


TRƯỜNG: ĐH Công thương TPHCM KHOA: CÔNG NGHỆ THÔNG TIN BỘ MÔN: KHDL&TTNT MH: LẬP TRÌNH PYTHON	BUỔI 2. HÀM TRONG LẬP TRÌNH PYTHON	
--	---	---

A. MỤC TIÊU

- Xây dựng được các hàm chức năng và các lớp hàm.

B. DỤNG CỤ - THIẾT BỊ THÍ NGHIỆM CHO MỘT SV

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

I. Tóm tắt lý thuyết

1. Hàm:

Tạo hàm, gọi hàm

```
def print_hello():
    print('Hello!')

print_hello()
print('1234567')
print_hello()
```

→ Hello!
1234567
Hello!

Đối số của hàm

Ví dụ 1:

```
def my_function(fname, lname):
    print(fname + " " + lname)

my_function("Emil", "Refsnes")

def my_function(food):
    for x in food:
        print(x)

fruits = ["apple", "banana", "cherry"]
my_function(fruits)
```

→ apple
banana
cherry

Ví dụ 2:

```
def print_hello(n):
    print('Hello ' * n)
    print()

print_hello(3)
print_hello(5)
times = 2
print_hello(times)
```

Hello Hello Hello
Hello Hello Hello Hello Hello
Hello Hello

Ví dụ 3: Viết hàm vẽ hình chữ nhật theo mẫu

```
*****
*                                     *
*                                     *
*****
```

```
def draw_square():
    print('*' * 15)
    print('*', ' '*11, '*')
    print('*', ' '*11, '*')
    print('*' * 15)
```

Hàm có kết quả trả về:

```
def convert(t):
    return t*9/5+32

print(convert(20))
```

68

Ví dụ 4: Viết chương trình giải hệ phương trình:

$$\begin{cases} ax + by = e \\ cx + dy = f \end{cases}$$

Phân tích bài toán:

Ta biết nghiệm của phương trình là: $x = (de - bf)/(ad - bc)$ và $y = (af - ce)/(ad - bc)$

```
def solve(a,b,c,d,e,f):
    x = (d*e-b*f)/(a*d-b*c)
    y = (a*f-c*e)/(a*d-b*c)
    return [x,y]

xsol, ysol = solve(2,3,4,1,2,5)
print('The solution is x = ', xsol, 'and y = ', ysol)
```

The solution is x = 1.3 and y = -0.2

Hàm đệ quy

```
def tri_recursion(k):
    if(k > 0):
        result = k + tri_recursion(k - 1)
        print(result)
    else:
        result = 0
    return result

print("\n\nRecursion Example Results")
tri_recursion(6)
```

2. List:

<tên biến> = <[M1, M2, ..., Mn]> , Mi có thể cùng kiểu dữ liệu hoặc khác kiểu.

Ví dụ :

```
list1 = ["apple", "banana", "cherry"]
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]
```

```
list1 = ["abc", 34, True, 40, "male"]
```

Chiều dài list được xác định qua hàm len()

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

Chúng ta cũng có thể sử dụng cấu trúc list() để tạo ra một list mới.

```
thislist = list(("apple", "banana", "cherry")) # note the double round-brackets
print(thislist)
```

Truy xuất đến các item của list

```
[50] thislist = list(("banana", "apple", 'mango'))
print(thislist[1])
print(thislist[-1])
print(thislist[:2])
print(thislist[-3:-1])
```

```
apple
mango
['banana', 'apple']
['banana', 'apple']
```

Lặp thông qua list

```
thislist = ["apple", "banana", "cherry"]
for x in thislist:
    print(x)
```

Lặp thông qua các số chỉ mục

```
thislist = ["apple", "banana", "cherry"]
for i in range(len(thislist)):
    print(thislist[i])
```

Các phương thức của list:

Method	Description
<code>append()</code>	Adds an element at the end of the list
<code>clear()</code>	Removes all the elements from the list
<code>copy()</code>	Returns a copy of the list
<code>count()</code>	Returns the number of elements with the specified value
<code>extend()</code>	Add the elements of a list (or any iterable), to the end of the current list
<code>index()</code>	Returns the index of the first element with the specified value
<code>insert()</code>	Adds an element at the specified position
<code>pop()</code>	Removes the element at the specified position
<code>remove()</code>	Removes the item with the specified value
<code>reverse()</code>	Reverses the order of the list
<code>sort()</code>	Sorts the list

II. Bài tập hướng dẫn mẫu

1. Tạo một hàm nhận vào hai đối số name và age và in ra giá trị của chúng.

```
# khai báo function demo
#param 1 name
#param 2 age
def demo(name, age):
    # in ra giá trị name và age
    print(name, age)

# gọi function demo
demo("ABC", 25)
```

→ ABC 25

2. Sử dụng eval (input()) để cho phép người dùng nhập danh sách.

```
L = eval(input('Enter a list: '))
print('The first element is ', L[0])
```

```
Enter a list: [5,7,9]
The first element is 5
```

3. Viết hàm trộn 2 dãy một chiều thành 1 dãy một chiều với mỗi phần tử của dãy mới là tổng của 2 phần tử tương ứng từ 2 dãy cho trước. Trong quá trình trộn 2 dãy nếu dãy nào còn phần tử thì các phần tử còn lại của dãy đó sẽ đưa vào dãy mới.

Ví dụ:

- Dãy a: 3 9 1 4
- Dãy b: 2 7 4 3 2 8
- Dãy kết quả: 5 16 5 7 2 8.

```

a = [3, 9, 1, 4]
b = [2, 7, 4, 3, 8, 2]
min = len(b) if len(a)>len(b) else len(a)
c = a if len(a)>len(b) else b
for i in range(min):
    c[i] = a[i] + b[i]
print(c)

```

[5, 16, 5, 7, 8, 2]

4. Tạo một list trái cây (bằng tiếng anh) fruits. Từ list fruits tạo một list mới chỉ chứa các loại trái cây có chứa ký tự “a”.

```

fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = []
for x in fruits:
    if "a" in x:
        newlist.append(x)
print(newlist)

```

['apple', 'banana', 'mango']

```

fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = [x for x in fruits if "a" in x]
print(newlist)

```

['apple', 'banana', 'mango']

5. Tạo một array chứa các số nguyên, sau đó tạo một filter array chứa các giá trị lớn hơn 42.

```

import numpy as np

arr = np.array([41, 42, 43, 44])

# Create an empty list
filter_arr = []

# go through each element in arr
for element in arr:
    # if the element is higher than 42, set the value to True, otherwise False:
    if element > 42:
        filter_arr.append(True)
    else:
        filter_arr.append(False)

newarr = arr[filter_arr]

print(filter_arr)
print(newarr)

```

6. Tạo một dãy filter chỉ chứa các phần tử chẵn từ dãy gốc.

```

import numpy as np

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

filter_arr = arr % 2 == 0

newarr = arr[filter_arr]

print(filter_arr)
print(newarr)

```

III. Bài tập ở lớp

- Viết hàm tạo dãy số nguyên a với n phần tử và dãy b chỉ chứa các phần tử chẵn của a.
- Viết hàm trộn 2 dãy một chiều thành 1 dãy một chiều với mỗi phần tử của dãy mới là tổng của 2 phần tương ứng từ 2 dãy cho trước. Trong quá trình trộn 2 dãy nếu dãy nào còn phần tử thì các phần tử còn lại của dãy đó sẽ đưa vào dãy mới.

Ví dụ:

Dãy a: 3 9 1 4

Dãy b: 2 7 4 3 2 8

Dãy kết quả: 5 16 5 7 2 8.

- Cho dãy 2 chiều a có m dòng, n cột chứa số nguyên, viết các hàm sau:

- Tạo dãy a chứa các số nguyên ngẫu nhiên.
- Xuất các phần tử thuộc dòng k.
- Xuất các phần tử thuộc cột k.
- Tìm dòng có tổng lớn nhất.
- Tìm cột có tích nhỏ nhất.
- Xuất ra các phần tử thuộc dòng chẵn và cột lẻ trong a.
- Tính trung bình cộng các phần tử chẵn thuộc dòng lẻ của a.
- Tính trung bình cộng các phần tử thuộc biên.
- Tính trung bình tích các phần tử không thuộc biên.

- Viết hàm trộn 2 dãy một chiều thành 1 dãy một chiều với mỗi phần tử của dãy mới là min của 2 phần tương ứng từ 2 dãy cho trước. Trong quá trình trộn 2 dãy nếu dãy nào còn phần tử thì các phần tử còn lại của dãy đó sẽ đưa vào dãy mới.

Ví dụ:

✓ Dãy a: 3 9 1 4

✓ Dãy b: 2 7 4 3 2 8

✓ Dãy kết quả: 2 7 1 3 2 8

- Tạo một danh sách mới từ 2 danh sách sử dụng điều kiện sau. Cho 2 danh sách, viết hàm tạo một danh sách mới sao cho danh sách mới chứa các số lẻ từ danh sách đầu tiên và các số chẵn từ danh sách thứ hai.

- Cho dãy 2 chiều a có m dòng, n cột chứa số nguyên, viết các hàm sau:

- Tính tổng các phần tử thuộc tam giác trên của đường chéo phụ kể cả đường chéo phụ trong ma trận vuông a cấp n.
- Chuyển các phần tử âm thành trị tuyệt đối của nó trong a.
- Thay các phần tử chẵn trong a bằng số nguyên x cho trước.
- Kiểm tra a có toàn chẵn không?
- Cho ma trận vuông a cấp n, viết hàm kiểm tra a có đối xứng không. Biết rằng ma trận đối xứng là ma trận có $a[i][j] = a[j][i]$.

- f. Kiểm tra ma trận vuông a cấp n có đường chéo chính tăng dần không?
- g. Xuất các phần tử thuộc tam giác dưới của đường chéo phụ kể cả đường chéo phụ trong ma trận vuông a cấp n .
- h. Kiểm tra ma trận vuông a cấp n có đường chéo phụ giảm dần không?