensional nada mais é do que uma **lista de listas** (uma forma de ***Nested list***)
- ❑ Uma lista bidimensional pode também ser vista como uma tabela de 2 dimensões: linhas e colunas

```
myList = [0,1,2,3]
```

Lista unidimensional

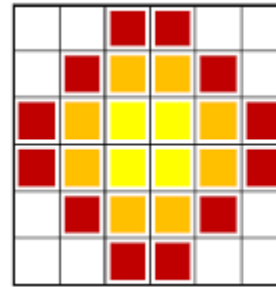
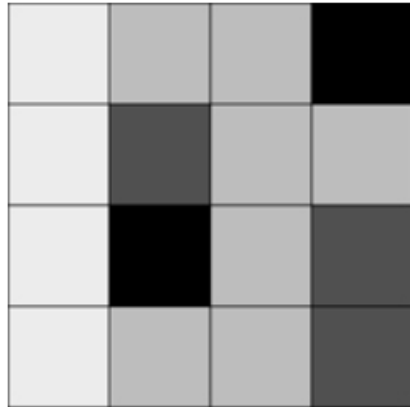
```
myList = [ [0, 1, 2, 3],  
           [3, 2, 1, 0],  
           [3, 5, 6, 1],  
           [3, 8, 3, 4] ]
```

→ Sub-listas de myList

Lista bidimensional

❖ Listas | bidimensionais

❑ Representação de uma lista bidimensional:



```
11 11 00 00 11 11
11 00 01 01 00 11
00 01 10 10 01 00
00 01 10 10 01 00
11 00 01 01 00 11
11 11 00 00 11 11
```

```
myList = [ [236, 189, 189, 0],
            [236, 80, 189, 189],
            [236, 0, 189, 80],
            [236, 189, 189, 80] ]
```



RGB de cada pixel, p.e.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \quad \text{Original matrix}$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}^T \Rightarrow \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

❖ Listas | bidimensionais

☐ Representação de uma lista bidimensional

Architecture of Multidimensional Collections in Java

Rows

	C0	C1	C2	C3	C4	C5	C6	C7	C8
R0	11	12	13	14	15	16	17	18	
R1	21	22	23	24	25	26			
R2	31	32	33	34	35	36	37		
R3	41	42	43	44	45				
R4	51	52	53	54	55	56	57	58	59
R5	61	62	63	64	65	66	67	68	

Columns

Each row can have different number of objects

Exemplo de representação de uma lista bidimensional

4 linhas

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

5 colunas

→ Lista constituída por 4 sub-listas
Cada uma delas com 5 elementos

❖ Listas | bidimensionais

- ❑ Iterar uma lista bidimensional como um objeto único

```
1  """
2  Listas bidimensionais
3  """
4  # Uma lista bidimensional é uma lista de listas
5  lista = [[1,2,3], [4,5,6], [7,8,9]]
6  print(lista)
7
```

```
C:\WINDOWS\System32\cmd.  ×  +  v
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
Press any key to continue . . . |
```

❖ Listas | bidimensionais

- ❑ Iterar uma lista bidimensional linha a linha

```
1  """
2  Listas bidimensionais
3  """
4  # Uma lista bidimensional é uma lista de listas
5  lista = [[1,2,3], [4,5,6], [7,8,9]]
6
7  # Iterar uma lista linha a linha (sub-lista a sub-lista)
8  for linha in lista:
9      print(linha)
```

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
Press any key to continue . . . |
```

❖ Listas | bidimensionais

- ❑ Iterar uma lista bidimensional através dos índices

```
Exemplo1.py > ...
1  # Listas bidimensionais
2
3
4  lista = [[1,2,3], [4,5,6], [7,8,9]]
5
6  # Iterar uma lista através dos índices
7  for i in range(len(lista)):          # len(lista) dá-nos o nº de linhas da lista
8      for j in range(len(lista[i])):    # len(lista[i]) dá-nos o nº de elementos da linha i da lista
9          print(lista[i][j], end=" ")
10         print()
11
12
13
14
15
```

C:\WINDOWS\py.exe

```
1 2 3
4 5 6
7 8 9
```


❖ Listas | bidimensionais

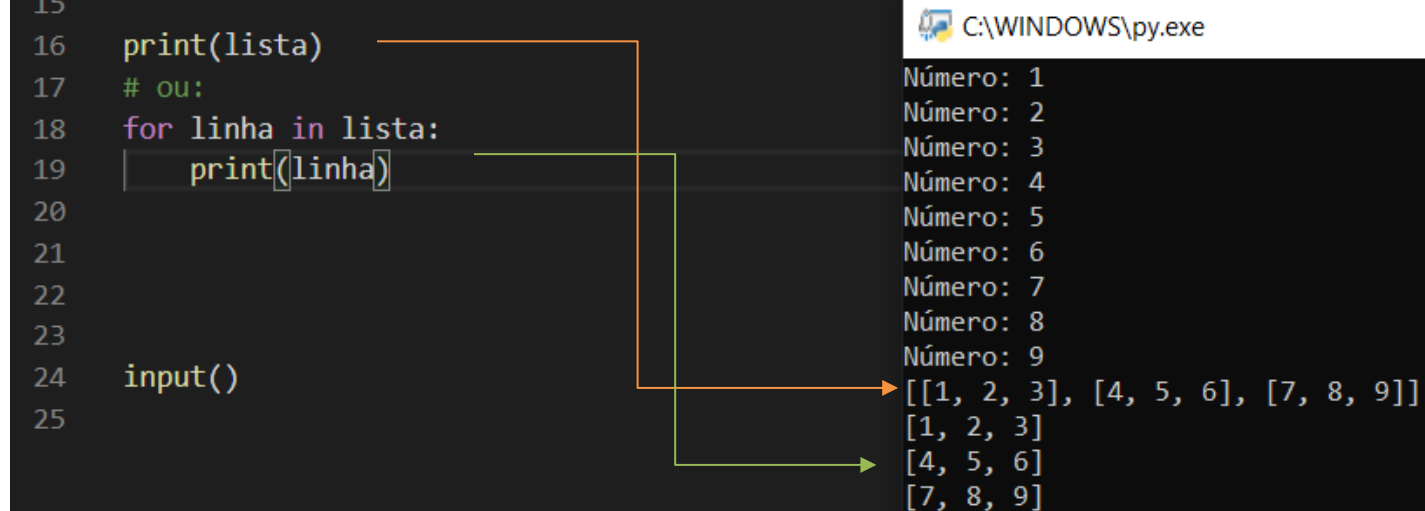
❑ Input de uma lista bidimensional

```
Exemplo1.py > ...
1  # Listas bidimensionais
2
3
4  lista = []
5
6  lin = 3      # nº de linhas
7  col = 3      # nº de colunas ou nº de elementos em cada linha
8
9              # Iterar as diversas linhas da lista
10 for i in range(lin):
11     lista.append([])          # acrescenta uma lista vazia para cada linha
12     for j in range(col) :
13         numero = int(input("Número: "))
14         lista[i].append(numero) # acrescenta à lista o numero lido
15
16 print(lista)
17 # ou:
18 for linha in lista:
19     print(linha)
20
21
22
23
24 input()
25
```

C:\WINDOWS\py.exe

Número: 1
Número: 2
Número: 3
Número: 4
Número: 5
Número: 6
Número: 7
Número: 8
Número: 9

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



❖ Listas | bidimensionais

- ❑ Função que cria uma lista com determinada dimensão

```
Exemplo1.py > cria_lista
1  import random
2
3  # Listas bidimensionais
4
5  def cria_lista(nlin, ncol):
6  # função que cria uma lista de valores aleatórios, com nlin linhas, e ncol elementos por cada linha
7      lista = []
8      for i in range(nlin):
9          lista.append([])          # acrescenta uma lista vazia para cada linha
10         for j in range(ncol) :
11             numero = random.randint(0,50)
12             lista[i].append(numero) # acrescenta à lista o numero aleatório
13     return lista
14
15
16
17 lista = cria_lista(3,3)
18 for linha in lista:
19     print(linha)
20
21
22
```

```
C:\WINDOWS\py.exe
[16, 6, 33]
[16, 21, 25]
[13, 16, 31]
```

❖ Listas | bidimensionais

❑ Funções *built-in Python* funcionam em listas de uma dimensão

```
Exemplo1.py > ...
12     lista[i].append(numero) # acrescenta à lista o numero aleatório
13     return lista
14
15
16     lista = cria_lista(3,3)
17
18     # Funções para minipular listas funcionam em listas unidimensionais!
19     lista[0].sort()      # ordena a 1ª linha ordem ascendente
20
21     lista[1].sort()
22     lista[1].reverse()  # ordena a 2ª linha ordem descendente
23
24     for linha in lista:
25         print(linha)
26
27     numero = 2
28     pos = lista[0].index(numero)
29     print("{0} aparece na posição {1} ".format(numero, pos))
30
31     print("{0} aparece {1} vezes ".format(numero, lista[0].count(numero)))
32
33
34
```

C:\WINDOWS\py.exe

Numero: 1
Numero: 2
Numero: 2
Numero: 3
Numero: 4
Numero: 4
Numero: 5
Numero: 6
Numero: 6

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

2 aparece na posição 1
2 aparece 2 vezes

❖ Listas | bidimensionais

❑ Funções *built-in Python* funcionam em listas de uma dimensão

```
23 lista = [[1,2,3], [2,3,4], [3,4,5]]
24 lista.index(5)
```

Exception has occurred: ValueError ×
5 is not in list
File "C:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 06\Exemplo_listas.py", line 24, in <module>
lista.index(5)

25

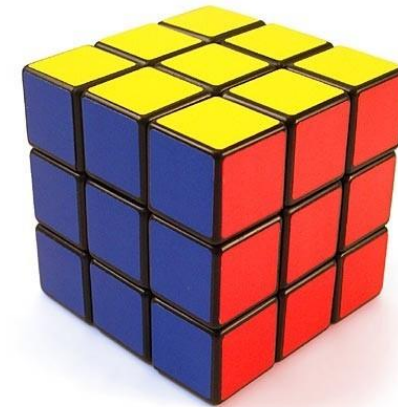
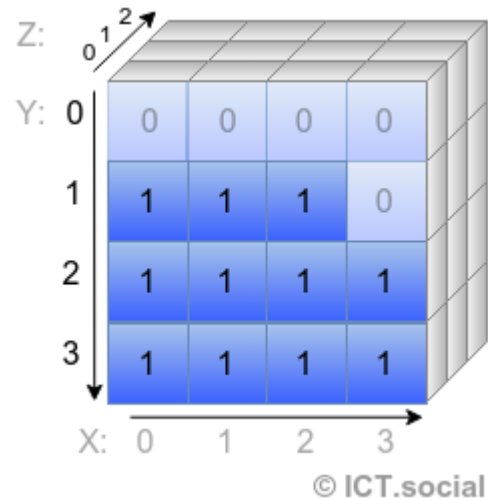
```
lista = [[1,2,3], [2,3,4], [3,4,5]]
pos= lista[2].index(5)
print(pos)
```

c:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 06 - VS Code Console

```
2
Press any key to continue . . .
```

❖ Listas | tridimensionais

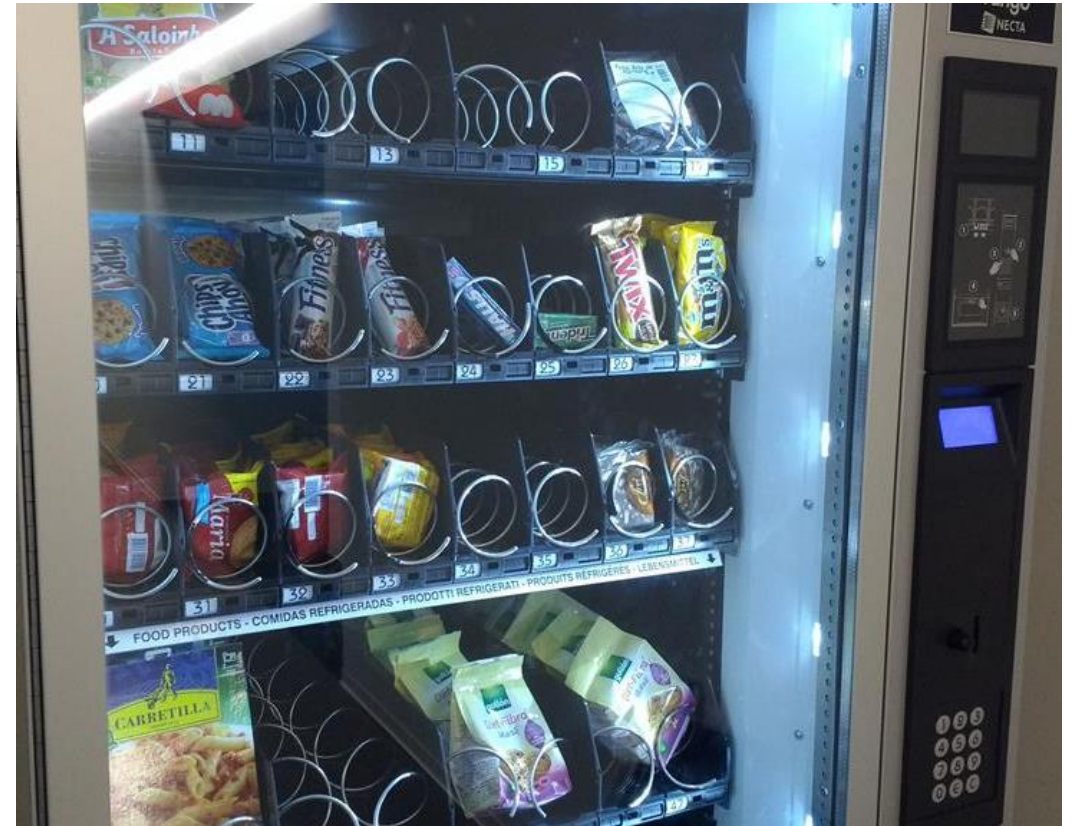
- ☐ Por vezes é necessário recorrermos a uma lista com mais dimensões
- ☐ Todos nós podemos pelo menos imaginar uma lista 3D
- ☐ Uma lista tridimensional pode ser vista como uma lista contendo linhas, colunas e profundidade



❖ Listas | tridimensionais

❑ Exemplos de aplicabilidade:

- ❑ Parque de estacionamento com vários pisos, filas e lugares
- ❑ Representação de produtos numa máquina de *vending*
- ❑ Menus de apps com 3 (ou mais) níveis de profundidade
- ❑ Etc.



❖ Listas | tridimensionais

```
Exemplo1.py > ...
4
5 def cria_lista(nlin, ncol, nprof):
6     # função que cria uma lista de valores aleatórios, com nlin linhas, e ncol elementos por cada linha
7     lista = []
8     for i in range(nlin):
9         lista.append([])          # acrescenta uma lista vazia para cada linha
10        for j in range(ncol) :
11            lista[i].append([])    # em cada linha, acrescenta uma coluna à lista
12            for k in range(nprof):
13                numero = input("Numero: ")
14                lista[i][j].append(numero) # Para cada linha / coluna, acrescenta dados à lista (profundidade)
15        return lista
16
17
18 lista = cria_lista(3,3, 3)
19
20 for linha in lista:
21     print("linha :", linha)
22
23
24
25
26 input()
27
```

C:\WINDOWS\py.exe

Numero: 12
Numero: 13
Numero: 14
Numero: 15
Numero: 16
Numero: 17
Numero: 18
Numero: 19
Numero: 20
Numero: 21
Numero: 22
Numero: 23
Numero: 24
Numero: 25
Numero: 26
Numero: 27
linha : [['01', '02', '03'], ['04', '05', '06'], ['07', '08', '09']]
linha : [['10', '11', '12'], ['13', '14', '15'], ['16', '17', '18']]
linha : [['19', '20', '21'], ['22', '23', '24'], ['25', '26', '27']]