Data Structure

List

Quang-Vinh Dinh
Ph.D. in Computer Science

Outline

Abstract Data Types

```
1  # create a set
2  animals = {"cat", "dog", "tiger"}
3
4  print(type(animals))
5  print(animals)
```

```
<class 'set'>
{'dog', 'cat', 'tiger'}
```

tuple_name = (element-1, …, element-n)

List

data = [4, 5, 6, 7, 8, 9]

index   0   1   2   3   4   5

| 4 | 5 | 6 | 7 | 8 | 9 |

bracket

dictionary_name = {key-1:value-1, …, key-n:value-n}

element

comma

colon

Strings

name = | 'A' | 'I' |

index   0   1

1

String

❖ Create and iterate a string

name = 'AI'

name = | A | I |

index   0   1

```
1  # create a string
2  name = 'AI'
3  print(name)
```

AI

```
1  # iterate a string
2  name = 'AI'
3  for character in name:
4      print(character)
```

A
I

```
1  # iterate a string
2  name = 'AI'
3  length = 2
4
5  for index in range(length):
6      print(name[index])
```

A
I

List

❖ **A container that can contain elements**

```
// create a list
data = [6, 5, 7, 1, 9, 2]
```

list_name = [element-1, …, element-n]

data =
6	5	7	1	9	2

index   0   1   2   3   4   5

```
1.    # danh sách trống
2.    emty_list = []
3.
4.    # danh sách số tự nhiên nhỏ hơn 10
5.    my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
6.
7.    # danh sách kết hợp nhiều kiểu dữ liệu
8.    mixedList = [True, 5, 'some string', 123.45]
9.    n_list = ["Happy", [2,0,1,5]]
10.
11.   #danh sách các loại hoa quả
12.   shoppingList = ['táo', 'chuối', 'cherries', 'dâu', 'mận']
```

List

❖ Index

data = [4, 5, 6, 7, 8, 9]

Forward index

0	1	2	3	4	5
4	5	6	7	8	9

Backward index

-6	-5	-4	-3	-2	-1

data[0]

4

data[3]

7

data[-1]

9

data[-3]

7

❖ Slicing

list[start:end:step]

data = [4, 5, 6, 7, 8, 9]

4	5	6	7	8	9

Forward index

0	1	2	3	4	5

data[:3]

4	5	6

data[2:4]

6	7

data[3:]

7	8	9

Giá trị mắc định của start là 0, của end là len(list), và của step là 1

List

❖ Add an element

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.append(4) # thêm 4 vào vị trí cuối list

data = | 6 | 5 | 7 | 1 | 9 | 2 | 4 |

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.insert(0, 4) # thêm 4 vào vị trí có
                     # index = 0

data = | 4 | 6 | 5 | 7 | 1 | 9 | 2 |

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.append(4)
4  print(data)
```

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

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.insert(0, 4)
4  print(data)
```

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

5

List

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

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

```
1  data = [6, 5, 7, 1]
2  print(data)
3  data.extend([9, 2])
4  print(data)
```

```
[6, 5, 7, 1]
[6, 5, 7, 1, 9, 2]
```

❖ **Updating an element**

data = | 6 | 5 | 7 | 1 | 9 | 2 |

thay đổi phần tử thứ 1
data[1] = 4

data = | 6 | 4 | 7 | 1 | 9 | 2 |

❖ **Add a list of elements**

data = | 6 | 5 | 7 | 1 |

data.extend([9, 2]) # thêm 9 và 2 vào vị trí cuối list

data = | 6 | 5 | 7 | 1 | 9 | 2 |

6

List

❖ **+ and * operators**

data1 = | 6 | 5 | 7 |

data2 = | 1 | 9 | 2 |

nối 2 list
data = data1 + data2

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data = | 6 | 5 |

nhân list với một số nguyên
data_m = data * 3

data_m = | 6 | 5 | 6 | 5 | 6 | 5 |

```
1  data1 = [6, 5, 7]
2  data2 = [1, 9, 2]
3
4  # concatenate
5  data = data1 + data2
6  print(data)
```

```
[6, 5, 7, 1, 9, 2]
```

```
1  data = [6, 5]
2
3  # multiply with a number
4  data_m = data*3
5  print(data_m)
```

```
[6, 5, 6, 5, 6, 5]
```

List

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.sort()
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[1, 2, 5, 6, 7, 9]
```

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.sort(reverse = True)
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[9, 7, 6, 5, 2, 1]
```

❖ **sort() – Sắp xếp các phần tử**

data =

6	5	7	1	9	2

data.sort()

data =

1	2	5	6	7	9

data =

6	5	7	1	9	2

data.sort(reverse = True)

data =

9	7	6	5	2	1

8

List

❖ **Deleting an element**

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.pop(2) # tại vị trí index = 2

data = | 6 | 5 | 1 | 9 | 2 |

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.remove(5) # xóa phần tử đầu tiên
có giá trị là 5

data = | 6 | 7 | 1 | 9 | 2 |

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.pop(2)  # by index
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[6, 5, 1, 9, 2]
```

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.remove(2) # by value
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[6, 5, 7, 1, 9]
```

```
1  data = [6, 5, 2, 1, 9, 2]
2  print(data)
3  data.remove(2) # by value
4  print(data)
```

```
[6, 5, 2, 1, 9, 2]
[6, 5, 1, 9, 2]
```

List

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  del data[1:3]
5  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[6, 1, 9, 2]
```

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  data.clear()
5  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[]
```

❖ **Delete elements**

data = | 6 | 5 | 7 | 1 | 9 | 2 |

xóa phần tử thứ 1 và 2
del data[1:3]

data = | 6 | 1 | 9 | 2 |

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.clear()

data = []

10

List

index() – Trả về vị trí đầu tiên

data = | 6 | 5 | 7 | 1 | 9 | 2 |

\# trả về vị trí của phần tử đầu tiên có giá trị là 9
data.index(9) = 4

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  indexOf9 = data.index(9)
5  print(indexOf9)
```

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

reverse() – Đảo ngược vị trí các phần tử

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data.reserse()

data = | 2 | 9 | 1 | 7 | 5 | 6 |

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  data.reverse()
5  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[2, 9, 1, 7, 5, 6]
```

List

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  numOf7 = data.count(7)
5  print(numOf7)
```

```
[6, 5, 7, 1, 9, 2]
1
```

count() – Trả về số lần xuất hiện của một phần tử

data = | 6 | 5 | 7 | 1 | 9 | 2 |

\# trả về số lần phần tử 7 xuất hiện trong list
data.count(7) = 1

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  aCopy = data.copy()
5  print(aCopy)
```

```
[6, 5, 7, 1, 9, 2]
[6, 5, 7, 1, 9, 2]
```

copy() – copy một list

data = | 6 | 5 | 7 | 1 | 9 | 2 |

data_copy = data.copy()

data_copy = | 6 | 5 | 7 | 1 | 9 | 2 |

Built-in Functions for List

len(), min(), and max()

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
```

```
[6, 5, 7, 1, 9, 2]
```

data =

6	5	7	1	9	2

\# trả về số phần tử
len(data) = 6

```
1  # get a number of elements
2  length = len(data)
3  print(length)
```

```
6
```

\# trả về số phần tử có giá trị nhỏ nhất
min(data) = 1

\# trả về số phần tử có giá trị lớn nhất
max(data) = 9

```
1  # get the min and max values
2  print(min(data))
3  print(max(data))
```

```
1
9
```

Example

Công thức

https://www.mathsisfun.com/calculus/integration-introduction.html
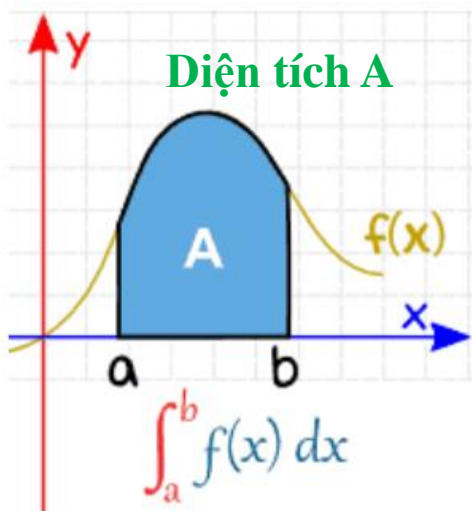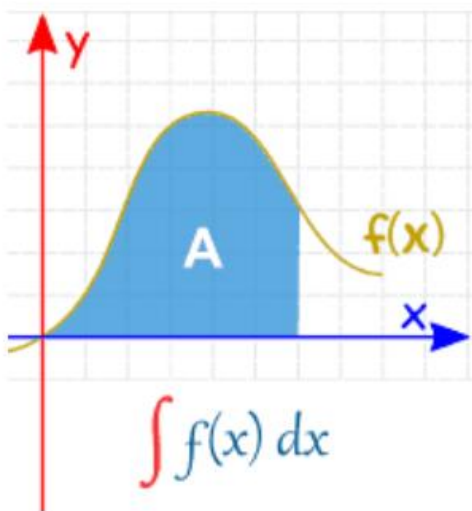
$$F(a) = \int_{-\infty}^{a} f(x) d(x)$$

$$F(b) = \int_{-\infty}^{b} f(x) d(x)$$

Diện tích A

$$A = F(b) - F(a)$$

$$f(x) \geq 0$$

Áp dụng cho hàm rời rạc (1D)

Diện tích A

$f(x)$	1	8	5	7	3	5	8	3	2	9

| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

$$F(3) = \sum_{x \leq 3} f(x) = f(0) + f(1) + f(2) + f(3)$$
$$= 1 + 8 + 5 + 7 = 21$$

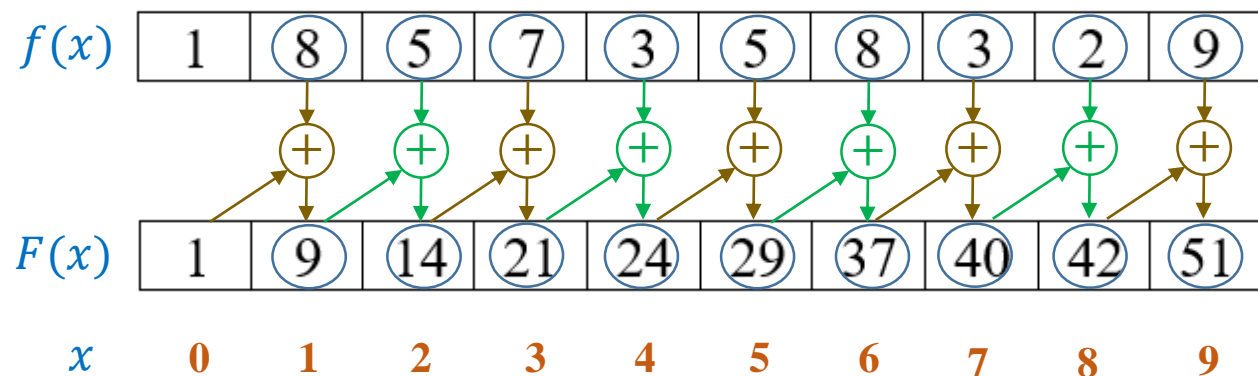$$F(6) = \sum_{x \leq 6} f(x) = 1 + 8 + 5 + 7 + 3 + 5 + 8 = 37$$

$$A = F(6) - F(3) = \sum_{4 \leq x \leq 6} f(x) = 3 + 5 + 8 = 16$$

14

Example

Áp dụng cho hàm rời rạc (1D)

Diện tích A

| 1 | 8 | 5 | 7 | 3 | 5 | 8 | 3 | 2 | 9 |

x 0   1   2   3   4   5   6   7   8   9

$$F(3) = \sum_{x \leq 3} f(x) = f(0) + f(1) + f(2) + f(3)$$
$$= 1 + 8 + 5 + 7 = 21$$

$$F(6) = \sum_{x \leq 6} f(x) = 1 + 8 + 5 + 7 + 3 + 5 + 8 = 37$$

$$A = F(6) - F(3) = \sum_{4 \leq x \leq 6} f(x) = 3 + 5 + 8 = 16$$

Tính chất

$$F(x) = f(x) + F(x-1)$$
$$F(7) = f(7) + F(6) = 3 + 37 = 40$$

Xây dựng integral array dùng tính chất $F(x) = f(x) + F(x-1)$

$f(x)$

| 1 | 8 | 5 | 7 | 3 | 5 | 8 | 3 | 2 | 9 |

$F(x)$

| 1 | 9 | 14 | 21 | 24 | 29 | 37 | 40 | 42 | 51 |

x 0   1   2   3   4   5   6   7   8   9

Tính tổng với độ phức tạp ~ O(1)

$$\sum_{a \leq x \leq b} f(x) = F(b) - F(a-1)$$

$$\sum_{4 \leq x \leq 6} f(x) = F(6) - F(3) = 37 - 21 = 16$$

15

Built-in Functions

❖ **sorted(aList) – Sắp xếp các phần tử**

sorted(iterable, reverse=reverse)

data = | 6 | 5 | 7 | 1 | 9 | 2 |

sorted_data = sorted(data)

sorted_data = | 1 | 2 | 5 | 6 | 7 | 9 |

data = | 6 | 5 | 7 | 1 | 9 | 2 |

sorted_data = sorted(data, reverse=True**)**

sorted_data = | 9 | 7 | 6 | 5 | 2 | 1 |

```
1  # sorted
2  data = [6, 5, 7, 1, 9, 2]
3  print(data)
4
5  sorted_data = sorted(data)
6  print(sorted_data)
```

```
[6, 5, 7, 1, 9, 2]
[1, 2, 5, 6, 7, 9]
```

```
1  # sorted
2  data = [6, 5, 7, 1, 9, 2]
3  print(data)
4
5  sorted_data = sorted(data, reverse=True)
6  print(sorted_data)
```

```
[6, 5, 7, 1, 9, 2]
[9, 7, 6, 5, 2, 1]
```

16

Built-in Functions

sum()

$$summation = \sum_{i=0}^{n} data_i$$
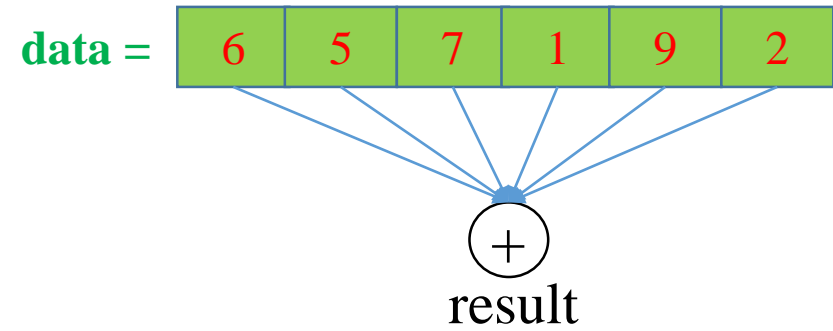
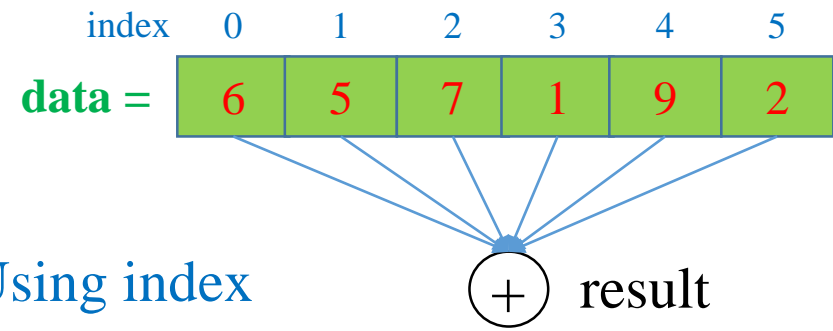data = | 6 | 5 | 7 | 1 | 9 | 2 |

tính tổng
sum(data) = 30

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  summation = sum(data)
5  print(summation)
```

```
[6, 5, 7, 1, 9, 2]
30
```

data = | 6 | 5 | 7 | 1 | 9 | 2 |

$+$

result

```
1  # custom summation - way 1
2  def computeSummation(data):
3      result = 0
4
5      for value in data:
6          result = result + value
7
8      return result
9
10 # test
11 data = [6, 5, 7, 1, 9, 2]
12 summation = computeSummation(data)
13 print(summation)
```

```
30
```

Built-in Functions

data =

6	5	7	1	9	2

sum()

$$summation = \sum_{i=0}^{n} data_i$$

data =

6	5	7	1	9	2

\# tính tổng

sum(data) = 30

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3
4  summation = sum(data)
5  print(summation)
```

```
[6, 5, 7, 1, 9, 2]
30
```

Using index

+ result
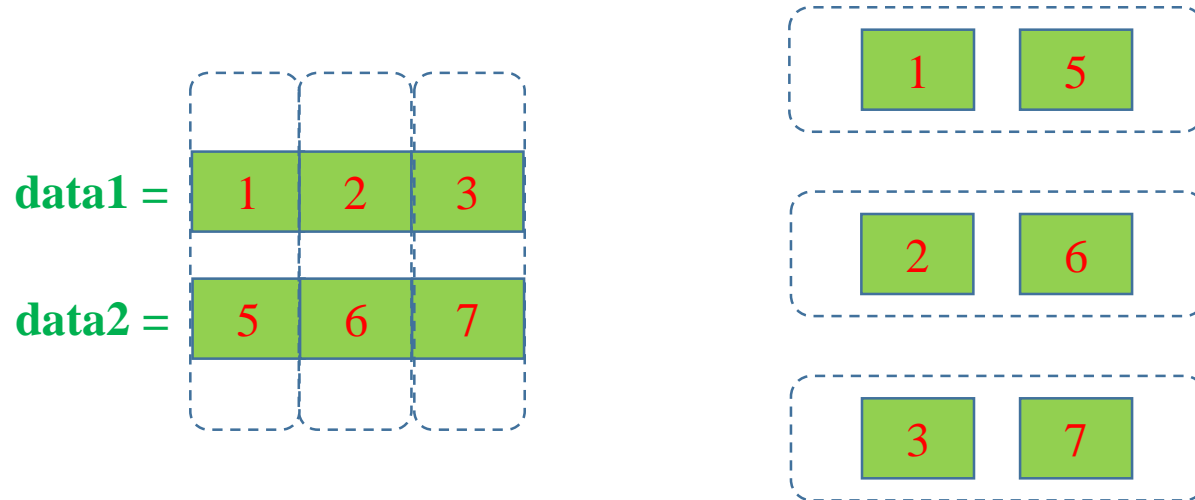
```
1   # custom summation - way 2
2   def computeSummation(data):
3       result = 0
4
5       length = len(data)
6       for index in range(length):
7           result = result + data[index]
8
9       return result
10
11  # test
12  data = [6, 5, 7, 1, 9, 2]
13  summation = computeSummation(data)
14  print(summation)
```

```
30
```

18

Built-in Functions

zip()

data1 = | 1 | 2 | 3 |

data2 = | 5 | 6 | 7 |

| 1 | 5 |

| 2 | 6 |

| 3 | 7 |

```python
1  l1 = [1, 2, 3]
2  l2 = [5, 6, 7]
3
4  # print in pairs
5  length = len(l1)
6  for i in range(length):
7      print(l1[i], l2[i])
```

```
1 5
2 6
3 7
```

```python
1  l1 = [1, 2, 3]
2  l2 = [5, 6, 7]
3
4  # print in pairs
5  for v1, v2 in zip(l1, l2):
6      print(v1, v2)
```

```
1 5
2 6
3 7
```

Built-in Functions

reversed()

data = | 6 | 1 | 7 |

reversed(data) = | 7 | 1 | 6 |

```python
1  # for and list
2  data = [6, 1, 7]
3  for value in data:
4      print(value)
```

```
6
1
7
```

```python
1  # reversed
2  data = [6, 1, 7]
3  for value in reversed(data):
4      print(value)
```

```
7
1
6
```

Built-in Functions

enumerate()

data =

6	1	7

enumerate(data) =

6	1	7

index   0   1   2

```python
# get index and value
data = [6, 1, 7]

length = len(data)
for index in range(length):
    print(index, data[index])
```

```
0 6
1 1
2 7
```

```python
# enumerate
data = [6, 1, 7]
for index, value in enumerate(data):
    print(index, value)
```

```
0 6
1 1
2 7
```

Examples

Sum of even numbers

data = | 6 | 5 | 7 | 1 | 9 | 2 |

```python
1   # sum of even number
2   def sum1(data):
3       result = 0
4
5       for value in data:
6           if value%2 == 0:
7               result = result + value
8
9       return result
10
11  # test
12  data = [6, 5, 7, 1, 9, 2]
13  summation = sum1(data)
14  print(summation)
```

8

Sum of elements with even indices

data = | 6 | 5 | 7 | 1 | 9 | 2 |

```python
1   # sum of numbers with even indices
2   def sum2(data):
3       result = 0
4
5       length = len(data)
6       for index in range(length):
7           if index%2 == 0:
8               result = result + data[index]
9
10      return result
11
12  # test
13  data = [6, 5, 7, 1, 9, 2]
14  summation = sum2(data)
15  print(summation)
```

22

Examples

square(aList)

data =

6	5	7	1	9	2

square(data) =

36	25	49	1	81	4

```python
1  # square function
2  def square(data):
3      result = []
4
5      for value in data:
6          result.append(value*value)
7
8      return result
9
10 # test
11 data = [6, 5, 7, 1, 9, 2]
12 print(data)
13 data_s = square(data)
14 print(data_s)
```

```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```

List Comprehension

```
1   # square function
2   def square(data):
3       result = []
4
5       for value in data:
6           result.append(value*value)
7
8       return result
```

omitted

```
1   # using list comprehension
2   def square(data):
3       result = [value*value for value in data]
4
5       return result
```

added

```
1   # using list comprehension
2   def square(data):
3       result = [value*value for value in data]
4
5       return result
6
7   # test
8   data = [6, 5, 7, 1, 9, 2]
9   print(data)
10  data_s = square(data)
11  print(data_s)
```

```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```

Sigmoid Function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$
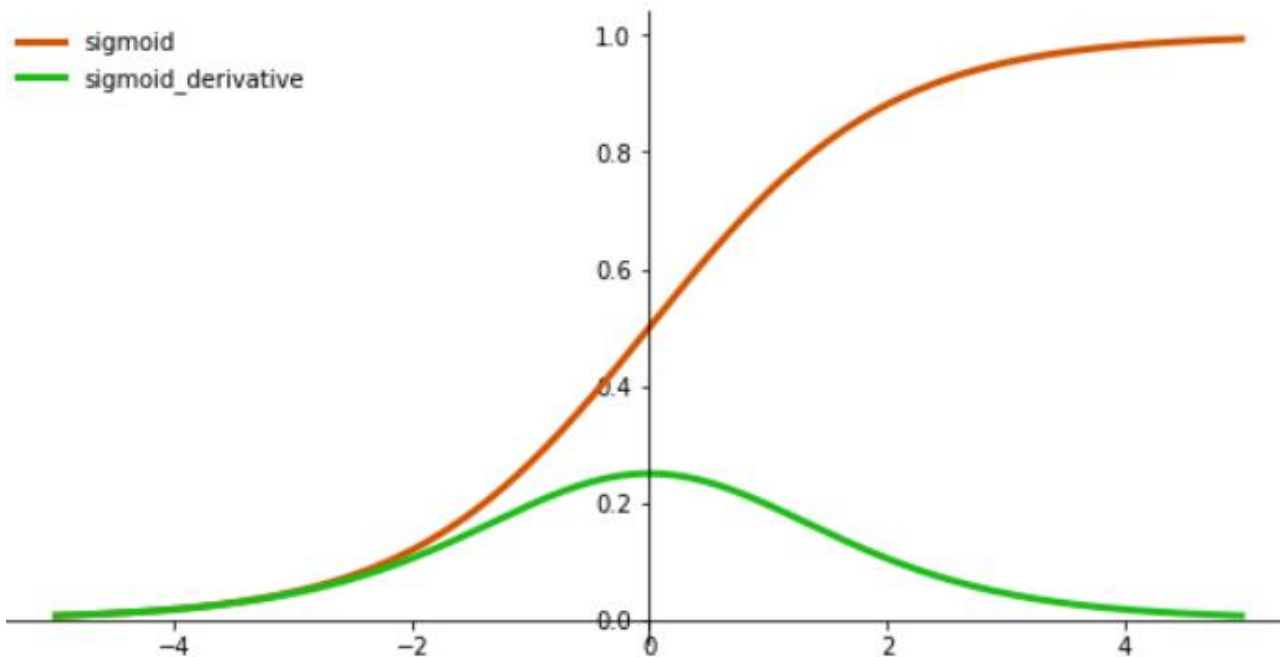
$$\sigma'(x) = \sigma(x)(1 - \sigma(x))$$

data =

1	5	-4	3	-2

data_a = **sigmoid(data)**

data_a =

0.731	0.993	0.017	0.95	0.119

— sigmoid
— sigmoid_derivative

List Comprehension

```python
import math

# sigmoid function
def sigmoid(x):
    result = 1 / (1 + math.exp(-x))
    return result


def sigmoidForList(data):
    result = [sigmoid(x) for x in data]
    return result


# test
data = [1, 5, -4, 3, -2]
print(data)
data_a = sigmoidForList(data)
print(data_a)
```

ReLU Function

$$\text{ReLU}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

$$\text{ReLU}'(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases}$$
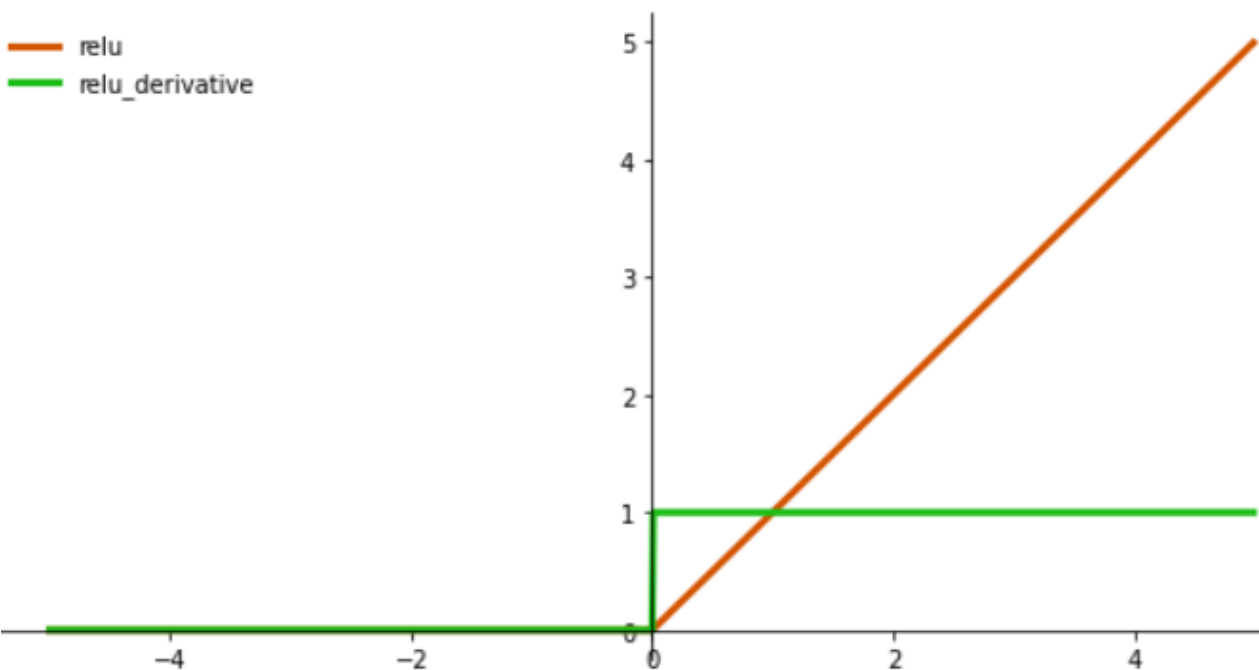
data =

1	5	-4	3	-2

data_a = ReLU(data)

data_a =

1	5	0	3	0

— relu
— relu_derivative

List Comprehension

```python
1  def relu(x):
2      result = 0
3      if x > 0:
4          result = x
5
6      return result
7
8  def reluForList(data):
9      result = [relu(x) for x in data]
10     return result
11
12 # test
13 data = [1, 5, -4, 3, -2]
14 print(data)
15 data_a = reluForList(data)
16 print(data_a)
```

```
[1, 5, -4, 3, -2]
[1, 5, 0, 3, 0]
```

26

ReLU Function

$$\text{ReLU}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

$$\text{ReLU}'(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases}$$
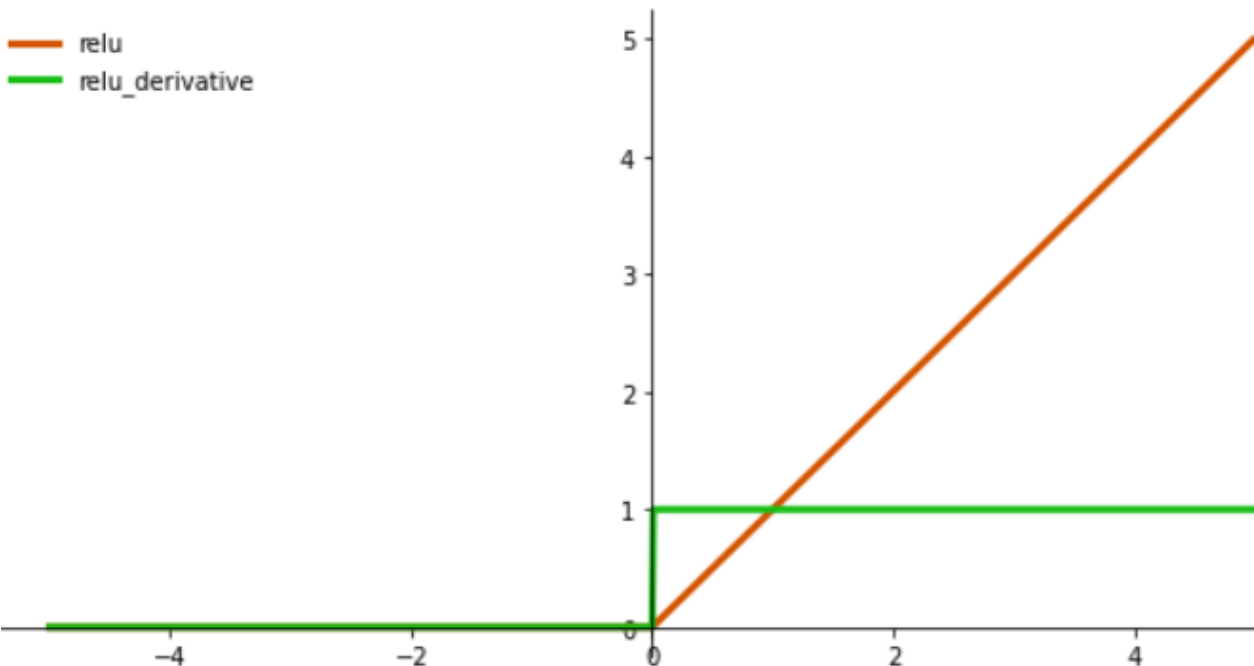
data =

1	5	-4	3	-2

data_a = ReLU(data)

data_a =

1	5	0	3	0

— relu
— relu_derivative

List Comprehension

```
2        result = 0
3        if x > 0:
4            result = x
```

```
1  # relu function
2  def relu(data):
3      result = [x if x>0 else 0 for x in data]
4      return result
5
6  # test
7  data = [1, 5, -4, 3, -2]
8  print(data)
9  data_a = relu(data)
10 print(data_a)
```

```
[1, 5, -4, 3, -2]
[1, 5, 0, 3, 0]
```

List Comprehension

[condition_to_branch_x for x in data condition_to_filter_x]

```
1  # quiz 1
2  data = [1, 5, -4, 3, -2]
3  print(data)
4
5  data_a = [x if x>0 else 0 for x in data]
6  print(data_a)
```

```
1  # quiz 3
2  data = [1, 5, -4, 3, -2]
3  print(data)
4
5  data_a = [x for x in data if x>0]
6  print(data_a)
```

```
1  # quiz 2
2  data = [1, 5, -4, 3, -2]
3  print(data)
4
5  data_a = [x if x>0 for x in data]
6  print(data_a)
```

```
1  # quiz 4
2  data = [1, 5, -4, 3, -2]
3  print(data)
4
5  data_a = [x for x in data if x>0 else 0]
6  print(data_a)
```

List Sorting

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.sort()
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[1, 2, 5, 6, 7, 9]
```

```
1  data = [6, 5, 7, 1, 9, 2]
2  print(data)
3  data.sort(reverse = True)
4  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[9, 7, 6, 5, 2, 1]
```

```
1  # sorted
2  data = [6, 5, 7, 1, 9, 2]
3  print(data)
4
5  sorted_data = sorted(data)
6  print(sorted_data)
```

```
[6, 5, 7, 1, 9, 2]
[1, 2, 5, 6, 7, 9]
```

```
1  # sorted
2  data = [6, 5, 7, 1, 9, 2]
3  print(data)
4
5  sorted_data = sorted(data, reverse=True)
6  print(sorted_data)
```

```
[6, 5, 7, 1, 9, 2]
[9, 7, 6, 5, 2, 1]
```

29

Mutable and Immutable

```python
1   # immutable
2   def square(data):
3       result = []
4       for value in data:
5           result.append(value*value)
6
7       return result
8
9   # test
10  data = [6, 5, 7, 1, 9, 2]
11  print(data)
12
13  data_s = square(data)
14  print(data_s)
```
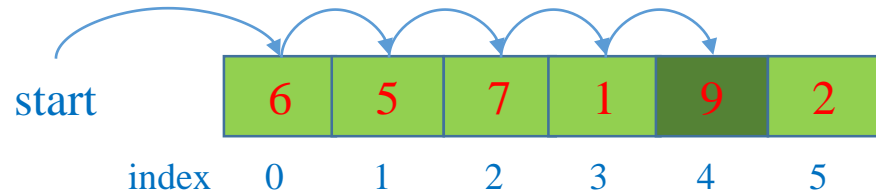
```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```

```python
1   # mutable
2   def square(data):
3       length = len(data)
4
5       for i in range(length):
6           value = data[i]
7           data[i] = value*value
8
9   # test
10  data = [6, 5, 7, 1, 9, 2]
11  print(data)
12
13  square(data)
14  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```
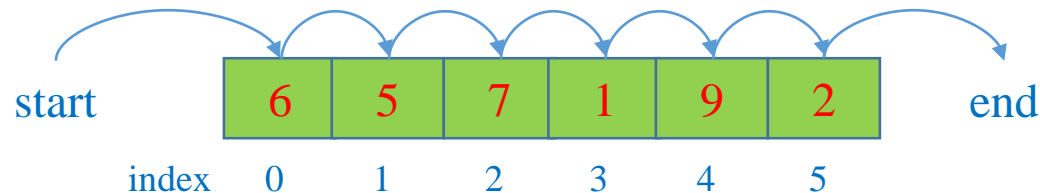
Algorithms on List

❖ **Linear searching**

data =

6	5	7	1	9	2

index   0   1   2   3   4   5

Searching for 9

start

6	5	7	1	9	2

index   0   1   2   3   4   5

Returning 4

Searching for 8

start

6	5	7	1	9	2

end

index   0   1   2   3   4   5

Returning ?

index = -1

If iterating the last item

item ← getItem()

if item==value,
index←getIndex()

Returning index

31
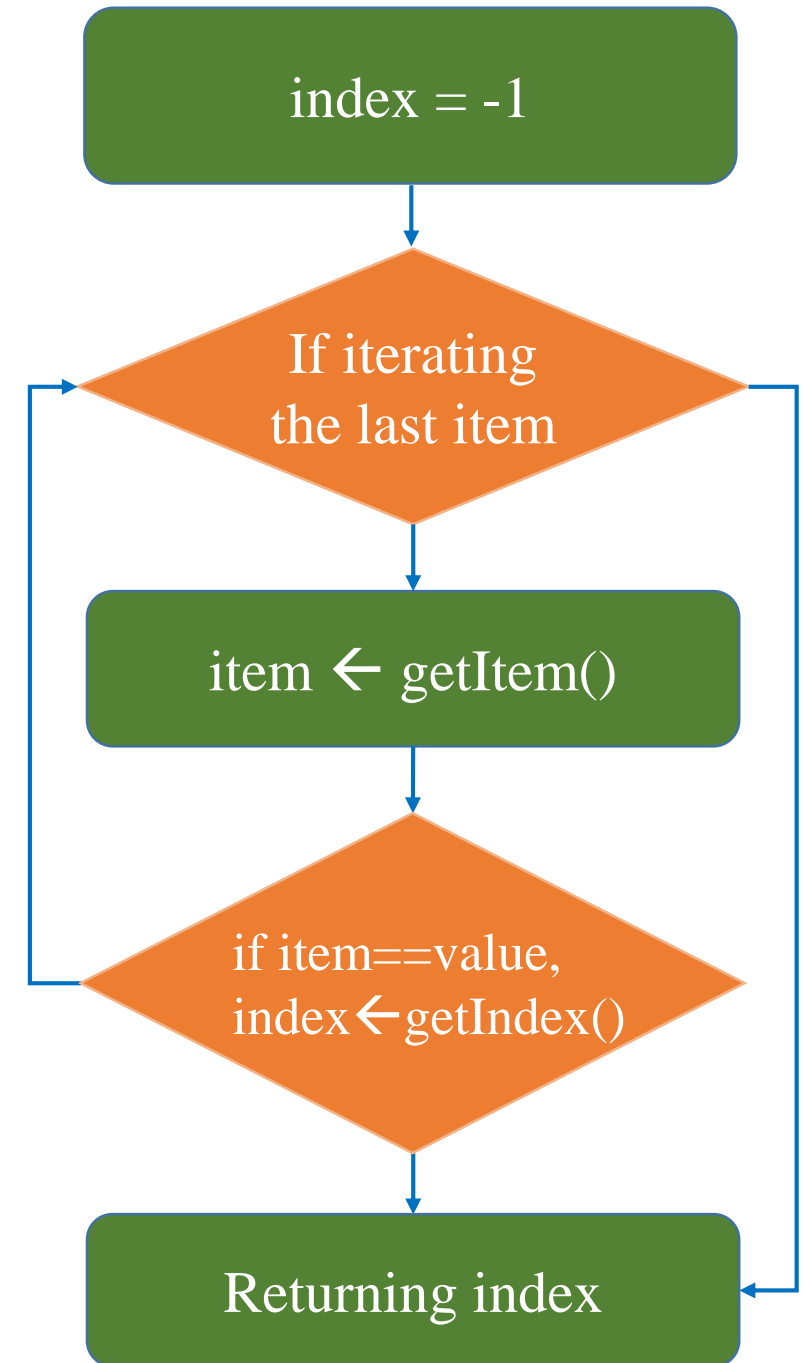
Algorithms on List

❖ **Sorting using min(), remove(), and append()**

data = | 6 | 5 | 7 | 1 | 9 | 2 |

result = [ ]

min(data) = 1

result.append(1) = | 1 |

data.remove(1) = | 6 | 5 | 7 | 9 | 2 |

. . .

```
          ┌─────────────────────┐
          │  if the list is     │
          │  non-empty?         │
          └─────────────────────┘
                    │
          ┌─────────────────────┐
          │  item ← getMin()    │
          └─────────────────────┘
                    │
          ┌─────────────────────┐
          │  append(item) to result │
          └─────────────────────┘
                    │
          ┌─────────────────────┐
          │  remove(item) from list │
          └─────────────────────┘
```
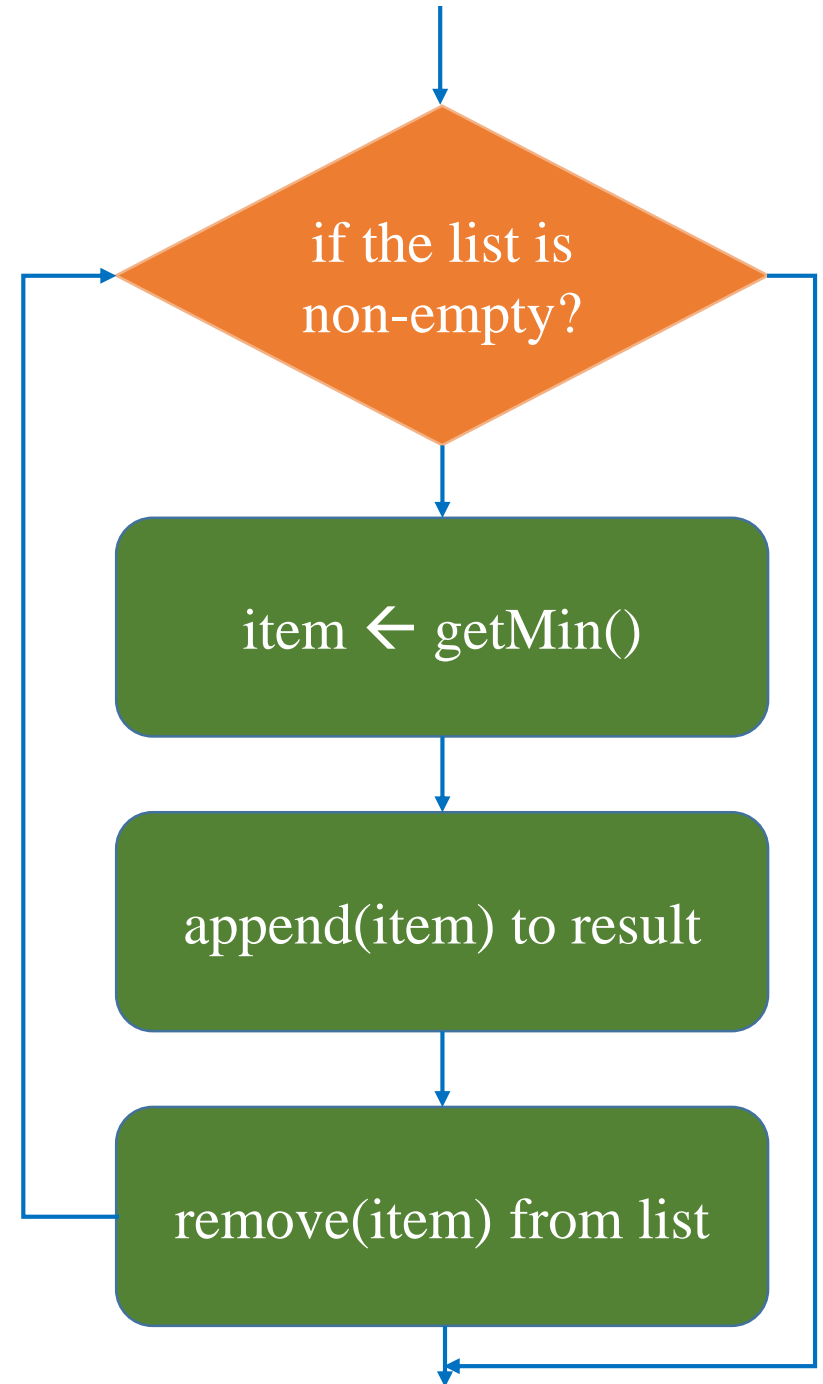
Algorithms on List

❖ **Sorting using min(), remove(), and append()**

data = | 6 | 5 | 7 | 9 | 2 |

result = | 1 |

min(data) = 2

result.append(2) = | 1 | 2 |

data.remove(2) = | 6 | 5 | 7 | 9 |

. . .

if the list is non-empty?

item ← getMin()

append(item) to result

remove(item) from list
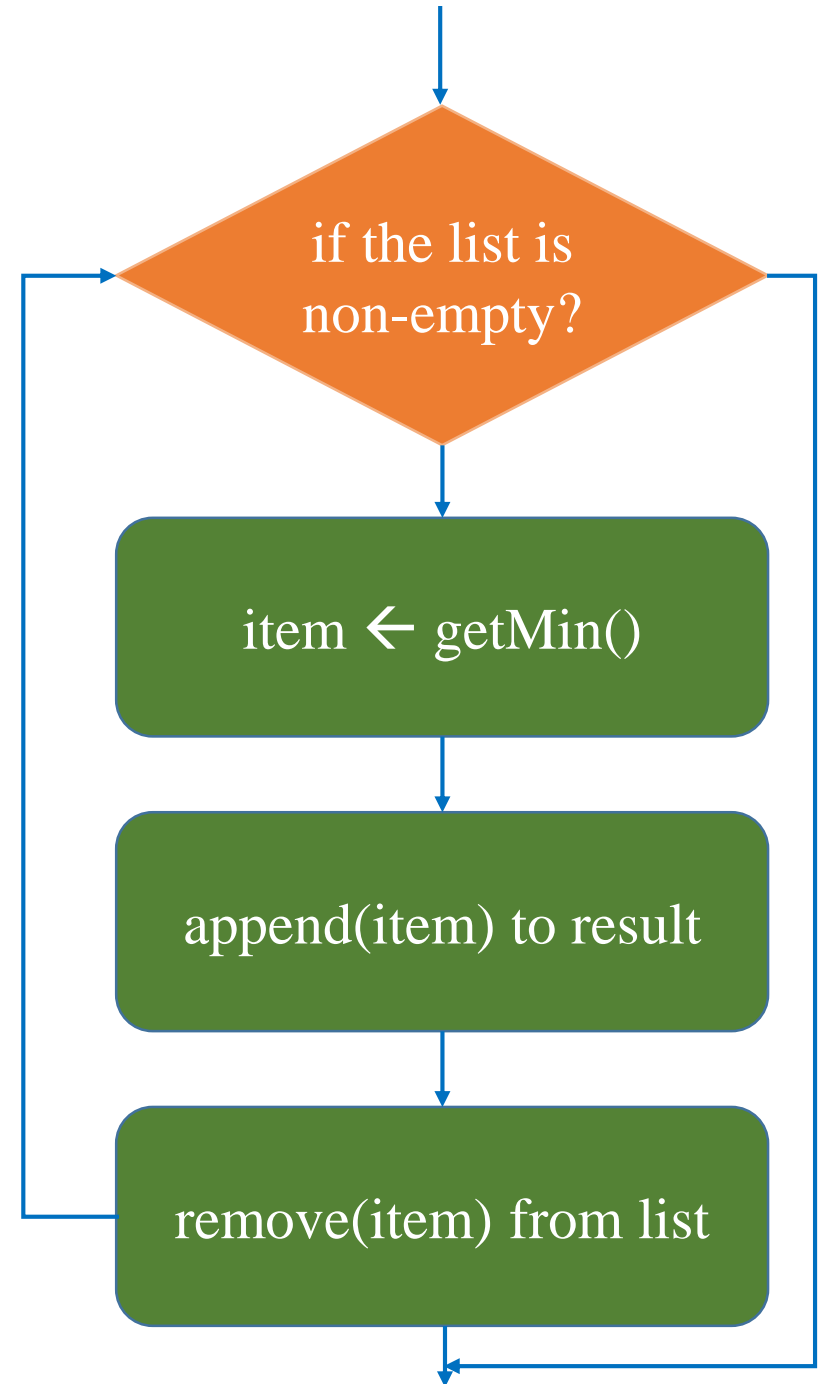
Converting to List

aList ← list(iterable)

name = 'AI'

data =

A	I

index   0   1

---

data = list(range(4, 10)

index

0	1	2	3	4	5

data =

4	5	6	7	8	9

```python
1  name = 'AI'
2  data = list(name)
3
4  print(name)
5  print(data)
```

```
AI
['A', 'I']
```

```python
1  data = list(range(4, 10))
2  print(data)
```

```
[4, 5, 6, 7, 8, 9]
```

Plotting a function

❖ **Using matplotlib**

```
1   # ví dụ 1
2   import matplotlib.pyplot as plt
3
4   # data
5   x_data = [0, 1, 2, 3]
6   y_data = [1, 3, 4, 2]
7
8   # plot
9   plt.plot(x_data, y_data)
10  plt.ylabel('y values')
11  plt.xlabel('x values')
12
13  # show
14  plt.show()
```

x-data

y-data

35
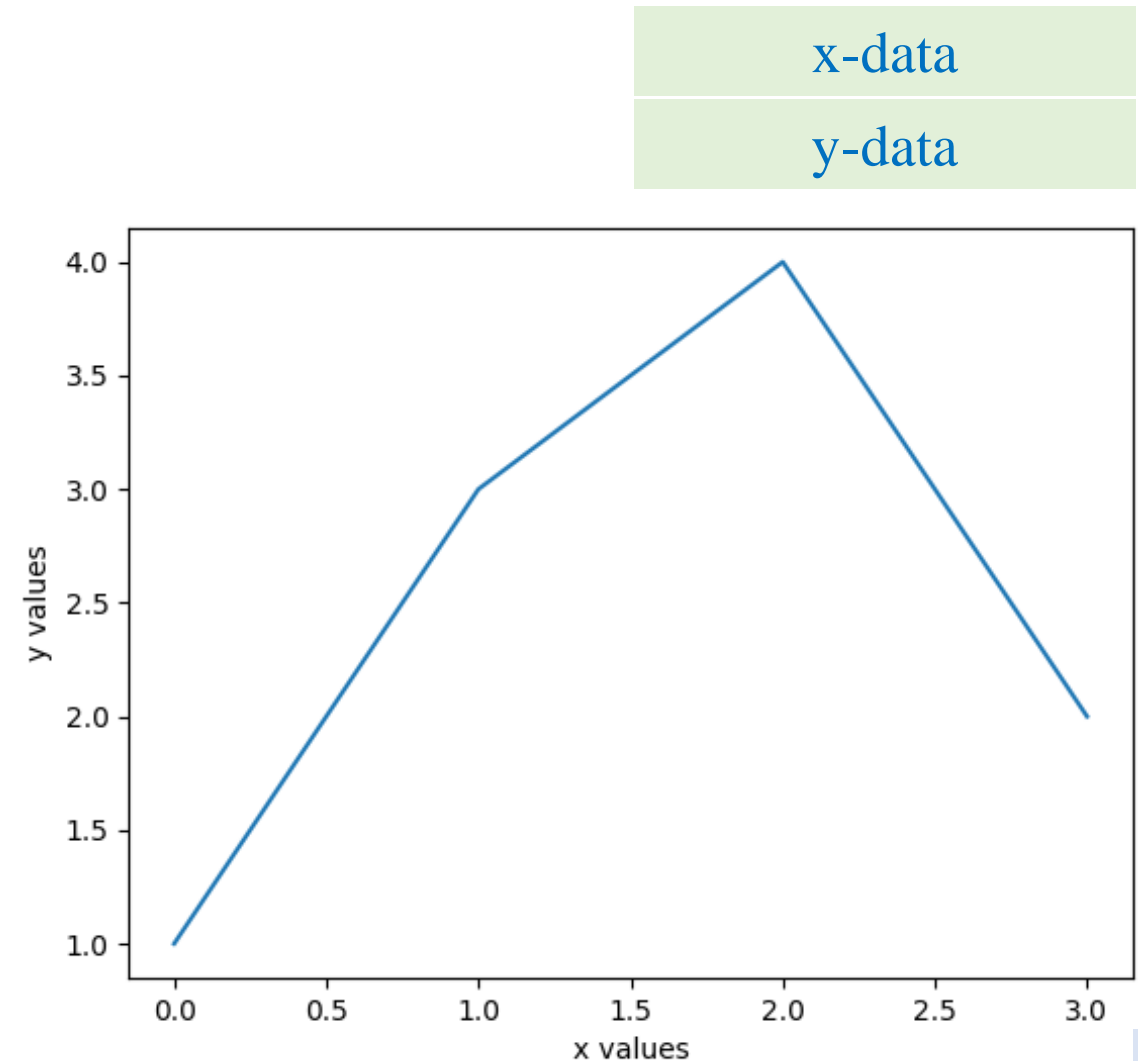
Plotting a function

❖ **Using matplotlib**

$x \in [-8, 8]$

$y = x^2$

How to do it?
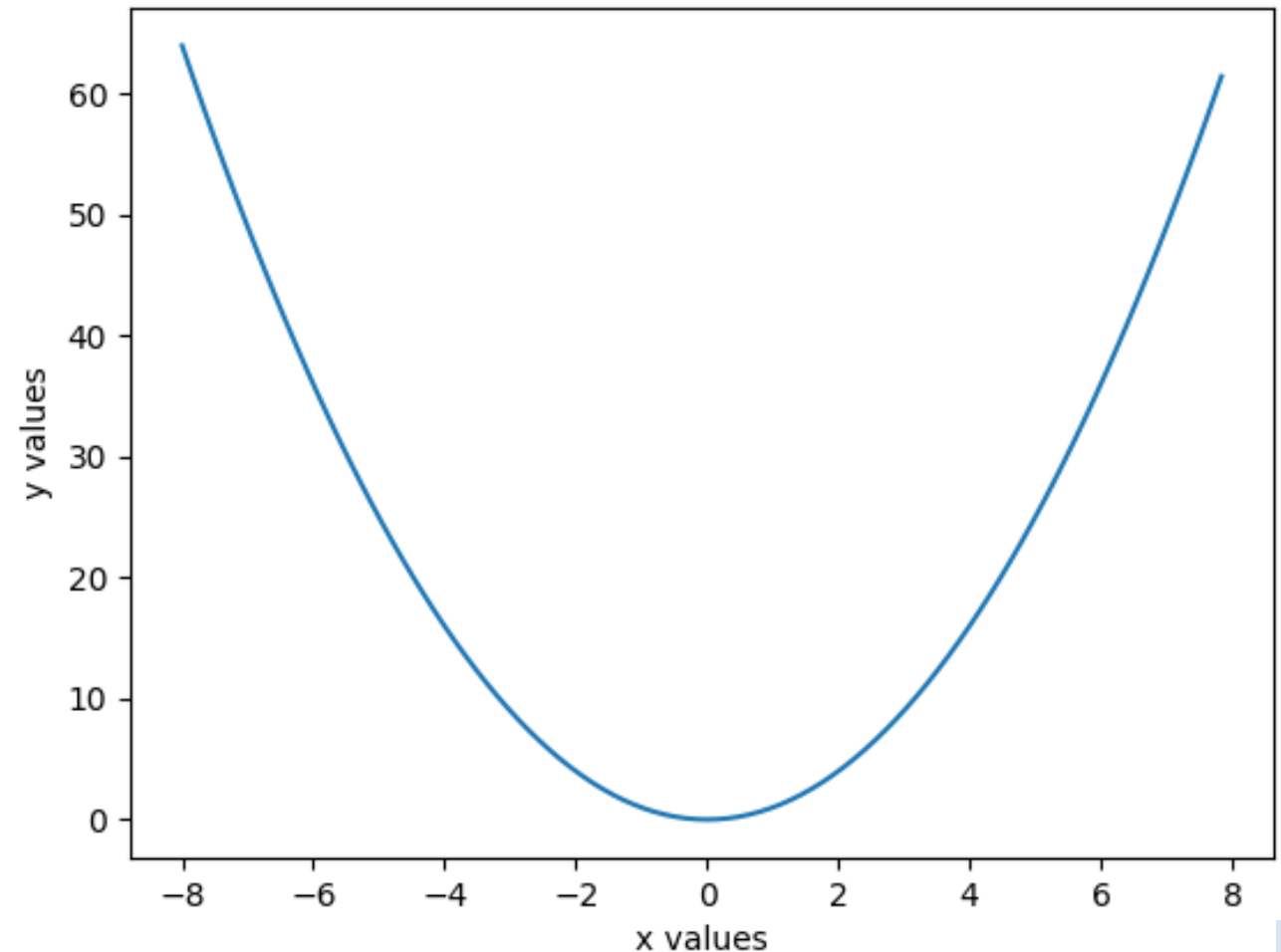
List and File

❖ **Read and Write from/to a log file**

```python
1  # write to a log file
2  with open(r'log.txt', 'w') as fp:
3      for item in data:
4          # write each item on a new line
5          fp.write(f'{item}\n')
```

```python
1  # read from the log file
2  data = []
3  with open(r'log.txt', 'r') as fp:
4      lines = fp.readlines()
5      for line in lines:
6          line = line.strip('\n')
7          data.append(float(line))
```


37
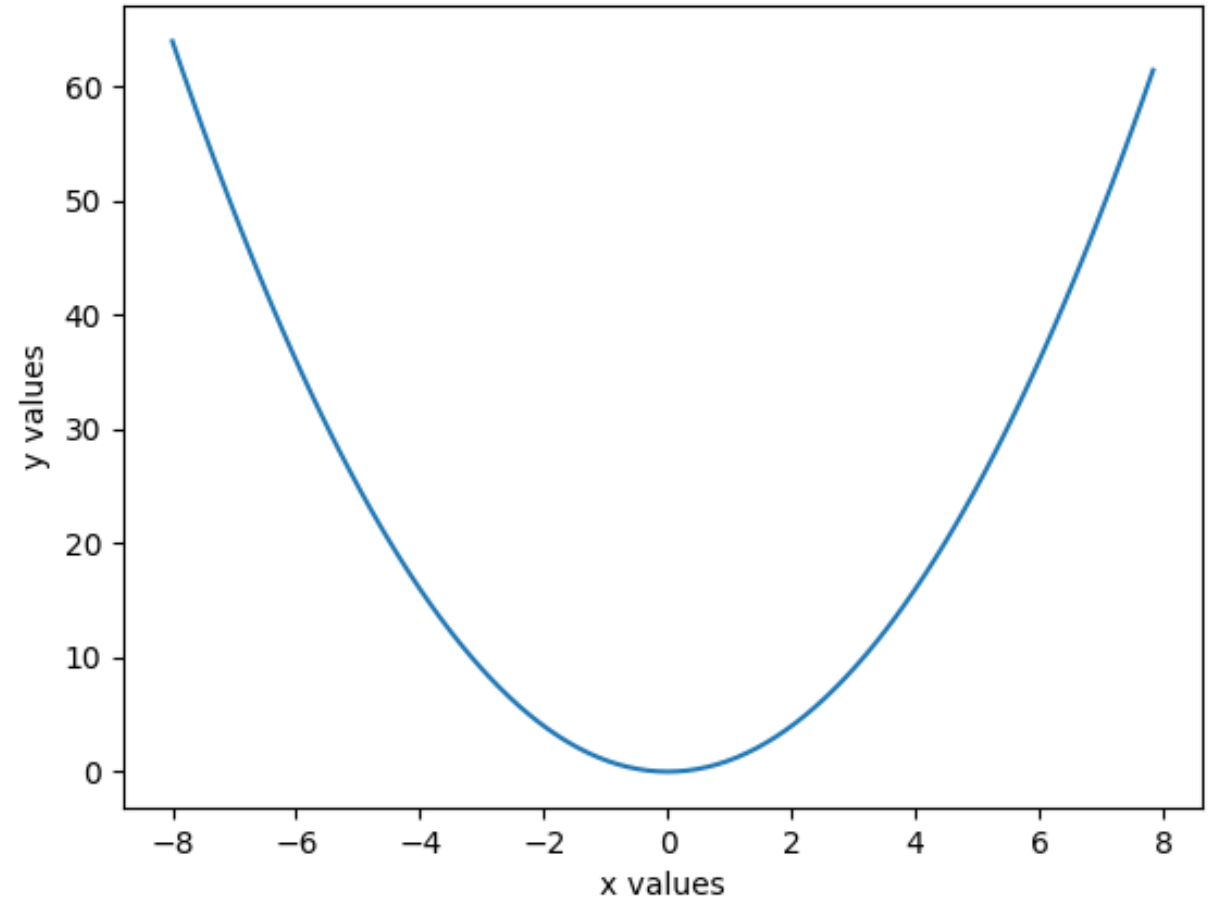
Outline

- ➤ **Introduction**
- ➤ **List**
- ➤ **Some algorithms on List**
- ➤ **Addresses**
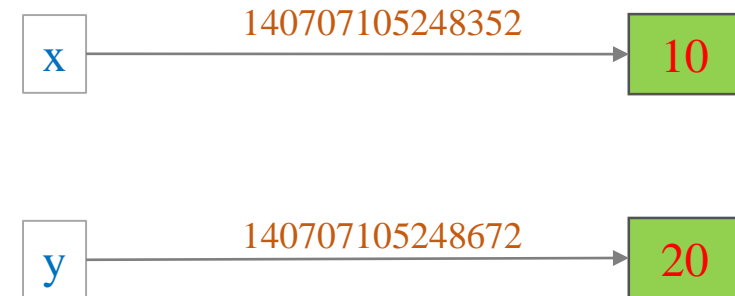- ➤ **Common Errors**

Variables and Addresses

```
1  x = 20
2  y = x
3
4  print(x)
5  print(y)
6  print(id(x))
7  print(id(y))
```
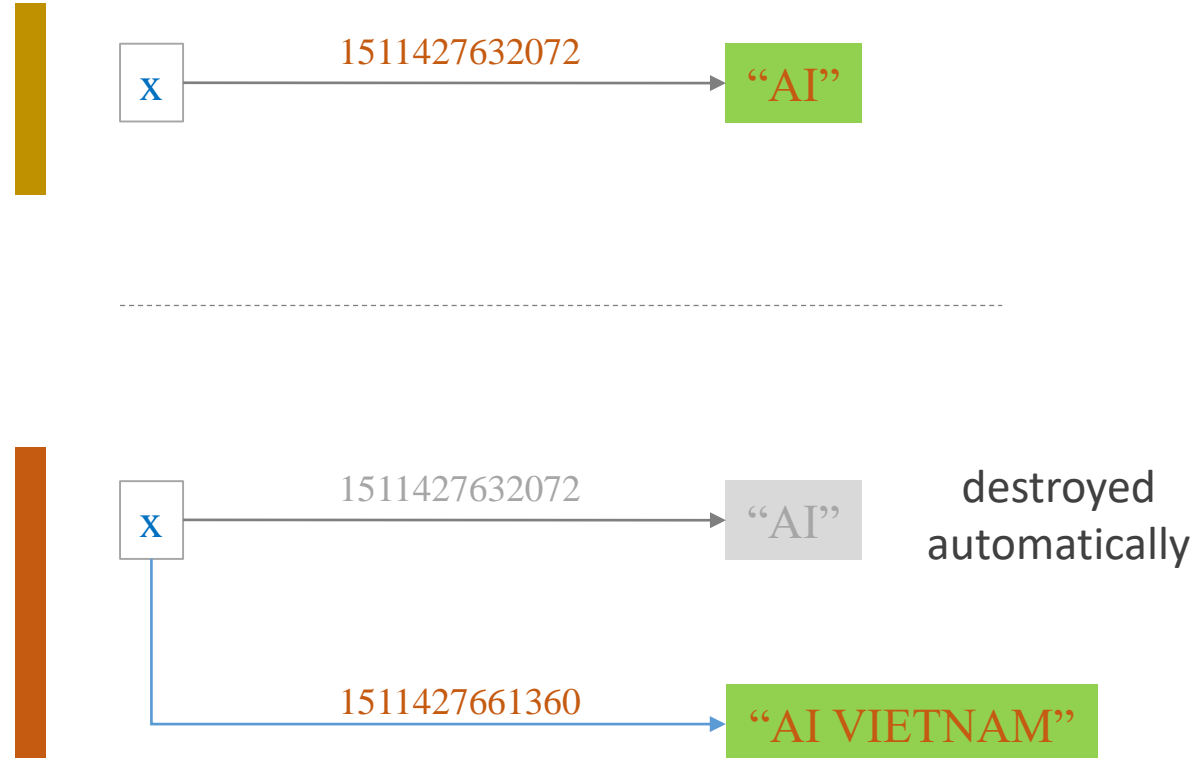
```
20
20
140707105248672
140707105248672
```
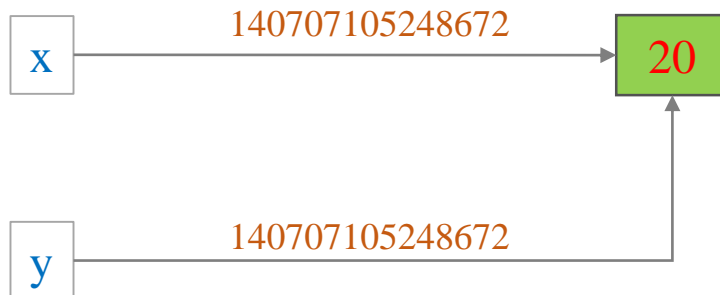
```
1  x = 20
2  y = x
3  x = 10
4
5  print(x)
6  print(y)
7  print(id(x))
8  print(id(y))
```

```
10
20
140707105248352
140707105248672
```

Variables and Addresses

❖ **Immutable types: Cannot be changed in place**

❖ **Including ints, floats, strings, and tuples**
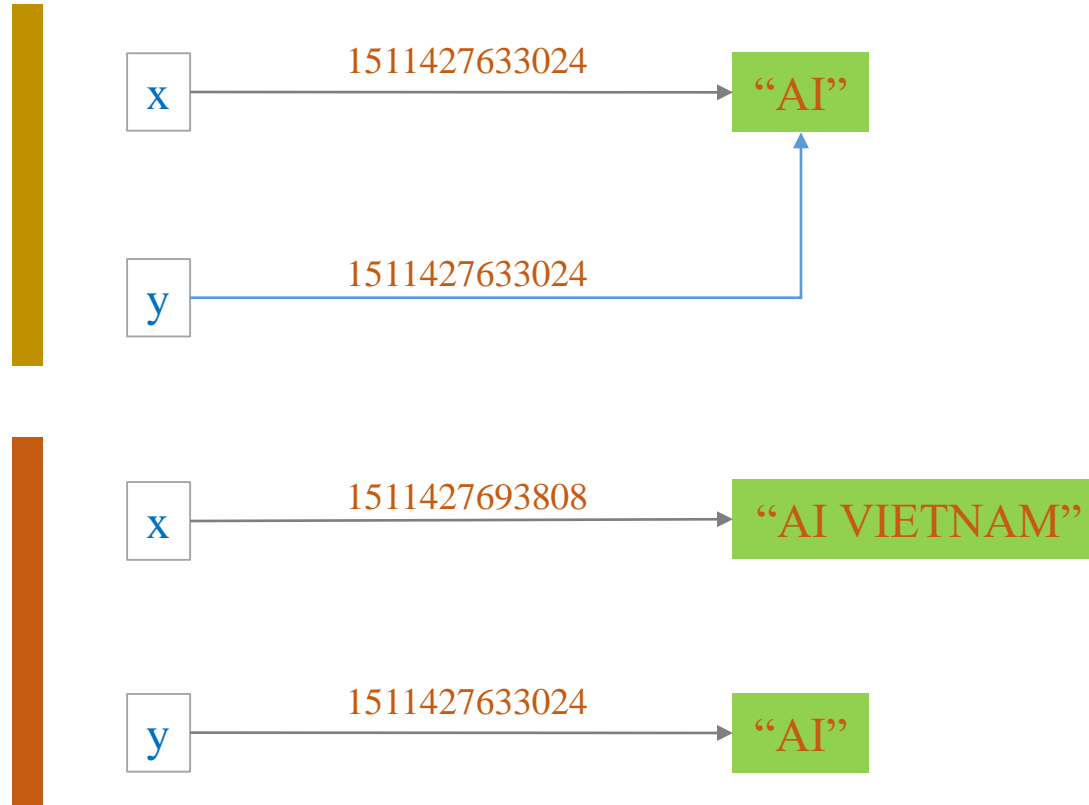


```
1   # immutable
2
3   x = "AI"
4   print(x)
5   print(id(x))
6   print()
7
8   x = x + " VIETNAM"
9   print(x)
10  print(id(x))
```

```
AI
1511427632072
```

```
AI VIETNAM
1511427661360
```

Variables and Addresses

❖ **Immutable types: Cannot be changed in place**

❖ **Including ints, floats, strings, and tuples**


```
1   # immutable
2
3   x = "AI"
4   y = x
5   print('x: ', x)
6   print('y: ', y)
7   print('x address: ', id(x))
8   print('y address: ', id(y))
9   print()
10
11  x = x + " VIETNAM"
12  print('x: ', x)
13  print('x address: ', id(x))
14  print('y: ', y)
15  print('y address: ', id(y))
```

```
x:  AI
y:  AI
x address:  1511427633024
y address:  1511427633024
```

```
x:  AI VIETNAM
x address:  1511427693808
y:  AI
y address:  1511427633024
```

Variables and Addresses

```
1  # aivietnam
2
3  a_list = [1, 2, 3]
4  print('a_list: ', a_list)
5  print('a_list address: ', id(a_list))
```

```
a_list:  [1, 2, 3]
a_list address:  1511426884232
```

```
1  a_list = a_list + [4, 5]
2  print('a_list: ', a_list)
3  print('a_list address: ', id(a_list))
```

```
a_list:  [1, 2, 3, 4, 5]
a_list address:  1511427661128
```
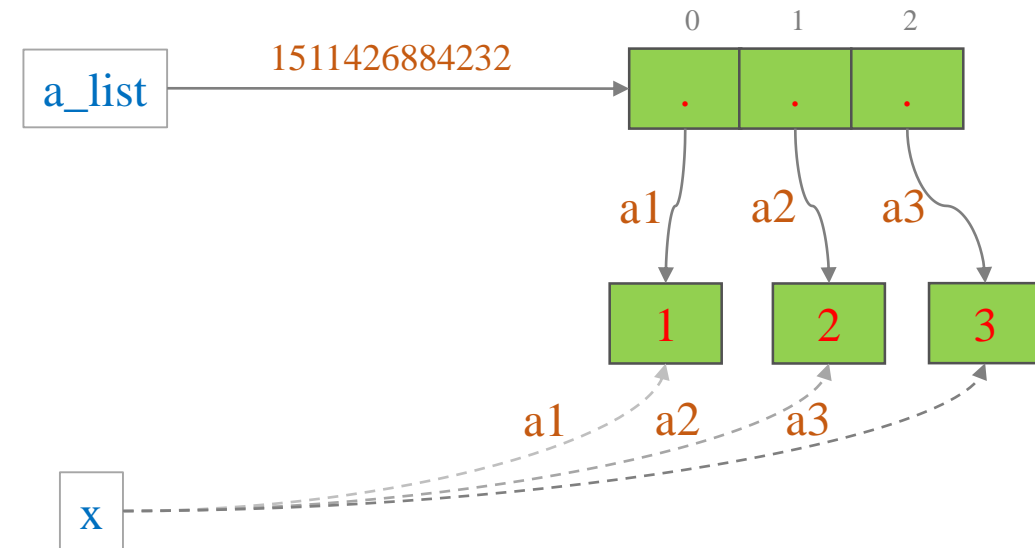
Variables and Addresses

```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4   for i in range(3):
5       print(f'a_list[{i}] address is {id(a_list[i])}')
6
7   print()
8   for x in a_list:
9       print(f'x address is {id(x)}')
```

```
a_list[0] address is 140707105248064
a_list[1] address is 140707105248096
a_list[2] address is 140707105248128

x address is 140707105248064
x address is 140707105248096
x address is 140707105248128
```
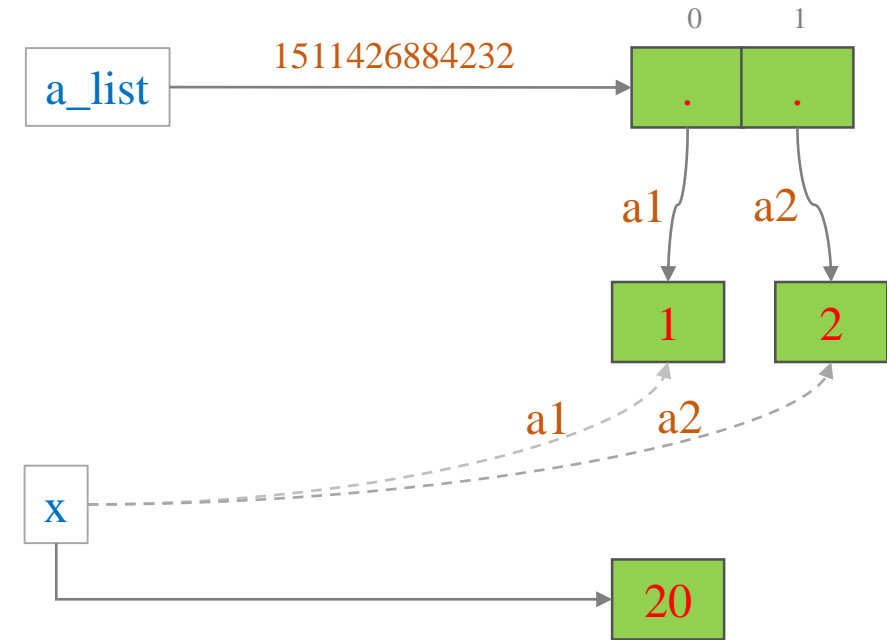

a1 = 140707105248064
a2 = 140707105248096
a3 = 140707105248128

Variables and Addresses

```
1   # aivietnam
2
3   a_list = [1, 2]
4   print(f'a_list is {a_list}')
5
6   for x in a_list:
7       x = 20
8
9   print(f'a_list is {a_list}')
```
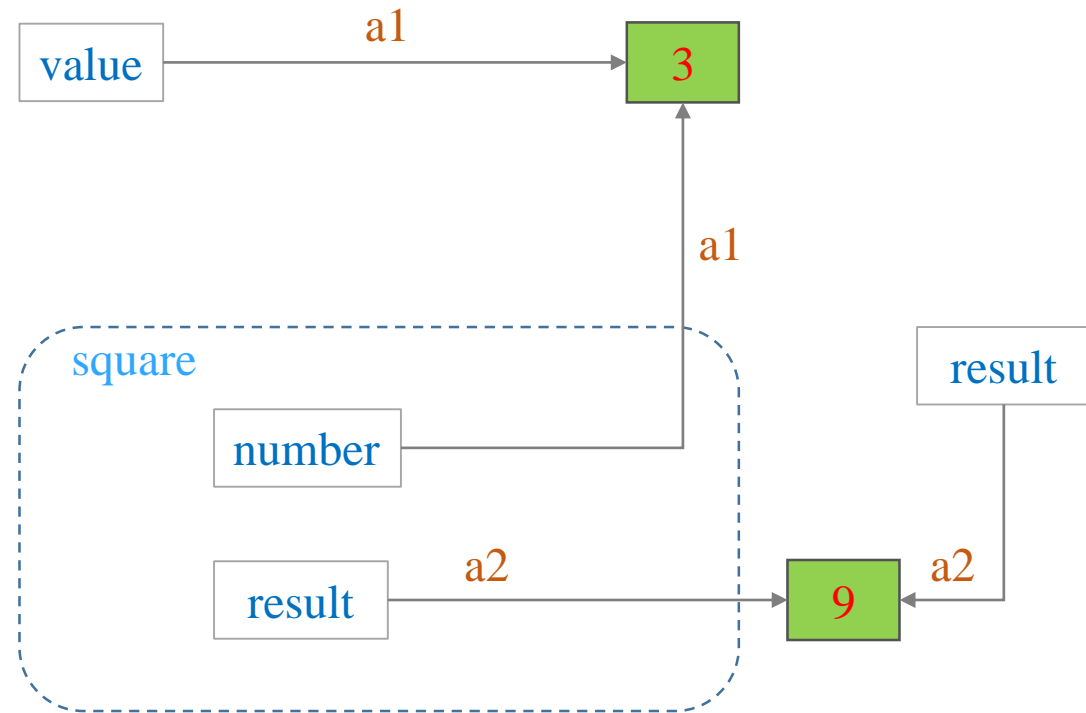
```
a_list is [1, 2]
a_list is [1, 2]
```


a1 = 140707105248064
a2 = 140707105248096

Variables and Addresses

```
1    # aivietnam
2
3    def square(number):
4        result = number*number
5
6        print(f'number address is {id(number)}')
7        print(f'result address is {id(result)}')
8
9        return result
10
11   # test the function square
12   value = 3
13   print(f'value address is {id(value)}')
14
15   result = square(value)
16   print(f'result address is {id(result)}')
```

```
value address is 140707105248128
number address is 140707105248128
result address is 140707105248320
result address is 140707105248320
```


a1 = 140707105248128
a2 = 140707105248320

Variables and Addresses

```python
1   # aivietnam
2
3   def square(number):
4       result = number*number
5
6       print(f'number address is {id(number)}')
7       print(f'result address is {id(result)}')
8
9       return result
10
11  # test the function square
12  value = 3
13  print(f'value address is {id(value)}')
14
15  result = square(value)
16  print(f'result address is {id(result)}')
```

```
value address is 140707105248128
number address is 140707105248128
result address is 140707105248320
result address is 140707105248320
```
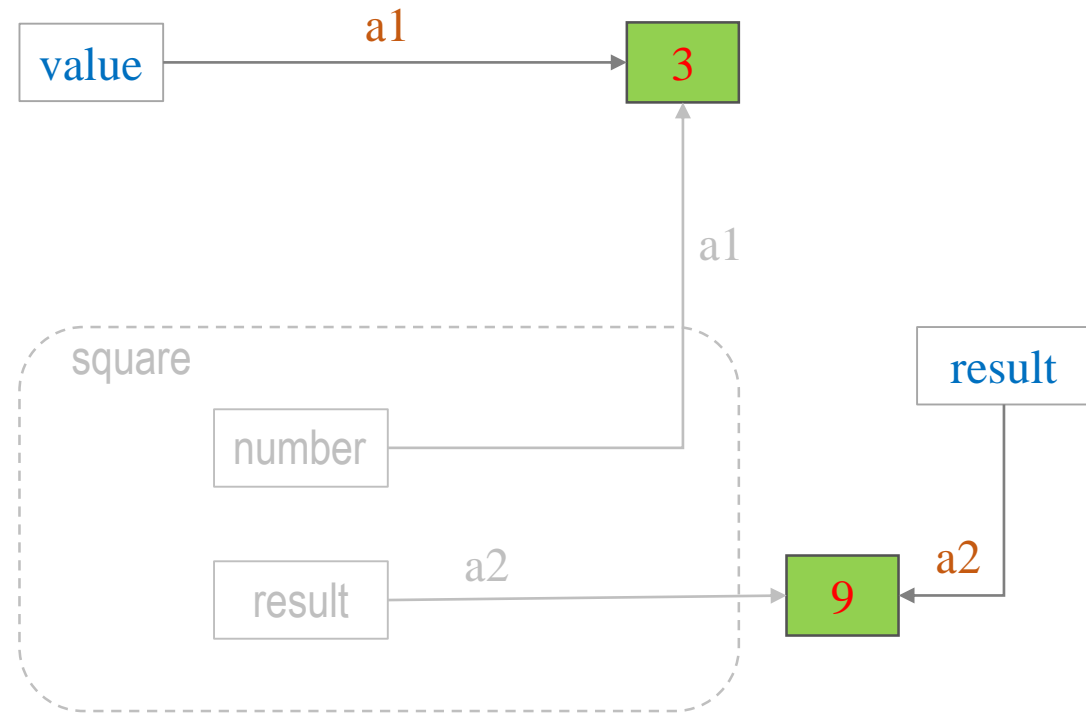
value ——a1——→ 3

a1

square

number

result ——a2——→ 9

result

a2

a1 = 140707105248128
a2 = 140707105248320

```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4   print(f'a_list address is {id(a_list)}')
5   print(f'a_list is {a_list} \n')
6
7   def add(input_list, number):
8       input_list.append(number)
9       print(f'input_list address is {id(input_list)}')
10
11  # test the add function
12  add(a_list, 4)
13  print(f'a_list is {a_list}')
```

```
a_list address is 1511427661128
a_list is [1, 2, 3]

input_list address is 1511427661128
a_list is [1, 2, 3, 4]
```

```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4
5   def add(input_list, number):
6       input_list.append(number)
7       print(f'number address is {id(number)}')
8
9   # test the add function
10  add(a_list, 4)
11  print(f'a_list is {a_list}')
12
13  print(f'a_list[3] address is {id(a_list[3])}')
```

```
number address is 140707105248160
a_list is [1, 2, 3, 4]
a_list[3] address is 140707105248160
```


a1 = 1511428290376
a2 = 140707105248160

46

```
1  # aivietnam
2
3  a_list = [1, 2, 3]
4  print(f'a_list address is {id(a_list)}')
5  print(f'a_list is {a_list} \n')
6
7  def add(input_list, number):
8      input_list.append(number)
9      print(f'input_list address is {id(input_list)}')
10
11  # test the add function
12  add(a_list, 4)
13  print(f'a_list is {a_list}')
```

```
a_list address is 1511427661128
a_list is [1, 2, 3]

input_list address is 1511427661128
a_list is [1, 2, 3, 4]
```
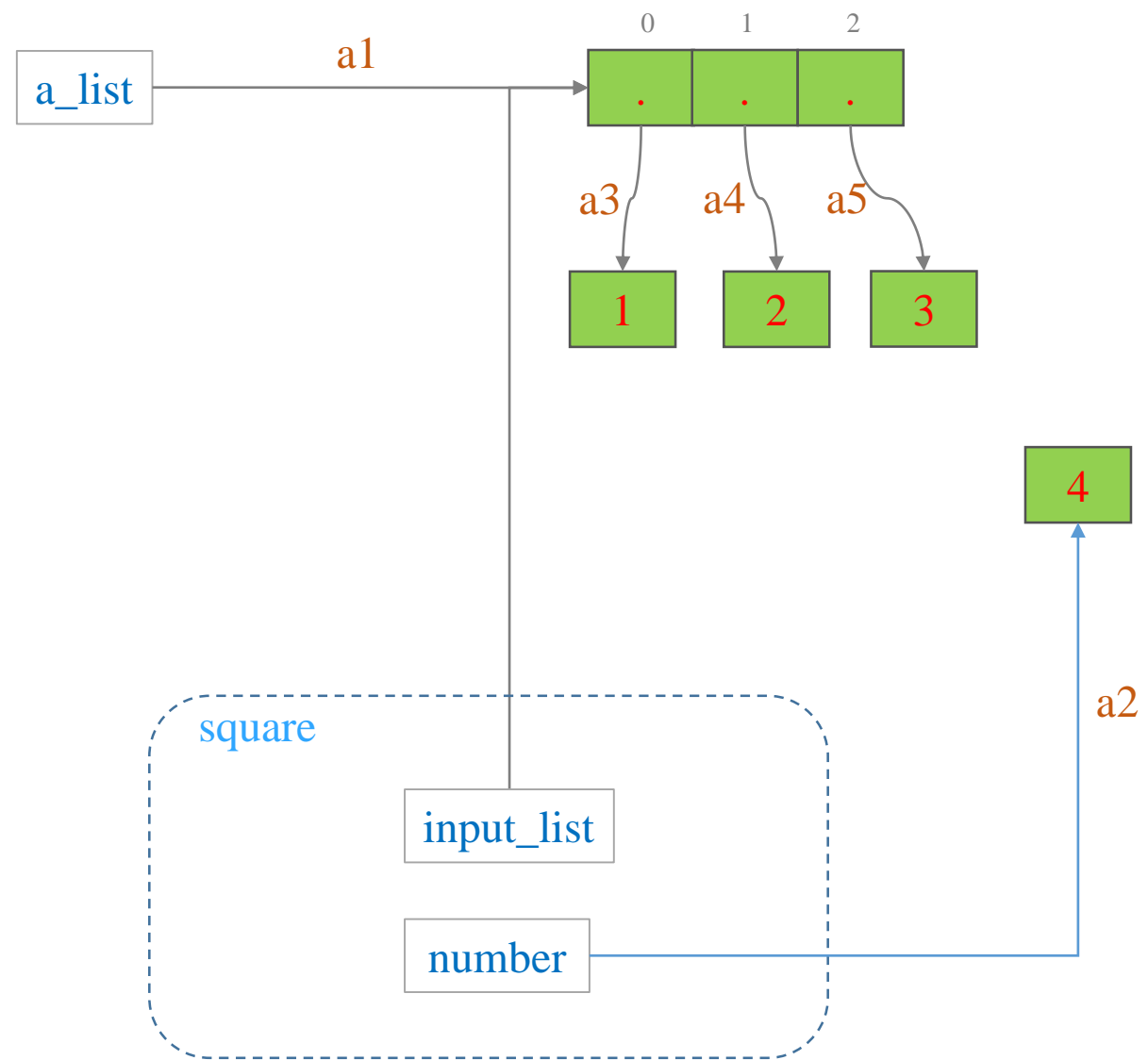
```
1  # aivietnam
2
3  a_list = [1, 2, 3]
4
5  def add(input_list, number):
6      input_list.append(number)
7      print(f'number address is {id(number)}')
8
9  # test the add function
10  add(a_list, 4)
11  print(f'a_list is {a_list}')
12
13  print(f'a_list[3] address is {id(a_list[3])}')
```

```
number address is 140707105248160
a_list is [1, 2, 3, 4]
a_list[3] address is 140707105248160
```
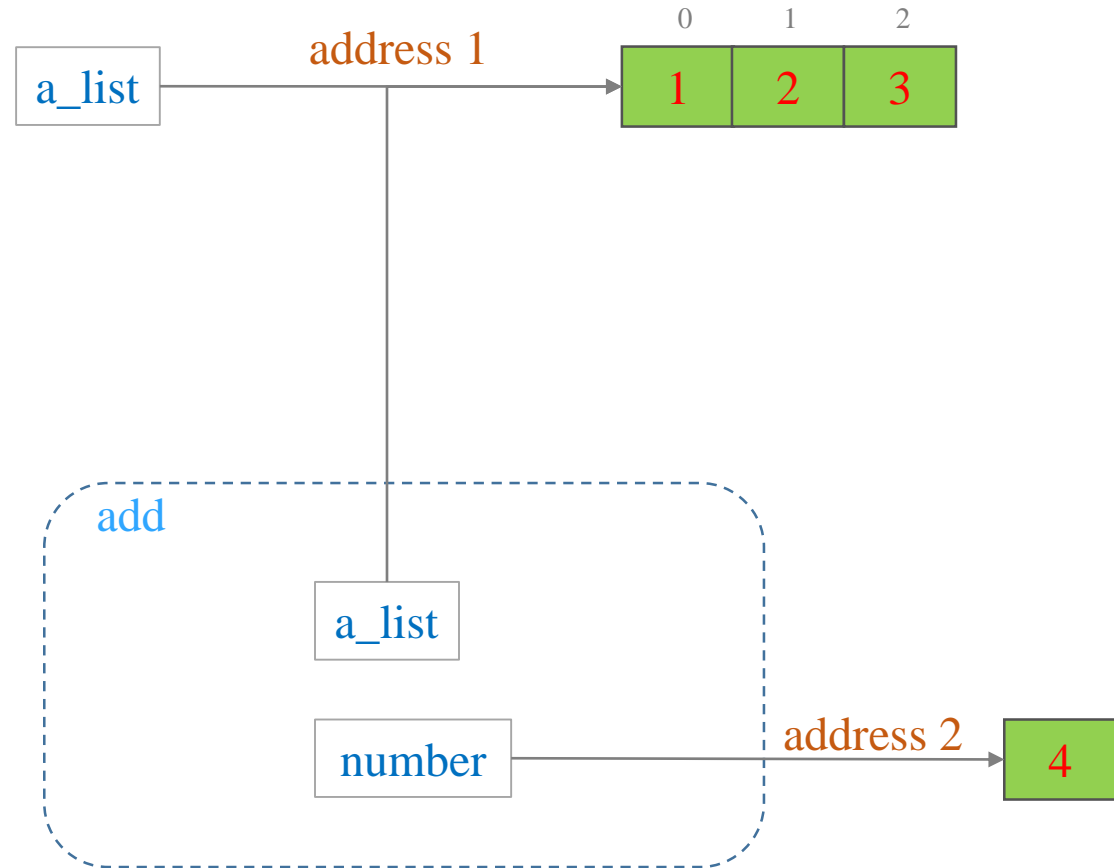

a1 = 1511428290376
a2 = 140707105248160

47

Variables and Addresses

```
1  # aivietnam
2
3  a_list = [1, 2, 3]
4  print(f'a_list is {a_list}')
5
6  def add(a_list, number):
7      a_list = a_list + [number]
8
9      return a_list
10
11 # test the add function
12 a_list = add(a_list, 4)
13 print(f'a_list is {a_list}')
```

```
a_list is [1, 2, 3]
a_list is [1, 2, 3, 4]
```

Variables and Addresses

```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4   print(f'a_list is {a_list}')
5
6   def add(a_list, number):
7       a_list = a_list + [number]
8
9       return a_list
10
11  # test the add function
12  a_list = add(a_list, 4)
13  print(f'a_list is {a_list}')
```

```
a_list is [1, 2, 3]
a_list is [1, 2, 3, 4]
```

	0	1	2

a_list — address 1 → | 1 | 2 | 3 |

add

	0	1	2	3

a_list — address 3 → | 1 | 2 | 3 | 4 |

number — address 2 → | 4 |

Variables and Addresses


```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4   print(f'a_list is {a_list}')
5
6   def add(a_list, number):
7       a_list = a_list + [number]
8
9       return a_list
10
11  # test the add function
12  a_list = add(a_list, 4)
13  print(f'a_list is {a_list}')
```

```
a_list is [1, 2, 3]
a_list is [1, 2, 3, 4]
```

Variables and Addresses


```
1   # aivietnam
2
3   a_list = [1, 2, 3]
4   print(f'a_list is {a_list}')
5
6   def add(a_list, number):
7       a_list = a_list + [number]
8
9       return a_list
10
11  # test the add function
12  a_list = add(a_list, 4)
13  print(f'a_list is {a_list}')
```
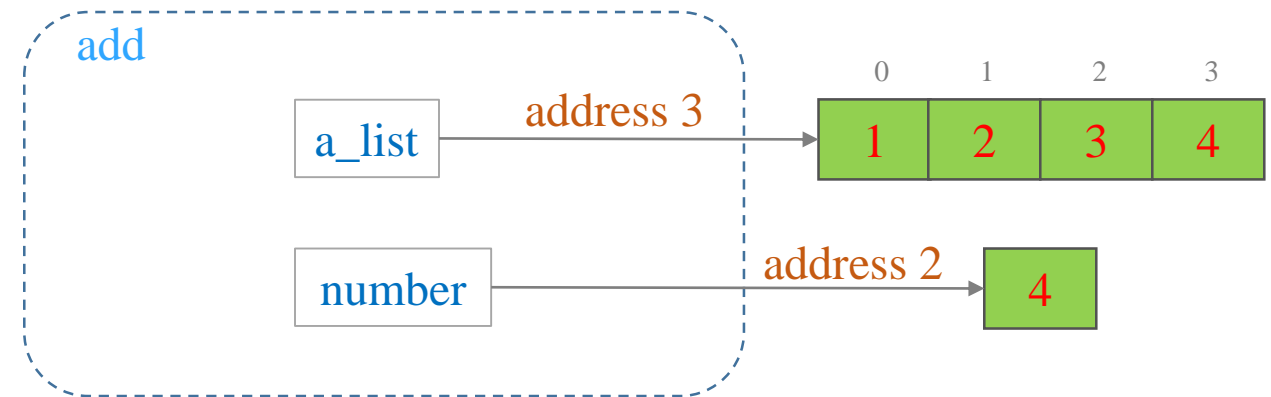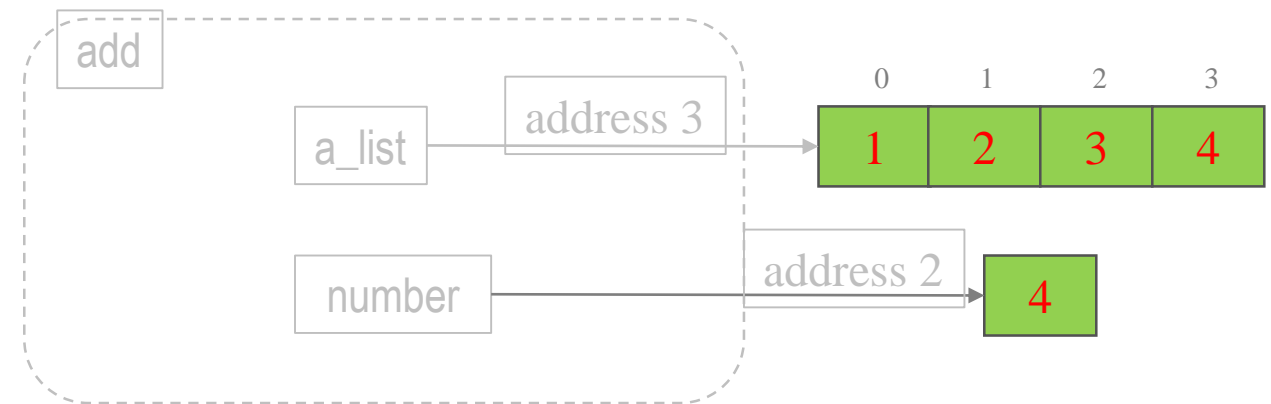
```
a_list is [1, 2, 3]
a_list is [1, 2, 3, 4]
```

Mutable and Immutable

```python
# immutable

def square(data):
    result = []
    for value in data:
        result.append(value*value)

    return result

# test
data = [6, 5, 7, 1, 9, 2]
print(data)

data_s = square(data)
print(data_s)
```



```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```

Mutable and Immutable

```python
1   # mutable
2   def square(data):
3       length = len(data)
4
5       for i in range(length):
6           value = data[i]
7           data[i] = value*value
8
9   # test
10  data = [6, 5, 7, 1, 9, 2]
11  print(data)
12
13  square(data)
14  print(data)
```

```
[6, 5, 7, 1, 9, 2]
[36, 25, 49, 1, 81, 4]
```


53

List Copy

```
1  # aivietnam - copy in python
2
3  list1 = [1, 2, 3, 4, 5]
4  list2 = list1
5
6  print('List 1: ', list1)
7  print('List 2: ', list2)
```

```
List 1:  [1, 2, 3, 4, 5]
List 2:  [1, 2, 3, 4, 5]
```
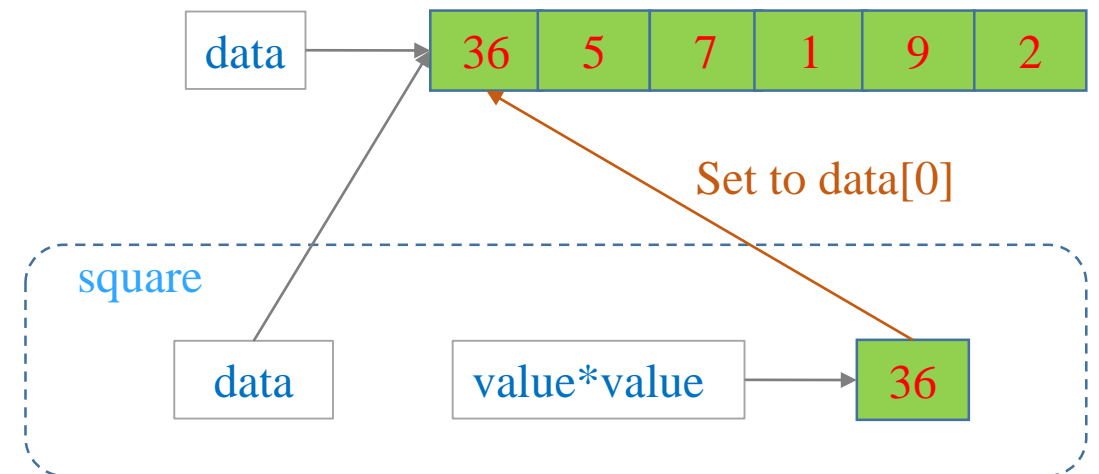
Memory address
1700123138824

| list1 | → |

1700123138824

| list2 |

list2 = list1

	0	1	2	3	4
	1	2	3	4	5

```
1  list1[3] = 9
2  print('List 1: ', list1)
3  print('List 2: ', list2)
```

```
List 1:  [1, 2, 3, 9, 5]
List 2:  [1, 2, 3, 9, 5]
```

| list1[3] = 9 |

	0	1	2	3	4
	1	2	3	4	5

Replace 4 by 9

```
1  print('List1 address: ', id(list1))
2  print('List2 address: ', id(list2))
```

```
List1 address:  1700123138824
List2 address:  1700123138824
```

1700123138824

| list1 | → |

	0	1	2	3	4
	1	2	3	9	5

1700123138824

| list2 |

List Copy

```python
1   # aivietnam - copy in python
2
3   import copy
4
5   list1 = [1, 2, 3, 4, 5]
6   list2 = copy.copy(list1)
7
8   print('List 1: ', list1)
9   print('List1 address: ', id(list1))
10
11  print()
12  print('List 2: ', list2)
13  print('List2 address: ', id(list2))
```

```
List 1:  [1, 2, 3, 4, 5]
List1 address:  1700123139912

List 2:  [1, 2, 3, 4, 5]
List2 address:  1700123139976
```

```python
1   list1[3] = 9
2   print('List 1: ', list1)
3   print('List 2: ', list2)
```

```
List 1:  [1, 2, 3, 9, 5]
List 2:  [1, 2, 3, 4, 5]
```

	0	1	2	3	4
list1 1700123139912	1	2	3	4	5

	0	1	2	3	4
list2 1700123139976	1	2	3	4	5

list2 = copy.copy(list1)

	0	1	2	**3**	4
list1[3] = 9	1	2	3	4	5

Replace 4 by 9

	0	1	2	3	4
list1 1700123139912	1	2	3	9	5

	0	1	2	3	4
list2 1700123139976	1	2	3	4	5

55

List Copy


```python
# aivietnam - copy in python

import copy

list1 = [[1, 2], 3]
list2 = copy.copy(list1)

print('List 1: ', list1)
print('List1 address: ', id(list1))


print()
print('List 2: ', list2)
print('List2 address: ', id(list2))
```

```
List 1:  [[1, 2], 3]
List1 address:  1700123189832

List 2:  [[1, 2], 3]
List2 address:  1700123189960
```

```python
print('List1[0] address: ', id(list1[0]))
print('List2[0] address: ', id(list2[0]))
```

```
List1[0] address:  1700123189704
List2[0] address:  1700123189704
```

```python
list1[0][0] = 9
print('List 1: ', list1)
print('List 2: ', list2)
```

```
List 1:  [[9, 2], 3]
List 2:  [[9, 2], 3]
```
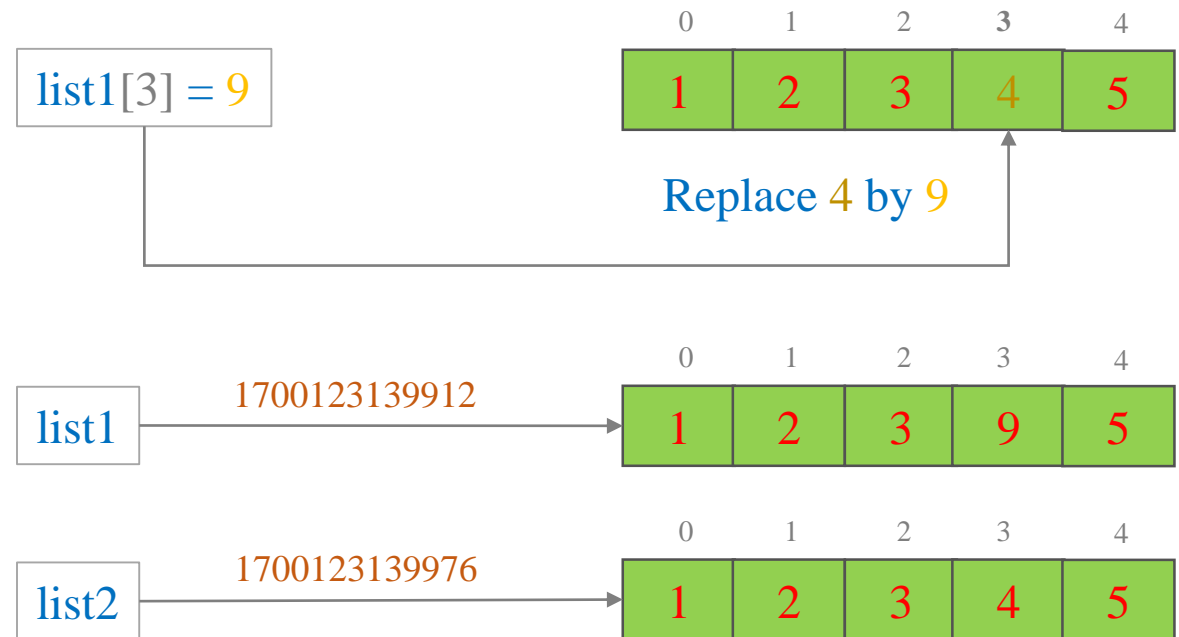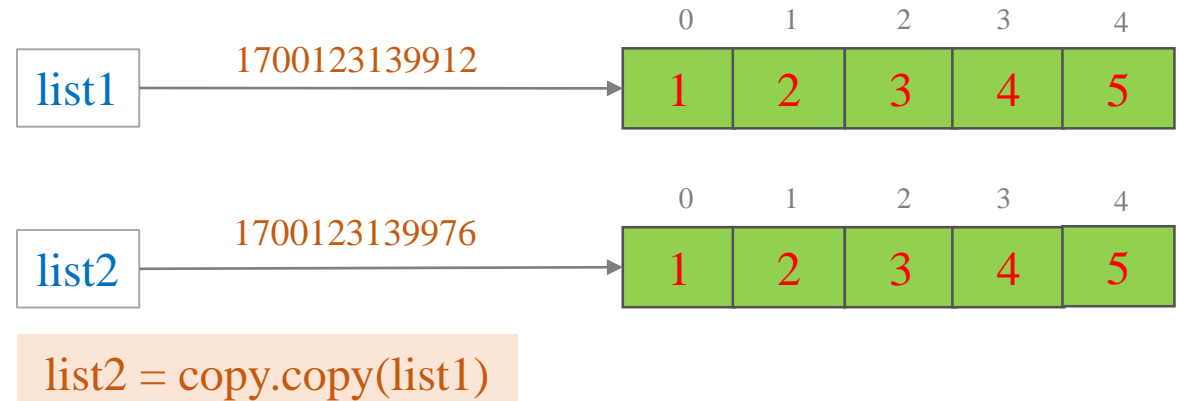
56
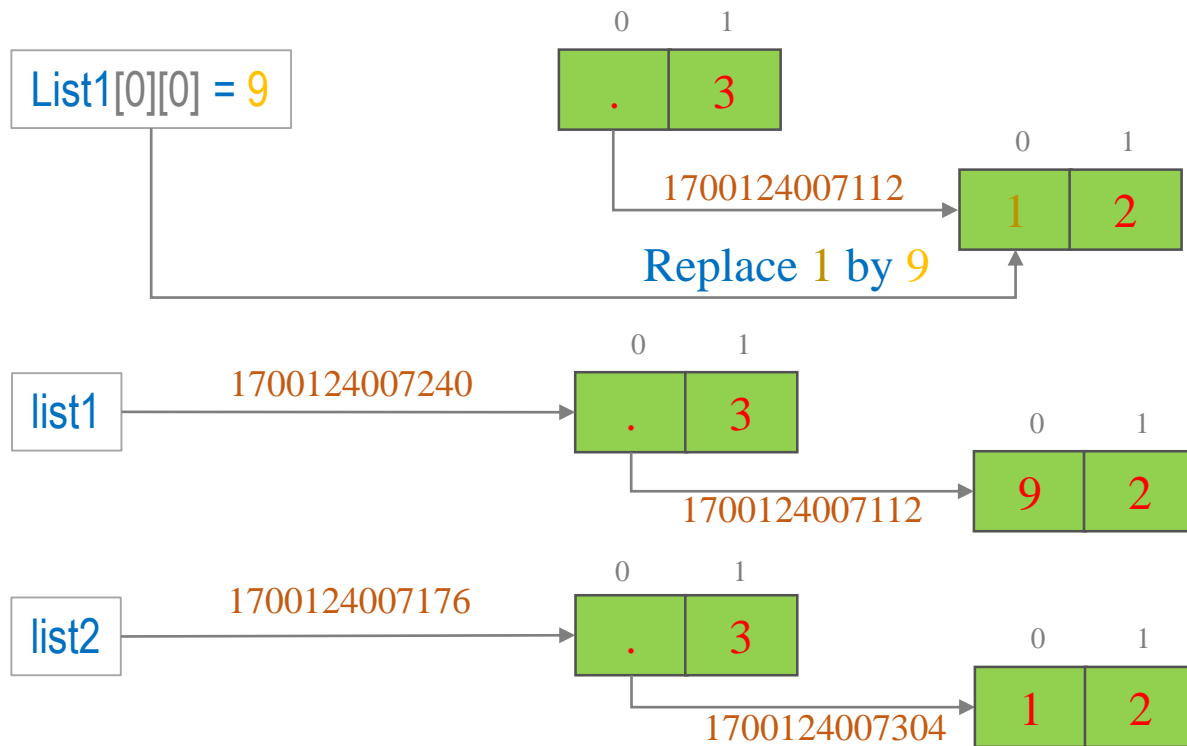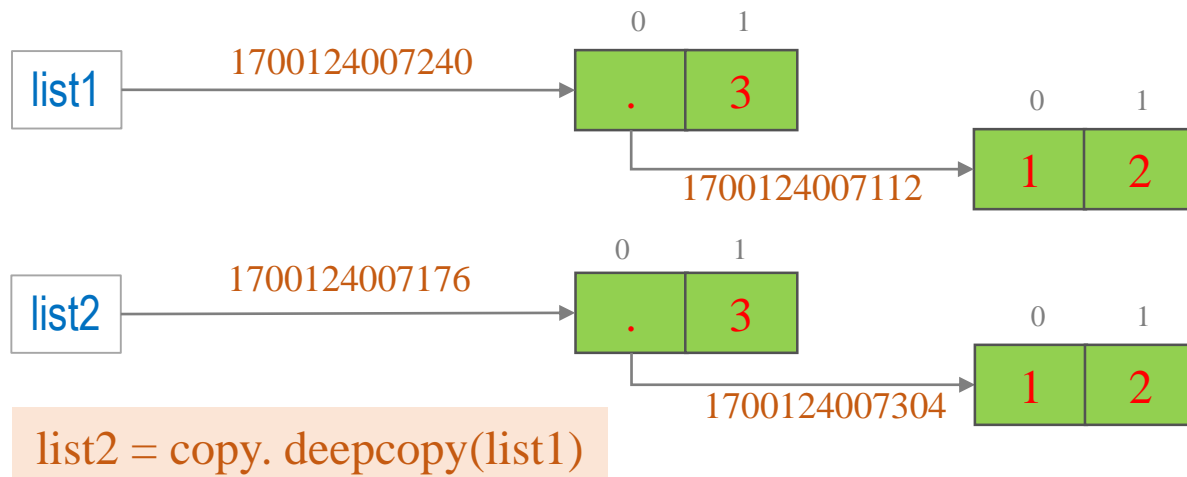
```
1   # aivietnam - copy in python
2
3   import copy
4
5   list1 = [[1, 2], 3]
6   list2 = copy.deepcopy(list1)
7
8   print('List 1: ', list1)
9   print('List1 address: ', id(list1))
10
11  print()
12  print('List 2: ', list2)
13  print('List2 address: ', id(list2))
```
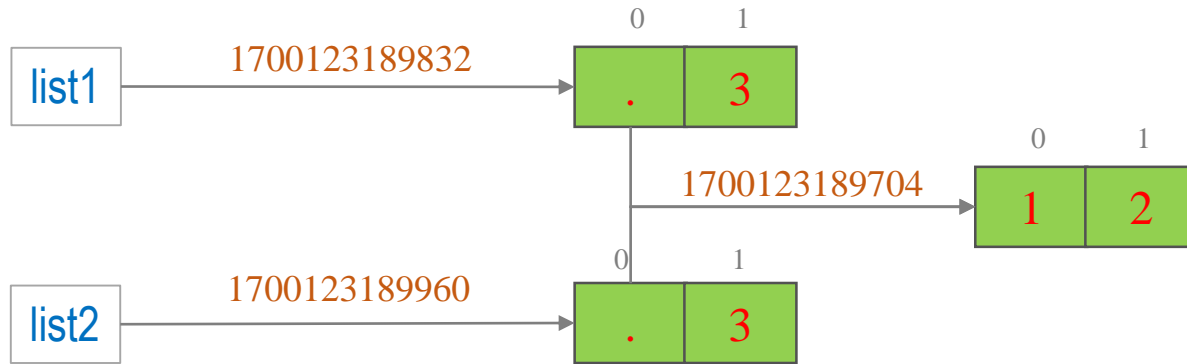
```
List 1:  [[1, 2], 3]
List1 address:  1700124007240

List 2:  [[1, 2], 3]
List2 address:  1700124007176
```

```
1   print('List1[0] address: ', id(list1[0]))
2   print('List2[0] address: ', id(list2[0]))
```

```
List1[0] address:  1700124007112
List2[0] address:  1700124007304
```

```
1   list1[0][0] = 9
2   print('List 1: ', list1)
3   print('List 2: ', list2)
```

```
List 1:  [[9, 2], 3]
List 2:  [[1, 2], 3]
```

list2 = copy. deepcopy(list1)

List1[0][0] = 9

Replace 1 by 9

Outline

- ➢ **Introduction**
- ➢ **List**
- ➢ **Some algorithms on List**
- ➢ **Addresses**
- ➢ **Common Errors**

Common Errors

❖ **Error 1**

```
4.    # khai báo biến a = 5
5.    a = 5
6.
7.    # thực hiên a + b, sau đó lưu vào biến c
8.    c = a + b
9.
10.   # in giá trị c
11.   print(c)
```

```
---------------------------------------------------------------
NameError                          Traceback (most recent call last)
<ipython-input-1-eae96ee94f9f> in <module>
      6
      7 # thực hiên a + b, sau đó lưu vào biến c
----> 8 c = a + b
      9
     10 # in giá trị c

NameError: name 'b' is not defined
```

Common Errors

❖ **Error 2**

```
4.    # khai báo biến a = 5
5.    a = 5
6.
7.    # in giá trị a
8.    Print(a)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-f09db6b2bf7e> in <module>
      6
      7 # in giá trị a
----> 8 Print(a)

NameError: name 'Print' is not defined
```

Common Errors

❖ **Error 3**

```
4.    # khai báo biến chuỗi s
5.    s = 'Hello AIVIETNAM"
6.
7.    # in giá trị s
8.    print(s)
```

```
File "<ipython-input-3-96feed73c6b1>", line 5
    s = 'Hello AIVIETNAM"
                        ^
SyntaxError: EOL while scanning string literal
```

Common Errors

❖ **Error 4**

```
4.    # khai báo biến a và b
5.    a = 5
6.    b = 0
7.
8.    # tính giá trị c bằng a chia cho b
9.    c = a / b
10.
11.   # in giá trị c
12.   print(c)
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-4-298e1112d534> in <module>
      7
      8 # tính giá trị c bằng a chia cho b
----> 9 c = a / b
     10
     11 # in giá trị c

ZeroDivisionError: division by zero
```

Common Errors

❖ **Error 5**

```
4.      # khai báo biến chuỗi s
5.      s = 'AI'
6.
7.      # khai báo biến n có kiểu integer
8.      n = 5
9.
10.     # tính giá trị c
11.     c = s + n
12.
13.     # in giá trị c
14.     print(c)
```

```
-------------------------------------------------------------------
TypeError                              Traceback (most recent call last)
<ipython-input-5-f1e2455fae51> in <module>
      9
     10 # tính giá trị c
---> 11 c = s + n
     12
     13 # in giá trị c

TypeError: must be str, not int
```

Common Errors

❖ **Error 6**

```
4.     # khai báo biến chuỗi s
5.     s = 'AI'
6.
7.     # khai báo biến n có kiểu integer
8.     n = 5
9.
10.    # tính giá trị c
11.    c = n + s
12.
13.    # in giá trị c
14.    print(c)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-8-cecd5289546d> in <module>
      9
     10 # tính giá trị c
---> 11 c = n + s
     12
     13 # in giá trị c

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Common Errors

❖ **Error 7**

```
4.      # khai báo biến a và b
5.      a = 5
6.        b = 6
7.
8.      # thực hiên a + b, sau đó lưu vào biến c
9.      c = a + b
10.
11.     # in giá trị c
12.     print(c)
```

```
  File "<ipython-input-7-31f64166c395>", line 6
    b = 6
    ^
IndentationError: unexpected indent
```

Common Errors

❖ **Error 8**

```
4.    import math
5.
6.    number = 20.2
7.    print(math.floor(number)
8.    print(math.pi)
```

```
  File "<ipython-input-15-920005110c33>", line 8
    print(math.pi)
        ^
SyntaxError: invalid syntax
```

Common Errors

❖ **Error 9**

```
3.
4.    print "aivietnam.ai"
```

```
File "<ipython-input-3-a46b1c9e05ed>", line 4
    print "aivietnam.ai"
                        ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("aivietnam.ai")?
```

Common Errors

❖ **Error 10**

```
4.  import mymodule
5.
6.  print("aivietnam.ai")
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-5-1b242c59080b> in <module>
      2 # Lỗi khai báo module không tồn tại
      3
----> 4 import mymodule
      5
      6 print("aivietnam.ai")

ModuleNotFoundError: No module named 'mymodule'
```

Common Errors

❖ **Error 11**

```
3.
4.    l = [1, 2, 3, 4, 5]
5.    print(l[0])
6.    print(l[5])
```

```
1


------------------------------------------------------------------
IndexError                          Traceback (most recent call last)
<ipython-input-6-b8c53176edc0> in <module>
      4 l = [1, 2, 3, 4, 5]
      5 print(l[0])
----> 6 print(l[5])

IndexError: list index out of range
```

Common Errors

❖ **Error 12**

```
4.   name = "aivietname.ai"
5.   print(name[0])
6.   print(name[50])
```

```
a

-------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-7-357ff533411a> in <module>
      4 name = "aivietname.ai"
      5 print(name[0])
----> 6 print(name[50])

IndexError: string index out of range
```

Common Errors

❖ **Error 13**

```
4.    number = 15
5.    if number < 10
6.        print("A small number")
7.    else:
8.        print("A large number")
```

```
File "<ipython-input-15-fedf173614ac>", line 5
    if number < 10
                  ^
SyntaxError: invalid syntax
```

Common Errors

❖ **Error 14**

```
4.    number = 15
5.    if number < 10:
6.        print("A small number")
7.    else
8.        print("A large number")
```

```
  File "<ipython-input-16-699752908646>", line 7
    else
        ^
SyntaxError: invalid syntax
```

Common Errors

❖ **Error 15**

```
1.    import math
2.
3.    number = -4
4.    print(math.sqrt(number))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-8-f25b4b744f6e> in <module>
      2
      3 number = -4
----> 4 print(math.sqrt(number))

ValueError: math domain error
```

Common Errors

❖ **Error 16**

```
4.   # import pytorch
5.   import torch
6.
7.   print(torch.cuda.is_available())
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-14-680f8ea2b256> in <module>
      1 # import thu viện pytorch
----> 2 import torch
      3
      4 print(torch.cuda.is_available())

ModuleNotFoundError: No module named 'torch'
```

Common Errors

❖ **Error 17**

```
4   index = 5
5   def my_function():
6       index += 1
7       print(index)
8
9   my_function()
```

```
---------------------------------------------------------------------
UnboundLocalError                         Traceback (most recent call last)
<ipython-input-4-63ee67900288> in <module>
      7       print(index)
      8
----> 9 my_function()

<ipython-input-4-63ee67900288> in my_function()
      4 index = 5
      5 def my_function():
----> 6       index += 1
      7       print(index)
      8

UnboundLocalError: local variable 'index' referenced before assignment
```

Common Errors

❖ **Error 18**

```
4  a_number = 5
5  a_string = 'value '
6  result = a_string + a_number
7
8  print(result)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-9772d690ff0d> in <module>
      4 a_number = 5
      5 a_string = 'value '
----> 6 result = a_string + a_number
      7
      8 print(result)

TypeError: can only concatenate str (not "int") to str
```

Common Errors

❖ **Error 19**

```
3   def a_function(x):
4       a_variable = 4
5       result = x*a_variable
6
7       return result
8
9   print(a_variable)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-43-2cce0357dce7> in <module>
      6
      7
----> 8 print(a_variable)

NameError: name 'a_variable' is not defined
```

Common Errors

❖ **Error 20**

```
4  str1 = '5'
5  str2 = 'hello'
6
7  value1 = int(str1)
8  value2 = int(str2)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-9-1f0b23b26eb1> in <module>
      6
      7 value1 = int(str1)
----> 8 value2 = int(str2)

ValueError: invalid literal for int() with base 10: 'hello'
```

Common Errors

❖ **Error 21**

```
3
4  def a_function(n):
5      return a_function(n)
6
7  a_function(5)
```

```
--------------------------------------------------------------------------------
RecursionError                                  Traceback (most recent call last)
<ipython-input-10-bda7ef50bf68> in <module>
      5        return a_function(n)
      6
----> 7 a_function(5)


<ipython-input-10-bda7ef50bf68> in a_function(n)
      3
      4 def a_function(n):
----> 5        return a_function(n)
      6
      7 a_function(5)


... last 1 frames repeated, from the frame below ...


<ipython-input-10-bda7ef50bf68> in a_function(n)
      3
      4 def a_function(n):
----> 5        return a_function(n)
      6
      7 a_function(5)


RecursionError: maximum recursion depth exceeded
```

Common Errors

❖ **Error 22**

```
1   # aivietnam - example
2
3   # create a list of numbers
4   numbers = [1, 2, 3, 4, 5]
5
6   for num in numbers:
7       # print the num
8       print(num)
9
10      # check if the num is equal to 3
11      if (num == 3):
12          # add 3 to the list
13          numbers.append(3)
```

numbers $= [1, 2, 3, 4, 5]$

Output

1

numbers $= [1, 2, 3, 4, 5]$

Output

$1, 2$

numbers $= [1, 2, 3, 4, 5]$

Output

$1, 2, 3$

numbers $= [1, 2, 3, 4, 5, 3]$

Common Errors

❖ **Error 22**

```python
1   # aivietnam - example
2
3   # create a list of numbers
4   numbers = [1, 2, 3, 4, 5]
5
6   for num in numbers:
7       # print the num
8       print(num)
9
10      # check if the num is equal to 3
11      if (num == 3):
12          # add 3 to the list
13          numbers.append(3)
```

Output

$1, 2, 3, 4, 5, 3$

numbers $= [1, 2, 3, 4, 5, 3]$

numbers $= [1, 2, 3, 4, 5, 3, 3]$

Common Errors

❖ **Error 22**

```
1   # aivietnam - example
2
3   # create a list of numbers
4   numbers = [1, 2, 3, 4, 5]
5
6   for num in numbers:
7       # print the num
8       print(num)
9
10      # check if the num is equal to 3
11      if (num == 3):
12          # add 3 to the list
13          numbers.append(3)
```

Output

numbers = $[1, 2, 3, 4, 5, 3]$ $1, 2, 3, 4, 5, 3$

Output

numbers = $[1, 2, 3, 4, 5, 3, 3]$ $1, 2, 3, 4, 5, 3, 3$

numbers = $[1, 2, 3, 4, 5, 3, 3, 3]$ ○ ○ ○

Common Errors

❖ **Errors 25**

```
1  # common error
2
3  name = 'John'
4  age  = 26
5
6  print('Hello {name}. Are you {age}?')
```

Hello {name}. Are you {age}?

```
1  # common error
2
3  fruits = {'banana': 2}
4  fruits['apple'] += 10
```

```
---------------------------------------------------
KeyError                                    Traceback
<ipython-input-1-139246a7a818> in <module>
      2
      3 fruits = {'banana': 2}
----> 4 fruits['apple'] += 10

KeyError: 'apple'
```

Common Errors

❖ **Errors 26**

```
1   # common error
2
3   fruits = ['apple', 'banana', 'peach']
4
5   for fruit in fruits:
6       if (fruit == 'peach'):
7           fruits.append('peach')
8
9       print('Do something with this fruit -', fruit)
```

```
1   # common error
2
3   fruits = ['apple', 'banana', 'peach']
4
5   for fruit in fruits:
6       if (fruit == 'banana'):
7           fruits.remove('banana')
8
9       print('Do something with this fruit -', fruit)
```

???

Common Errors

❖ **Errors 27**

```python
1  # common error
2
3  a = 0.1 + 0.1 + 0.1
4  if (a == 0.3):
5      print('Yes')
6  else:
7      print('No')
```

Quizzes

```python
# quiz 1
index = 5
def my_function():
    print(index)



my_function()
```
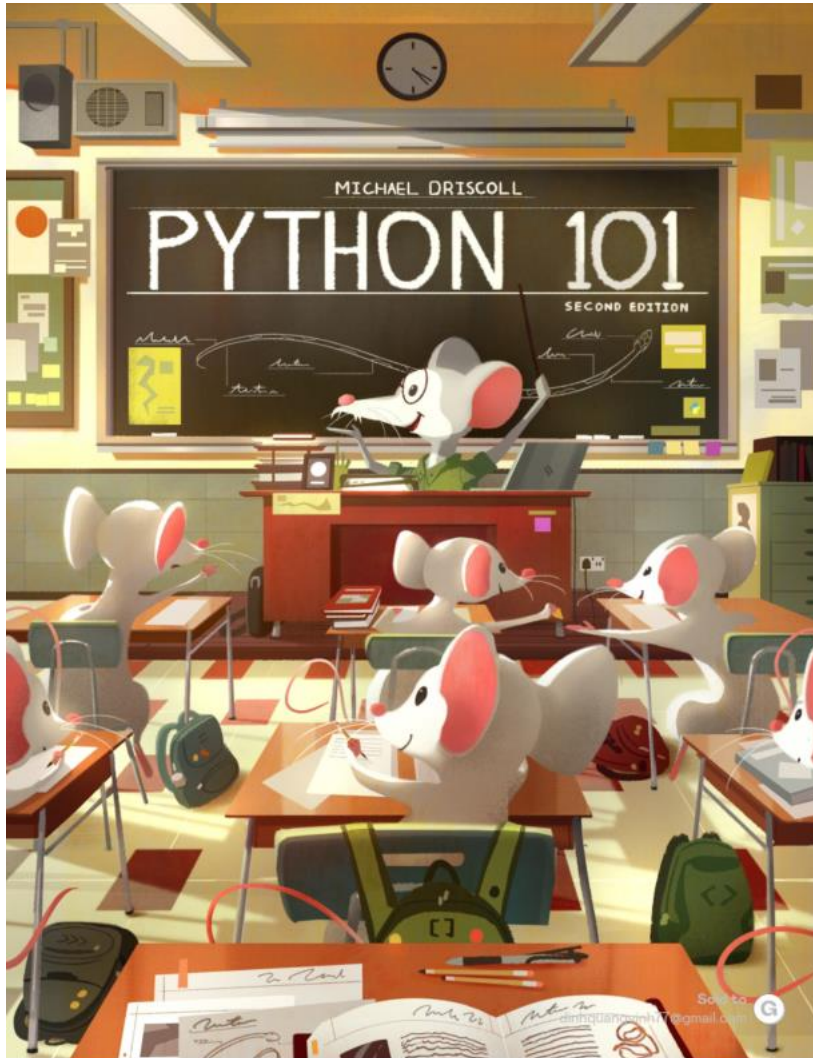
```python
# quiz 3
index = 5
def my_function():
    index = index + 1
    print(index)


my_function()
```

```python
# quiz 2
index = 5
def my_function():
    data = index + 1
    print(index)


my_function()
```

```python
# quiz 4
index = 5
def my_function(index):
    index = index + 1
    print(index)


my_function(7)
```

Further Reading

❖ **List (Chapters 6 and 13)**

❖ **Binary Searching and Sorting**

Binary Search Diagram

Worst-case binary search (8-element array)
Key = 9

| 1 | 2 | 5 | 6 | 7 | 9 | 11 | 13 | Step 1:
9 > 7 (choose right)

| 7 | 9 | 11 | 13 | Step 2:
9 < 11 (choose left)

| 7 | 9 | Step 3:
9 = 9 (key located)

Key located in 3 operations
log(8) = 3

ComputerHope.com