Lab2: Lab1 nâng cao

1. Numpy

In [2]:
```
import numpy as np

x = np.array([1, 2, 3, 4, 5, 6, 7, 8])

# Lấy số "3"
print(x[2])

# lấy giá trị 123
print(x[:3])

# lấy từ 3 đến 8
print(x[2:])

# lấy từ 3 đến 7
print(x[2:-1])

# lấy ra số 7 và 8
print(x[-2:])

# lấy phần tử 2 và 4
print(x[2], x[4])
print(x[[2, 4]])
```

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

In [30]:
```
# tạo mảng 2 chiều

data = [[1, 2], [3,4], [5,6]]
mang = np.array(data)
print(mang)

print(mang[0,1])

print(mang[-1,-1])

print(mang[[1,2]])

print(mang[1:, :])

print(mang[1], mang[2])

# cách khác

print(mang[1:])
```

```
print(mang[1:3])

# cách khác dùng -
print(mang[-2:])

# lấy 1,3,5
print(mang[:, 0])

# lấy 1,5
print(mang[[0,2], 0])

# lấy 1 và 5 theo fomat start:stop:step
print(mang[::2, 0])

# cách khác
print(mang[[0,2], 0])
```

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

In [31]:
```
X = [
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10,11,12],
    [13,14,15,16]
]

x = np.array(X)
print(x)
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15 16]]
```

In [47]:
```
# lấy 2,6,10,14
print(x[:, 1])

# lấy 7,11
```

```
print(x[1:3, 2])

# lấy 4,7,10
print(x[0, 3], x[1, 2], x[2, 1])

# lấy 16 15 14 13
print(x[-1, -1], x[-1, -2], x[-1, -3], x[-1, -4])
print(x[3,::-1])

print(x[-1, -1::-1])
```

```
[ 2  6 10 14]
[ 7 11]
4 7 10
16 15 14 13
[16 15 14 13]
[16 15 14 13]
```

In [ ]:

```
# Bài tập:
# Cho ma trận sau:
# [[ 99  99  99 ]
#  [ 99  99  99 ]
#  [ 99  99  99 ]]

#  Giả sử: 0 = O và 1 là X
#  Nhận đầu vào từ phía X và O luân phiên
#  cho các bạn 1 cặp chỉ số, nếu phía nhập (0,0) thì ma trận trở thành

# [[ X  99  99 ]
#  [ 99  99  99 ]
#  [ 99  99  99 ]]

#  Nếu phía O nhập (0,0) (trùng với X) thì yêu cầu nhập lại, nếu không thì điền vào
 import numpy as np

def is_valid_move(mat, row, col):
    return mat[row, col] == 99

def make_move(mat, row, col, player):
    mat[row, col] = 'X' if player == 1 else 'O'

def print_matrix(mat):
    for row in mat:
        print(' '.join(str(cell) for cell in row))
    print()  # Thêm dòng trống để dễ nhìn

def check_winner(mat, player_symbol):
    # Kiểm tra hàng và cột
    for i in range(3):
        if all(mat[i, j] == player_symbol for j in range(3)) or all(mat[j, i] == pl
            return True
    # Kiểm tra 2 đường chéo
    if all(mat[i, i] == player_symbol for i in range(3)) or all(mat[i, 2 - i] == pl
        return True
    return False
```

```
def play_game():
    matrix = np.full((3, 3), 99, dtype=object)  # Sửa dtype thành object
    current_player = 1
    moves = 0
    while moves < 9:
        print_matrix(matrix)
        try:
            move = input(f"Player {'X' if current_player == 1 else 'O'}, enter your
            row, col = map(int, move.replace('(', '').replace(')', '').split(','))
            if row not in range(3) or col not in range(3):
                print("Invalid input, row and column must be between 0 and 2.")
                continue
            if is_valid_move(matrix, row, col):
                make_move(matrix, row, col, current_player)
                player_symbol = 'X' if current_player == 1 else 'O'
                if check_winner(matrix, player_symbol):
                    print_matrix(matrix)
                    print(f"Player {player_symbol} wins!")
                    return
                current_player = 1 - current_player
                moves += 1
            else:
                print("Cell already taken, try again.")
        except (ValueError, IndexError):
            print("Invalid input, please enter the move in the format (row,col).")
    print_matrix(matrix)
    print("It's a draw!")

play_game()
```

In [75]:

```
# BT:
# Cho ma trận
# [
#     [1, 2, 3],
#     [4, 5, 6],
#     [7, 8, 9]
# ]

x = np.array([[1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]])
# 1. Lấy ra số 4 5 6
print(x[1])

# 2. Lấy ra số 2 5
print(x[:2, 1])
# 3. Lấy ra số 3 4
print(x[0, 2], x[1, 0])
# cach khac
print(x[0, -1], x[1, 0])
# cach khac
print(x[[0,1], [2, 0]])
# 4. Lấy ra số 9 6 3
print(x[2, 2], x[1, 2], x[0, 2])
```

```
print(x[::-1, 2])
```

```
[4 5 6]
[2 5]
3 4
3 4
[3 4]
9 6 3
[9 6 3]
```

In [88]:
```
x = np.array([1, 2, 3,4,5,6,7,8,9,10])

mang_chan = [ _ for _ in x if _ % 2 == 0]
print(mang_chan)

# tạo 1 mảng toàn số 1
mang_1 = np.ones((3,3))

print(mang_1)

m3 = np.arange(3)
print(mang_1 + m3)
```

```
[2, 4, 6, 8, 10]
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
[[1. 2. 3.]
 [1. 2. 3.]
 [1. 2. 3.]]
```

In [103…
```
x = np.arange(3)
print(len(x))
print(x.shape)
print(x.size)

x = x.reshape((3,1))
print(x)

y = np.arange(3)

print(x+y)
```

```
3
(3,)
3
[[0 1 2]
 [1 2 3]
 [2 3 4]]
```

In [3]:
```
# Bài tập về nhà
# Tạo mảng np có khích thước 150x5 hãy tưởng tượng mảng này chứa 150 mẫu về chiều c
# Chia mảng 4 cột đầu tiên thành 1 biến có tên là x và cột cuối cùng bằng y.
# Chia x thành  X_train và X_test chứa lần lượt là 70% và 30% dữ liệu, trong đó Y_t

# Tạo mảng np có kích thước 150x5
```

```python
data = np.random.uniform(0, 10,(150, 5))

# Chia mảng 4 cột đầu tiên thành 1 biến có tên là x và cột cuối cùng bằng y
x = data[:, :-1]
y = data[:, -1]

dataset_size = x.shape[0]
train_size = int(dataset_size * 0.7)


X_train = x[:train_size]
x_test = x[train_size:]

print(X_train.shape, x_test.shape)

# Tạo 10 tập dữ kiệu k chồng chéo của X_train .

split = X_train.shape[0]//10

print(split)

for counter in range(0, X_train.shape[0], split):
    print(X_train[counter:counter+split].shape)

# BTVN: Input, output, explain bài trên
```

```
(105, 4) (45, 4)
10
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(10, 4)
(5, 4)
```