PYTHON

1.Biến

-Kiểu boolean thì phải khai báo viết hoa là True

- Casting

+ Sử dụng các hàm str() to cast st to string , int() to cast st to int…..

-String

+ s.isalnum() : is alphanumeric (contain number and word except for space ) or not : Boolean type

+ s.isalpha() : is alphabetic (contain number not for space ) or not : Boolean type

+ s.isnumeric () : is number (number) or not : Boolean type

2.Kiến thức lập trình cơ bản

- Các vòng lặp ko có () nữa

-Các vòng lặp ko có {} nữa mà thay vào là :

2.1 Vòng lặp for

Cấu trúc : for i in range(x,y,z)

+Nếu chỉ có mỗi x thì nó sẽ đi từ số chỉ 0 đến x

+Nếu có x,y thì nó đi trong khoảng x đến y

+Nếu có x,y,z thì nó đi trong khoảng x,y nhưng với khoảng cách đơn vị là z

Và có thể dùng số âm để truy lùi

VD: for i in range(9,-1 ,-1):  
 print(i)

3 Function

-Cách viết 1 function cơ bản:

+ def + tên hàm()

-Cách gọi gọi nó ra thôi bro 😊))

3.0 Hàm print

print(' ', end='')

end là để nó ko xuống dòng , default của print là auto xuống dòng

print(‘h’ \* 3) nó sẽ in h x3

3.1 Hàm global

- global statement : If you need to modify a global variable from within a function, use the global statement

Vd:

def spam():  
➊ global eggs  
➋ eggs = 'spam'  
eggs = 'global'  
spam()  
print(eggs)

4 List

\*A List value look like this ['cat', 'bat', 'rat', 'elephant']

-Cách khai báo List : []

\*List Concatenation and List Replication

catNames =[]

names = input()

catNames = catNames +[names]

>>> **[1, 2, 3] + ['A', 'B', 'C']**[1, 2, 3, 'A', 'B', 'C'] -> dạng list với nhau có ngoặc mới concatenate với nhau đc

>>> **['X', 'Y', 'Z'] \* 3**['X', 'Y', 'Z', 'X', 'Y', 'Z', 'X', 'Y', 'Z']

\*The Multiple Assignment Trick

>>> **cat = ['fat', 'gray', 'loud']**>>> **size = cat[0]**>>> **color = cat[1]**>>> **disposition = cat[2]**you could type this line of code:  
>>> **cat = ['fat', 'gray', 'loud']**>>> **size, color, disposition = cat**

The number of variables and the length of the list must be exactly equal, or Python will give you a  
ValueError:

\*enumerate() with List

-On each iteration of the loop, enumerate() will return  
two values: the index of the item in the list, and the item in the list itself

-Dùng với vòng lặp for

Cấu trúc : for index ,item in enumerate(tên list):

\*Using the random.choice() and random.shuffle()

- **random.choice(array): chọn ngẫu nhiên trong list**

-random.shuffel : tráo cả list lun !!!

\*Using remove :để xóa item ra khỏi list

\*index() để tìm value

\*append() and isert() để thêm value vào list

+append : là add vào cuối danh sách

+insert(index,value) : add vào vị trí index

* Phương thức này của List only dùng cho string thì lỗi

\*sort()

-Cách dùng : default là giá trị tăng (123,abc,...)

-Đảo ngược sort(reverse =True)

-sort() uses “ASCIIbetical order” rather than actual alphabetical order for sorting strings. This  
means uppercase letters come before lowercase letters

>>> **spam = ['Alice', 'ants', 'Bob', 'badgers', 'Carol', 'cats']**>>> **spam.sort()**>>> **spam**['Alice', 'Bob', 'Carol', 'ants', 'badgers', 'cats']

If you need to sort the values in regular alphabetical order, pass str.lower for the key keyword argument in  
the sort() method call.  
>>> **spam = ['a', 'z', 'A', 'Z']**>>> **spam.sort(key=str.lower)**>>> **spam**['a', 'A', 'z', 'Z']

Note:

+A list value is a mutable data type: it can have values added ,remove ,..

+A string is a immutable data type it cannot be changed

=>The proper way to “mutate” a string is to use slicing and concatenation to build a new string by copying  
from parts of the old string. Enter the following into the interactive shell:  
>>> **name = 'Zoph**

**5.Dictionary**

**-Dictionary is muatable**

**-Cach goi dictionary{‘key’:’value’}**

**🡪 dictionary [key] = value**

**🡪Chỉ gọi key in dictionary chứ ko value in đc**

- Unlike lists, items in dictionaries are unordered. The first item in a list  
named spam would be spam[0]. But there is no “first” item in a dictionary.  
While the order of items matters for determining whether two lists are  
the same, it does not matter in what order the key-value pairs are typed  
in a dictionary

-dictionary.values() :lấy values ra

Vd:

**spam = {'color': 'red', 'age': 42}**>>> **for v in spam.values():**... **print(v)**

**for k in spam.keys(): \\ lấy key ra**... **print(k)**color  
age  
>>> **for i in spam.items(): \\ lấy ra cả hai lun**... **print(i)**('color', 'red')  
('age', 42)

-Nếu ko có key trong dictionary thì có thể dùng get(‘key’,số) số trả về số này nếu ko có key đấy

>>> **picnicItems = {'apples': 5, 'cups': 2}**>>> **'I am bringing ' + str(picnicItems.get('cups', 0)) + ' cups.'**'I am bringing 2 cups.'  
>>> **'I am bringing ' + str(picnicItems.get('eggs', 0)) + ' eggs.'**'I am bringing 0 eggs.'

6 OOP in Python

-Create a class : class Name

VD:

class MyClass:  
  x = 5

p1 = MyClass()  
print(p1.x)

-constructor : def \_\_init\_\_()

Cấu trúc : def \_\_init\_\_(self, atribute,attribute,….)

VD:

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age

Self : thay cho this kieu the 😊 (note không nhất thiết là chữ self)

- self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class

-self sử dụng để reference attribute của class nên bắt buộc function nào liên quan đến object (không phải static method) thì đều cần self

🡪 nó phải là first paremeter trong tất cả function , nó có thể có tên khác nhau cũng được

VD:

class Person:  
  def \_\_init\_\_(mysillyobject, name, age):  
    mysillyobject.name = name  
    mysillyobject.age = age  
  
  def myfunc(abc):  
    print("Hello my name is " + abc.name)  
  
p1 = Person("John", 36)  
p1.myfunc()

## -Object Methods

## Cấu trúc def function(self)

VD :

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age  
  
  def myfunc(self)://truyền self của object nãy  
    print("Hello my name is " + self.name)  
  
p1 = Person("John", 36)  
p1.myfunc()

-Update attribute

+normally we have to write a function like :

Vd:

def updateName(this,newName):  
 this.name = newName

c = Person("Nhan Vu",21)  
c.updateName("Minh")  
print(c.name , c.age)

+Python is more dynamic

c.name = "Minh"  
c.age = '100'

-Static method : method that no attach to any individual Object but call bt the Class itself

+vd: function ko có selfl làm paremeter

@staticmethod  
def read\_person():  
 print("Mother fucker ")

c = Person("Nhan Vu",21)  
print(c.name , c.age)  
  
  
Person.read\_person()

🡺Phải khai báo @staticmethod

🡪Khi gọi phải là class gọi ko phải object

-Get/set trong python

+Khá là đơn giản

+Get : object.attribute

+Set: object.attribute = .......

+Nếu muốn xem các mà get và set chạy

\*UNDERSCORE FUNCTION (phương thức ko cần gọi)

-Def \_\_str\_\_(self) : return string dùng để display info

🡪Khong can phải goi no

Vd:

def \_\_str\_\_(self):  
 return self.name + " "+ str(self.age)

khi gọi trong hàm main chỉ cần print(c)

Vd:

c = Person("Nhan Vu",21)  
  
print(c)

\*Kế thừa trong Python

- class Object(Father):

Kế thừa thằng Father

\*String manupliation

+The isX() Methods

These methods return a Boolean value that describes the nature of the string. Here are some  
common is*X* string methods:  
**isalpha()** Returns True if the string consists only of letters and isn’t blank  
**isalnum()** Returns True if the string consists only of letters and numbers and is not blank  
**isdecimal()** Returns True if the string consists only of numeric characters and is not blank  
**isspace()** Returns True if the string consists only of spaces, tabs, and newlines and is not blank  
**istitle()** Returns True if the string consists only of words that begin with an uppercase letter followed by  
only lowercase letters

-join() : hợp các string trong list string lại

Cấu trúc :

‘ (mối ở giữa) ’.join([list string])

Vd:

>>> **', '.join(['cats', 'rats', 'bats'])**'cats, rats, bats'  
>>> **' '.join(['My', 'name', 'is', 'Simon'])**'My name is Simon'  
>>> **'ABC'.join(['My', 'name', 'is', 'Simon'])**'MyABCnameABCisABCSimon'

-Split :tách string ra thành list string

VD:

>>> **'MyABCnameABCisABCSimon'.split('ABC')**['My', 'name', 'is', 'Simon']  
>>> **'My name is Simon'.split('m')**['My na', 'e is Si', 'on']

-strip() xóa khoảng trắng