AI VIETNAM
All-in-One Course

Python Review

1D, 2D, 3D List, Numpy

Year 2023

Outline

- 1D, 2D, 3D LIST
- 1D, 2D, 3D Numpy

2D LIST – Basic Exercise

Bài 1: Viết một chương trình Python để tính tổng các phần tử trên đường chéo chính của một ma trận vuông (mảng hai chiều) đã cho. Đường chéo chính là dãy các phần tử từ góc trên bên trái đến góc dưới bên phải của ma trận.
Ví dụ, với ma trận sau đây:

- **Input**:

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

- **Output**

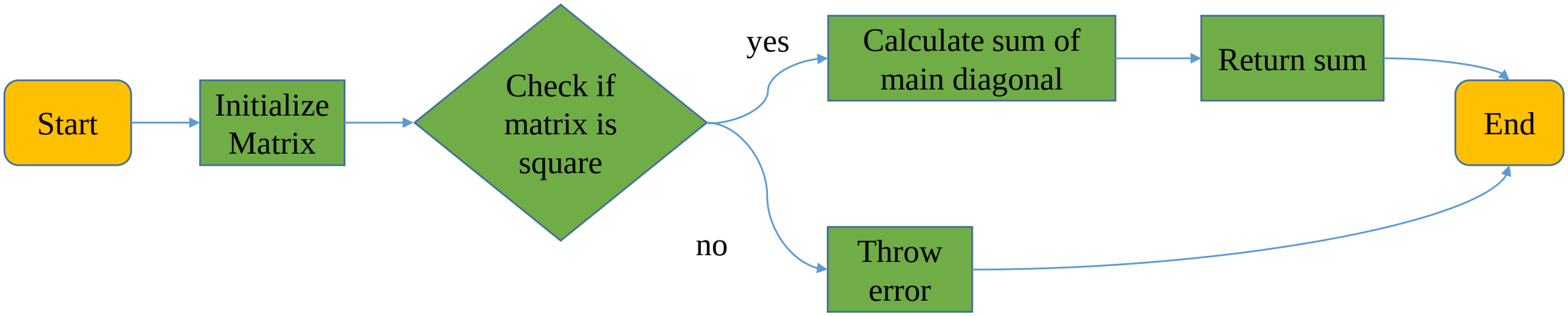
Tổng của các phần tử trên đường chéo chính là 1 + 5 + 9 = 15.

Gợi ý:

Sử dụng vòng lặp để duyệt qua các hàng và cột tương ứng trên đường chéo chính.
Sử dụng chỉ số hàng và chỉ số cột giống nhau để truy cập các phần tử trên đường chéo chính.

2D LIST – Basic Exercise

Ý tưởng:



2D LIST – Basic Exercise

Bài 2: Viết một chương trình Python để sắp xếp một ma trận (mảng hai chiều) đã cho theo tổng của các phần tử trong mỗi hàng. Tổng của một hàng được tính bằng cách cộng tất cả các phần tử trong hàng đó. Sau khi sắp xếp, các hàng có tổng nhỏ nhất sẽ xuất hiện đầu tiên trong ma trận.
Ví dụ:

- **Input**:

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

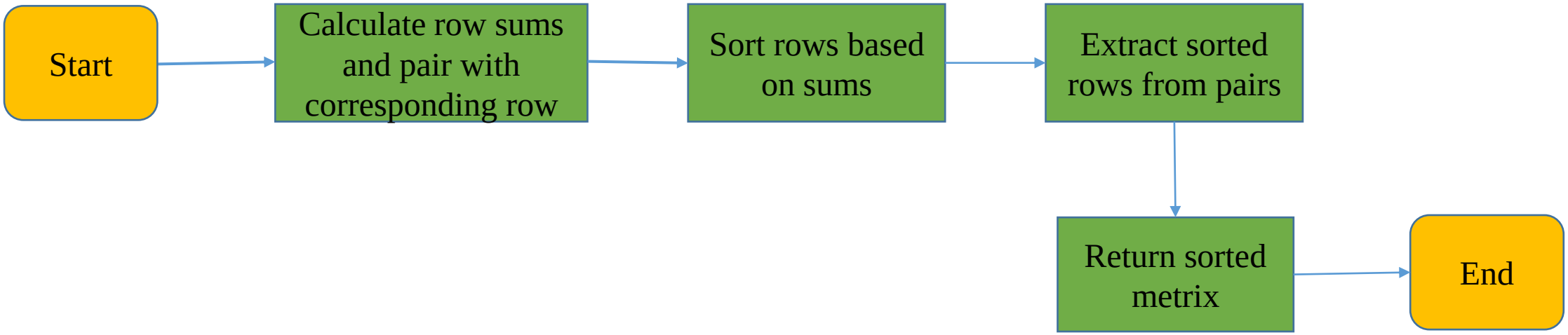
- **Output**:

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

Gợi ý:
Tạo một danh sách các cặp (tổng hàng, hàng) để lưu trữ tổng của mỗi hàng và hàng tương ứng.
Sử dụng hàm sorted() trong Python để sắp xếp danh sách theo tổng hàng.
Lặp qua các cặp đã sắp xếp để tạo lại ma trận ban đầu theo thứ tự đã sắp xếp.

2D LIST – Basic Exercise

Ý tưởng:



2D LIST – Basic Exercise

Bài 3: Một công ty có một danh sách hàng tồn kho của các sản phẩm khác nhau. Các thông tin hàng tồn kho bao gồm tên sản phẩm, số lượng trong kho, và giá bán của mỗi sản phẩm. Công ty muốn tính toán tổng giá trị của hàng tồn kho của mình.

Danh sách hàng tồn kho của công ty được biểu diễn bằng một ma trận có kích thước nx3, trong đó n là số lượng sản phẩm khác nhau. Các cột trong ma trận biểu diễn tên sản phẩm, số lượng trong kho và giá bán của mỗi sản phẩm.

Ví dụ, ma trận hàng tồn kho có thể nhìn như sau:

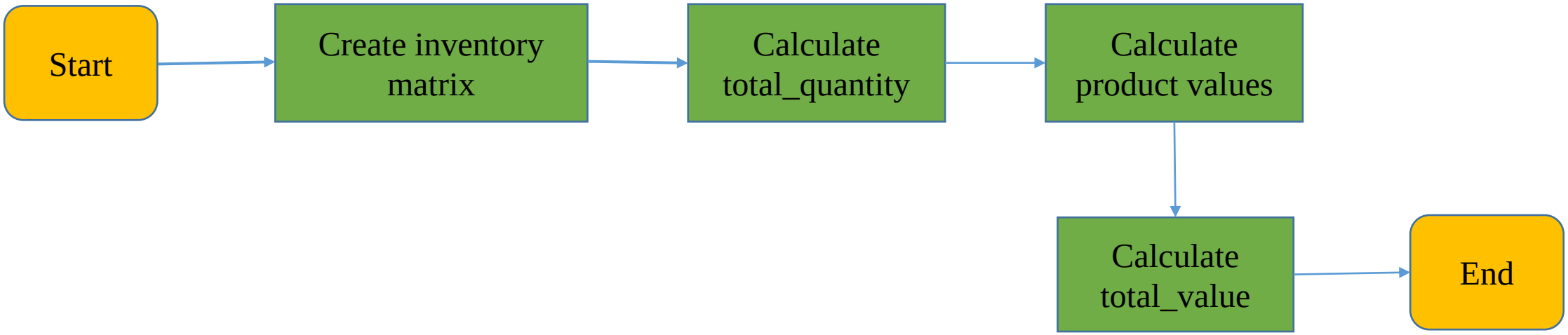
Hàng tồn kho:

```
|    Tên    | Số lượng | Giá bán |
| Sản phẩm 1 |    10    |   5.0   |
| Sản phẩm 2 |    5     |   2.5   |
| Sản phẩm 3 |    3     |   8.0   |
```

- Tính tổng số lượng hàng tồn kho của tất cả các sản phẩm?
- Tính giá trị hàng tồn kho của mỗi sản phẩm bằng cách nhân số lượng với giá bán của sản phẩm tương ứng?
- Tính tổng giá trị của toàn bộ hàng tồn kho?

2D LIST – Basic Exercise

Ý tưởng:



3D LIST – Basic Exercise

Bài 1: Viết hàm find_max để tìm giá trị lớn nhất trong một list 3D. List 3D có thể được biểu diễn như sau:

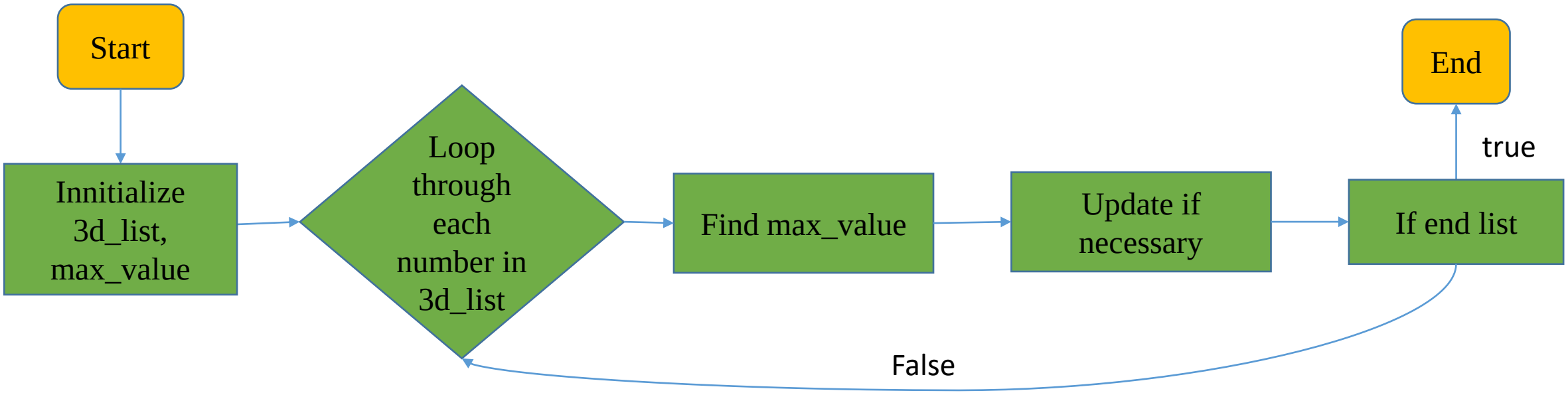
- **Input**

```
list_3d = [
    [
        [1, 2, 3],
        [4, 5, 6]
    ],
    [
        [7, 8, 9],
        [10, 11, 12]
    ]
]
```

- **Output**: max_value = 12

3D LIST – Basic Exercise

Ý tưởng:



3D LIST – Basic Exercise

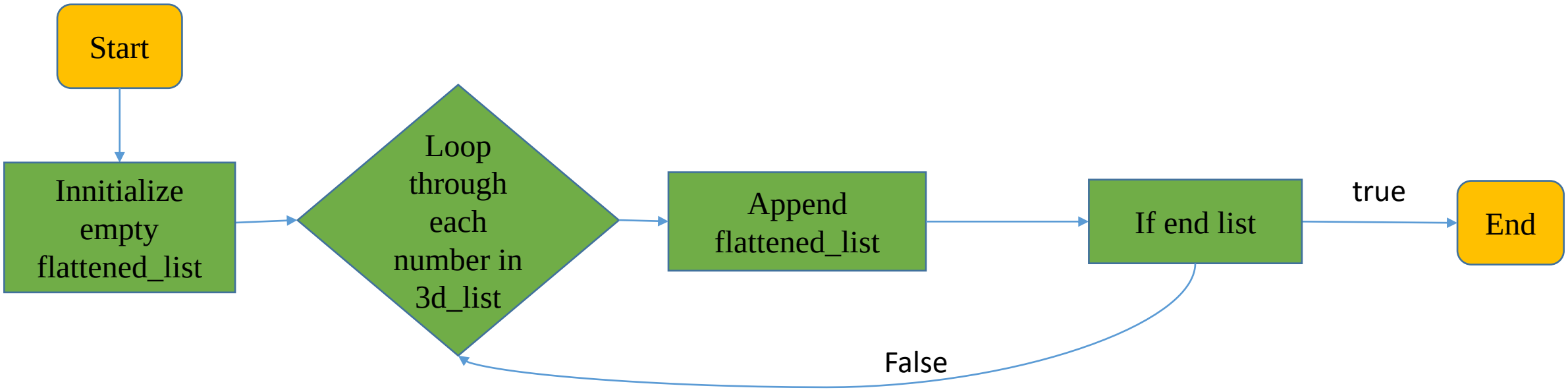
Bài 2: Viết hàm flatten_3d_list để làm phẳng một list 3D thành một list 1D (danh sách một chiều). Sử dụng cùng list 3D trong bài tập 1, hàm flatten_3d_list nên trả về:

Output: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

Lưu ý: Hãy cố gắng sử dụng nested for loops (vòng lặp for lồng nhau) hoặc list comprehension để giải quyết các bài tập này.

3D LIST – Basic Exercise

Ý tưởng:



3D LIST – Basic Exercise

Bài 3: Xoay list 3D theo chiều kim đồng hồ
Hãy viết một hàm rotate_3d_clockwise nhận vào một list 3D và trả về list 3D đã được xoay 90 độ theo chiều kim đồng hồ. Giả sử rằng kích thước của các list con luôn đồng nhất.

Ví dụ:

- **input:**

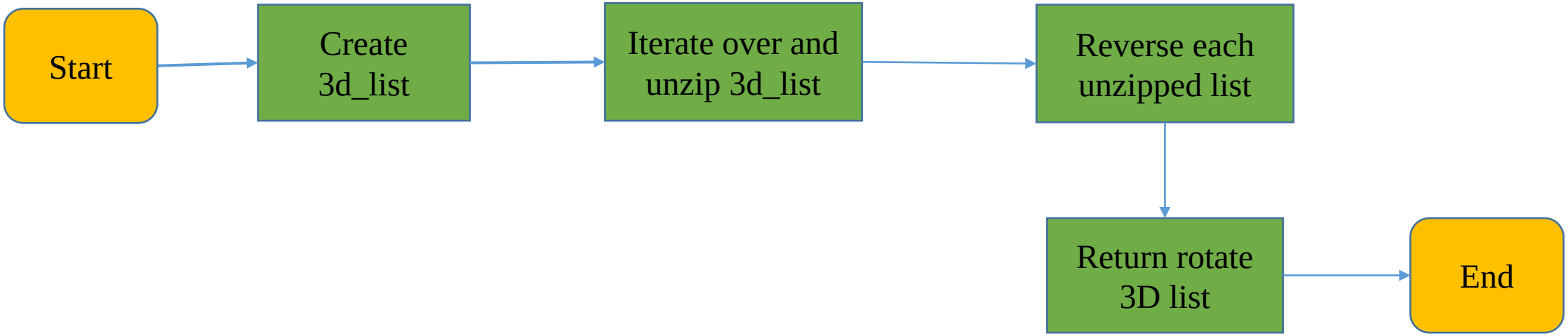
```
list_3d = [
    [[1, 2, 3], [4, 5, 6]],
    [[7, 8, 9], [10, 11, 12]]
]
```

- **output:**

```
[
    [[[7, 8, 9], [1, 2, 3]],
     [[10, 11, 12], [4, 5, 6]]]
]
```

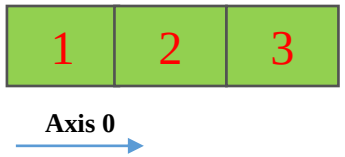
3D LIST – Basic Exercise

Ý tưởng:

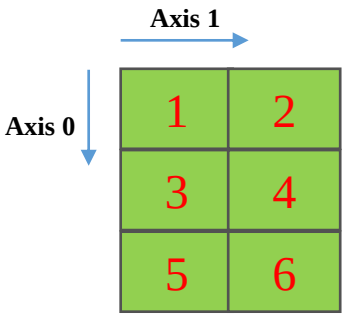


Outline

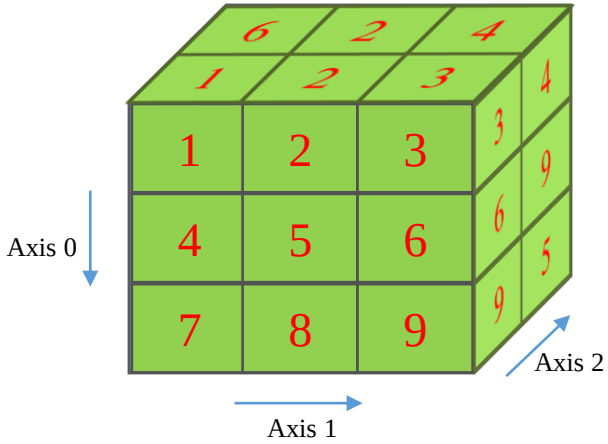
- 1D, 2D, 3D LIST
- 1D, 2D, 3D Numpy



1D array



2D array



3D array

1D array – Basic

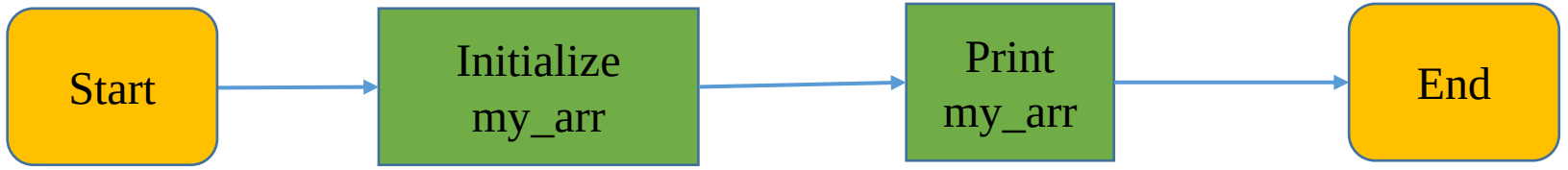
Bài 1: Bạn hãy tạo 1D array gồm 5 phần tử [1, 2, 3, 4, 5]. Sau đó in ra màn hình 1D array vừa tạo ?

Cấu trúc cần nhớ!

```
arr_np = np.array(python_list)
```

❑ Cách 1:

Ý tưởng:



1D array – Basic

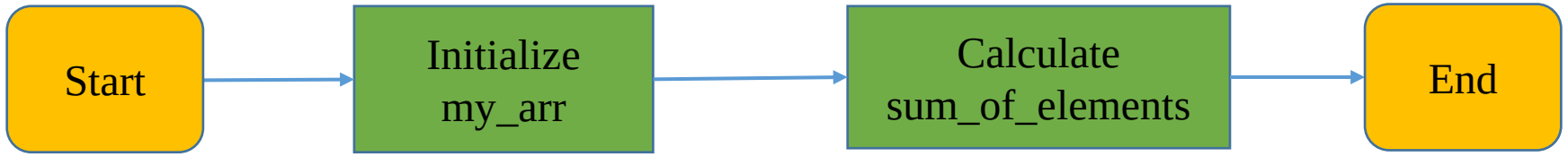
Bài 2: Bạn hãy xây dựng chương trình thực hiện tính tổng các phần tử trong 1D array?

- Input: my_arr = [1, 2, 3, 4, 5]
- Output: 15

Cấu trúc cần nhớ!

```
np.sum(arr)
```

Ý tưởng:



1D array – Basic

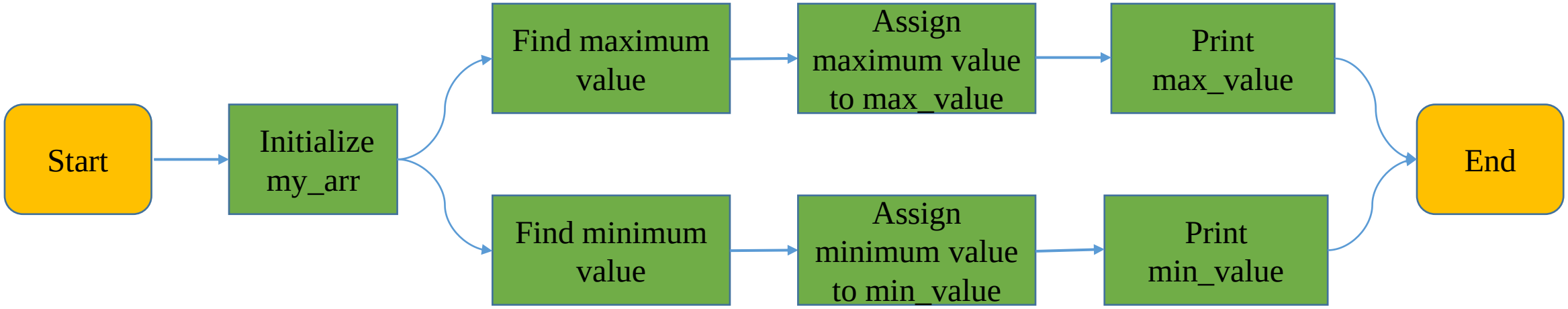
Bài 3: Bạn hãy xây dựng chương trình tìm giá trị lớn nhất và nhỏ nhất trong 1d array?

- Input: my_arr = [1, 2, 3, 4, 5]
- Output: max = 5, min = 1

Cấu trúc cần nhớ!

np.min(arr) np.max(arr)

Ý tưởng:



1D array – Basic

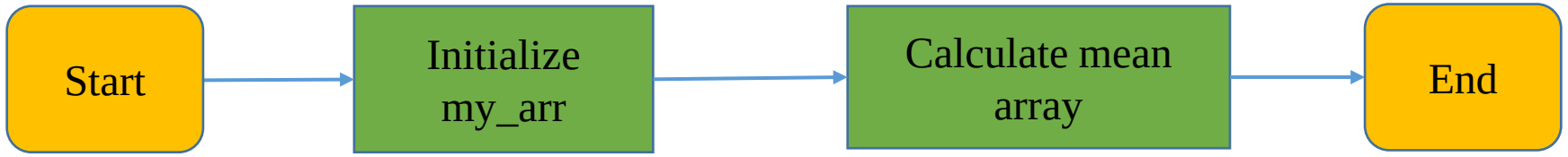
Bài 4: Bạn hãy xây dựng chương trình tính trung bình cộng của 1D array?

- Input: my_array = [1, 2, 3, 4, 5]
- Output: 3.0

Cấu trúc cần nhớ!

```
np.mean(arr)
```

Ý tưởng:



1D array – Basic

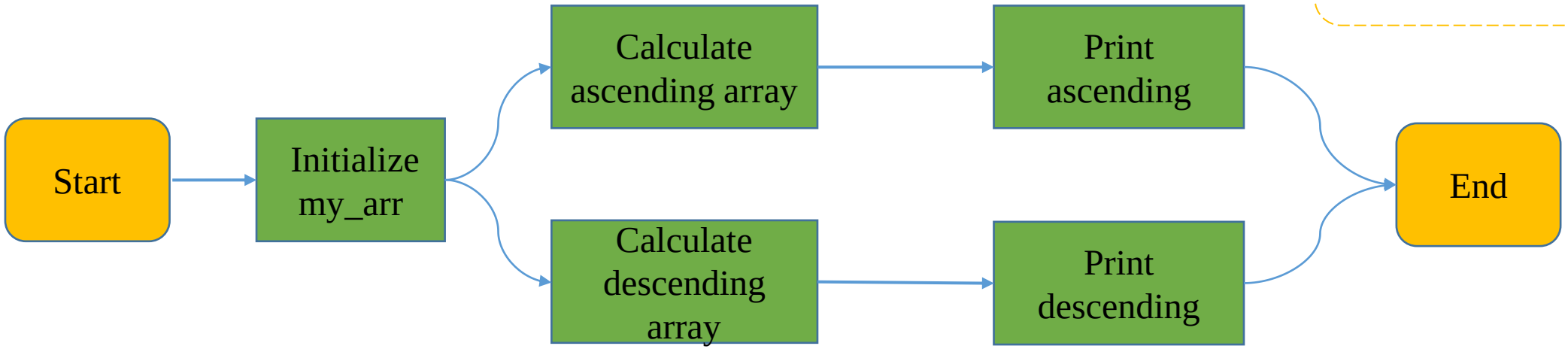
Bài 5: Bạn hãy xây dựng chương trình sắp xếp 1D array sau theo chiều tăng dần hoặc giảm dần?

- Input: my_array = [5, 3, 1, 4, 2]
- Output: ascending = [1, 2, 3, 4, 5], descending: [5, 4, 3, 2, 1]

Ý tưởng:

Cấu trúc cần nhớ!

```
np.sort(array)
```



1D array – Basic

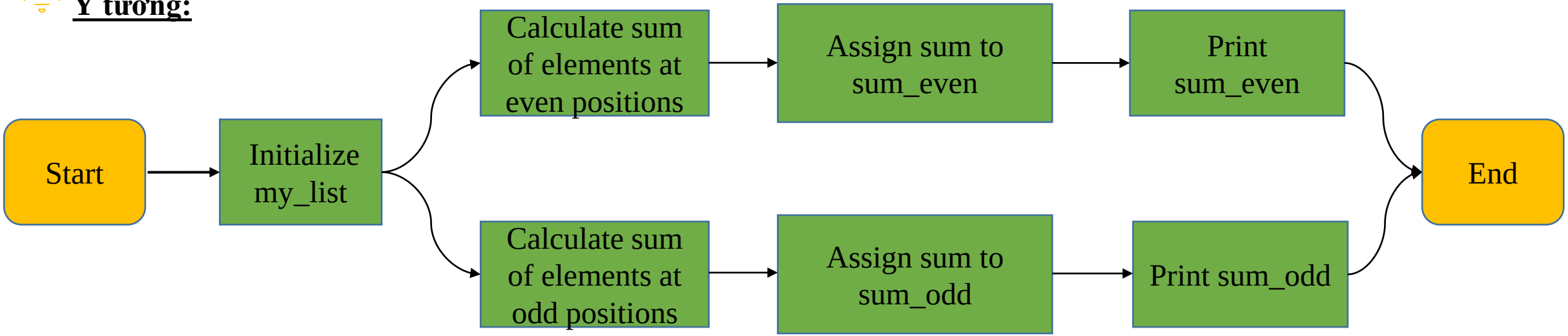
Bài 6: Bạn hãy xây dựng chương trình tổng các phần tử trên các vị trí chẵn hoặc lẻ?

- Input: my_array = [1, 2, 3, 4, 5]
- Output: even = 6, odd =9

Cấu trúc cần nhớ!

```
array[::2],
array[1::2]
```

Ý tưởng:



1D array - Advanced

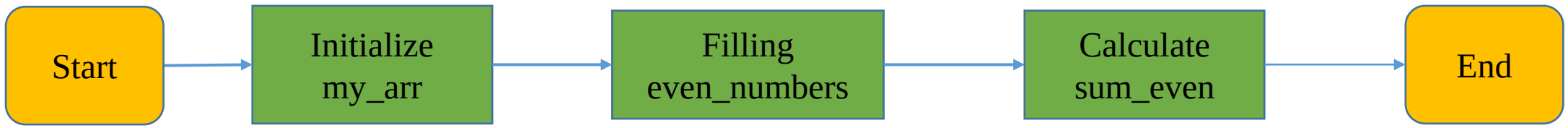
Bài 1: Bạn hãy xây dựng chương trình tìm các số chẵn trong array và tính tổng của chúng ?

- Input: my_array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
- Output: even_numbers = [2, 4, 6, 8, 10], sum_even = 30

Ý tưởng:

Cấu trúc cần nhớ!

array[condition]



1D array - Advanced

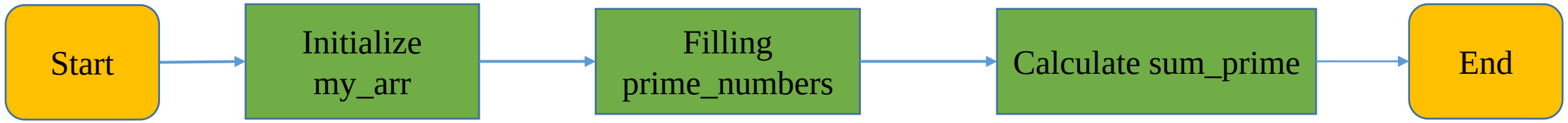
Bài 2: Bạn hãy xây dựng chương trình tính tổng của các số nguyên tố trong array?

- Input: my_array = [2, 3, 4, 5, 6, 7, 8, 9, 10]
- Output: prime_numbers = [2, 3, 5, 7], sum_prime = 17

Ý tưởng:

Cấu trúc cần nhớ!

```
array[np.vectorize(your_function)(array)]
```



1D array - Advanced

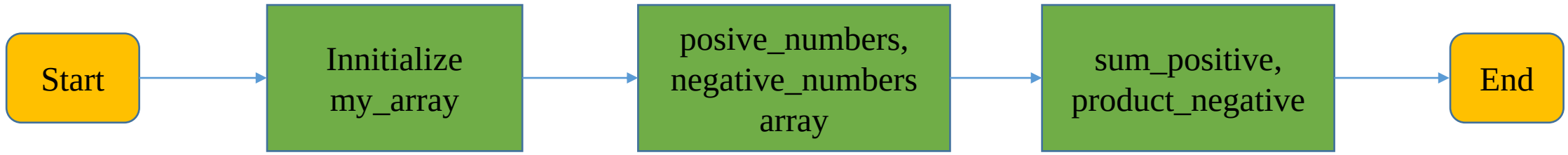
Bài 3: Bạn hãy xây dựng chương trình tính tổng các số dương và tích các số âm trong array?

- Input: my_array = [-1, 2, -3, 4, -5, 6]
- Output: sum_positive = 12, product_negative = -15

Ý tưởng:

Cấu trúc cần nhớ!

```
np.product(array)
```



1D array - Advanced

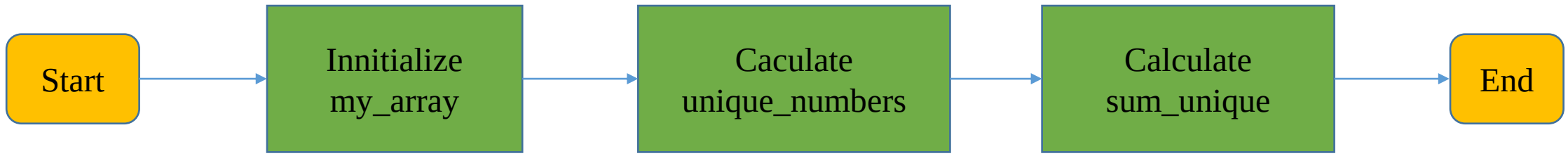
Bài 4: Bạn hãy xây dựng chương trình tính tổng các số trong array sau khi loại bỏ các số trùng lặp?

- Input: my_array = [1, 2, 3, 2, 4, 5, 1, 6, 7, 8, 5]
- Output: unique_numbers = [1, 2, 3, 4, 5, 6, 7, 8], sum_unique =36

Cấu trúc cần nhớ!

np.unique(array)

Ý tưởng:



Outline

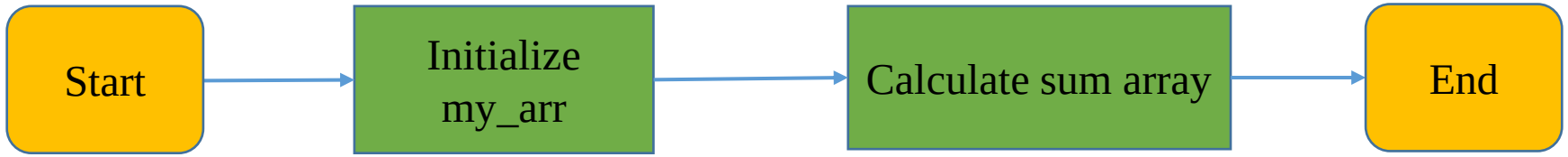
- 1D LIST
- 2D LIST

2D array - Basic

Bài 1: Bạn hãy xây dựng chương trình tính tổng các phần tử trong array 2D?

- Input: my_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
- Output: total = 45

Ý tưởng:

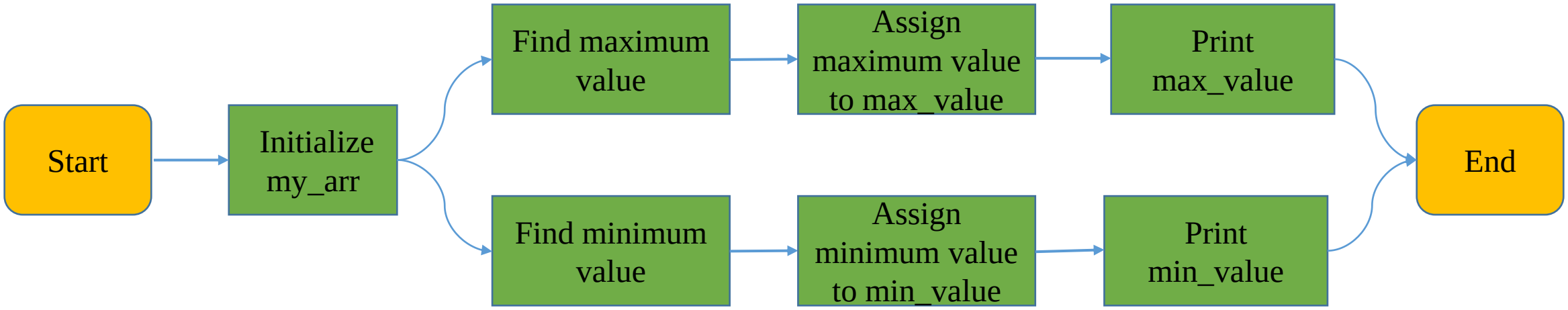


2D array - Basic

Bài 2: Bạn hãy xây dựng chương trình tìm giá trị lớn nhất và nhỏ nhất trong 2D array?

- Input: my_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
- Output: max_value = 9, min_value = 1

Ý tưởng:

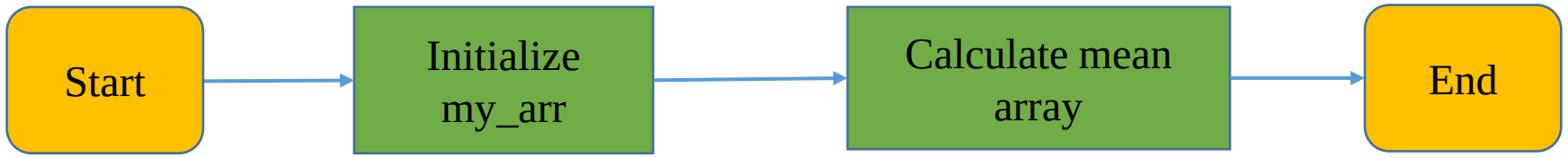


2D array - Basic

Bài 3: Bạn hãy xây dựng chương trình tính trung bình cộng của 2D array?

- Input: my_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
- Output: average = 5.0

Ý tưởng:



2D array - Basic

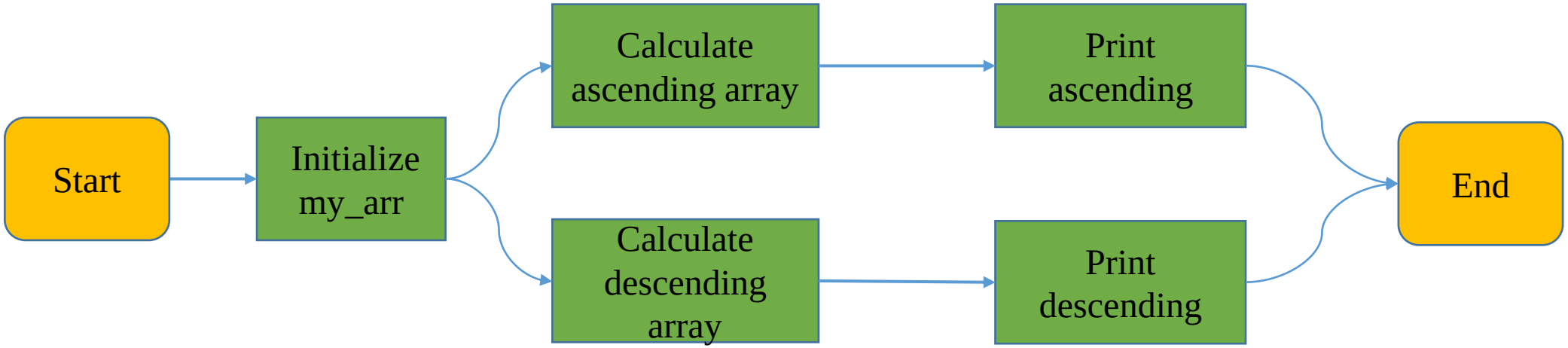
Bài 4: Bạn hãy xây dựng chương trình sắp xếp 2D array theo thứ tự tăng dần hoặc giảm dần?

- Input: my_array = [[5, 3, 1], [4, 2, 6], [9, 7, 8]]
- Output: sorted_ array = [[1, 3, 5], [2, 4, 6], [7, 8, 9]], reverse_sorted_ array = [[5, 3, 1], [6, 4, 2], [9, 8, 7]]

Cấu trúc cần nhớ!

np.sort(array, axis)

Ý tưởng:

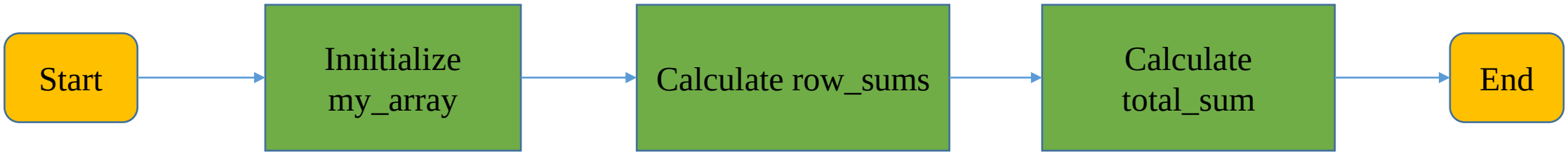


2D array - Advanced

Bài 1: Bạn hãy xây dựng chương trình tính tổng các phần tử trong mỗi hàng và tính tổng của các tổng hàng?

- Input: my_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
- Output: row_sums= [6, 15, 24], total_sum = 45

Ý tưởng:

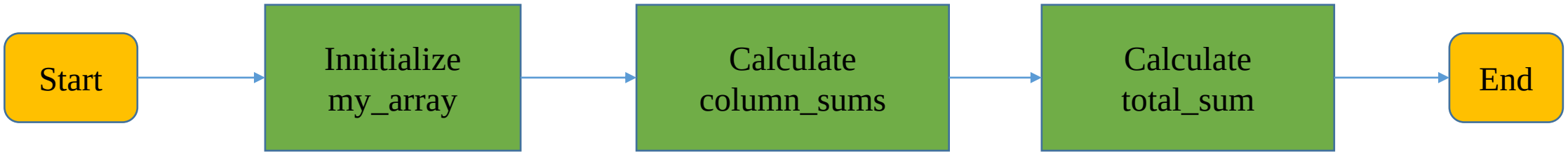


2D array - Advanced

Bài 2: Bạn hãy xây dựng chương trình tính tổng các phần tử trong mỗi cột và tính tổng của các tổng cột?

- Input: my_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
- Output: column_sums = [12, 15, 18], total_sum = 45

Ý tưởng:



2D array - Advanced

Bài 3: Bạn hãy xây dựng chương trình tìm phần tử lớn nhất và vị trí của phần tử đó trong danh sách 2D?

- Input: my_list = [[1, 2, 3], [4, 9, 6], [7, 8, 5]]
- Output: max_value = 9; max_row, max_col = 1, 1

Ý tưởng:

Cấu trúc cần nhớ!

```
np.unravel_index(np.argmax(array), array.shape)
```