Basic Python

5 building block

1.variable

2.datype

3.data structire

4.function

5.control flow

6.OOP

1.valiable

```
my_name = "toy"
age = 44
gpa = 3.75
movie_lover = True

```

```
my_name

```

```
'toy'
```

```
print(my_name,age,gpa,movie_lover)
```

```
toy 44 3.75 True
```

```
s23 = 5552
```

บันทึกสำเร็จแล้ว ×

```
# remove variable (ตัวแปร)
del s23
```

```
age +=1
age +=1
age -=1
print(age)
```

```
45
```

Data type

```
# int float str bool
```

check data type

```
print(type(age))
print()

```

```
<class 'int'>
```

```
# convert type
x = 100
print(x,type(x))
x = str(100)
print(x,type(x))
```

```
100 <class 'int'>
100 <class 'str'>
```

```
# convert boolen true -> int 1
y = False # T=1 ,F=0
print(y,type(y))
y1 = int(y)
print(y1, type(y))
```

```
False <class 'bool'>
0 <class 'bool'>
```

```
# string : concat ต่อกัน
```

บันทึกสำเร็จแล้ว ×

```
'hellohello'
```

```
#  type hint
age : int = 34
```

```python
my_name : str = "toy"

```

```python
print (age , type (age))
```

```
34 <class 'int'>
```

2.function

```python
print(pow(5,2))
```

```
25
```

```python
# greeting : def สร้าง function ใหม่
def greeting () :
    print( "hello!" )

```

```python
greeting()
```

```
hello!
```

```python
#Ex2  input data เพิ่ม  parameter

def greeting (name) :
    print( "hello!" + name )
```

```python
greeting("Lisa")
```

```
hello!Lisa
```

```python
# EX3 เปลี่ยน parameter

def greeting (name= "john" , location = "London") :
    print( "hello!" + name )
    print( " he is at " + location)
```

บันทึกสำเร็จแล้ว ×

```
hello!kook
 he is at London
```

```
#สร้างฟังก์ชั่น บวกเลขเอง
def add_two_nums( x , y ) :
    print (x+y)

```

```
add_two_nums(5,6)
```

```
11
```

```
# ให้ function return ค่า = ส่งค่าไป result ไปฝากตัวแปรใหม่
def add_two_nums( x , y ) :
    return x+y



```

```

def add_two_nums( x , y ) :
    print ( " you input two number for add")
    return x+y
```

```
result = add_two_nums(5,6)
print (result)

```

```
you input two number for add
11
```

```
#### Use hint

def add_two_nums( x : int  , y : int ) :
    return (x+y)
```

```
add_two_nums(1,3)
```

```
4
```

บันทึกสำเร็จแล้ว ×

```
w line

text = "hello world"

long_text = """ this is a
very long text """

```

```
print ( text )
print ( long_text )
```

```
hello world
 this is a
very long text
```

```
### fstring <-  string template

my_name = " kook prin"
location  = "thai"

text = f" hi my name is {my_name} and i live in {location}"

print(text)
```

```
 hi my name is  kook prin and i live in thai
```

```
text = "a duck walk in to a bar"
print(text)
```

```
a duck walk in to a bar
```

```
len(text)
```

```
23
```

```
# slicing : ดึงตัวอักษรแต่ละตัวออกมา
text[0]
```

```
'a'
```

```
text[-1]
```

```
'r'
```

```
print(text[0],text[-1])
```

```
a r
```

บันทึกสำเร็จแล้ว ×

```

text
```

```
'a duck walk in to a bar'
```

```
text[ 2 : 6 ] #5+1=6
```

```
'duck'
```

```
text[12:18]
```

```
'in to '
```

```
text[7:]
```

```
'walk in to a bar'
```

```
text[ -3 ]
```

```
'b'
```

```
# string is inmutable

name = "python"
print(name[1 :])
name = "c" + name[1: ]
print(name)
```

```
ython
cython
```

```
text = "a duck walk in to a bar"

```

```

```

```
len (text)
```

```
23
```

```
# function vs method
text.upper()

```

บันทึกสำเร็จแล้ว ×

```
text.title()
```

```
'A Duck Walk In To A Bar'
```

```
text.lower()
```

```
'a duck walk in to a bar'
```

```
# Replace คำ
text.replace( "duck","lion" )
```

```
'a lion walk in to a bar'
```

```
# ตัดคำ split = token

words = text.split(" ")
print(words)
```

```
['a', 'duck', 'walk', 'in', 'to', 'a', 'bar']
```

```
words
```

```
['a', 'duck', 'walk', 'in', 'to', 'a', 'bar']
```

```
# เชื่อมคำ เชื่อมด้วยเครื่องหมาย " "

" ".join(words)
```

```
'a duck walk in to a bar'
```

3. data structure

1. list [ ]

2. tuple ()

3. dictionaly {}

4. set {unique}

```
### 3. data structure

#  1. list []
```

บันทึกสำเร็จแล้ว ×

```
#  4. set {unique}
```

```
# list -> เปลี่ยน update ค่าได้
shopping_items = [ "banana","egg","milk" ]

```

```
shopping_items[0] = "mango"


print (shopping_items)
```

```
['mango', 'egg', 'milk']
```

append

```
# list method
# append() = ต่อกัน

shopping_items.append("longan")
print (shopping_items)
```

```
['mango', 'egg', 'milk', 'longan']
```

sort

```
# sort item

shopping_items.sort()
print (shopping_items)
```

```
['egg', 'longan', 'mango', 'milk']
```

```
shopping_items.sort(reverse=True) # desc
print (shopping_items)
```

```
['milk', 'mango', 'longan', 'egg']
```

```
scores = [ 88,90,87,85]
```

```
print ( sum(scores) , min(scores),len(scores))
```

```
350 85 4
```

บันทึกสำเร็จแล้ว

```
print(sum(scores) / len(scores))
```

```
87.5
```

```
# ประกาศ function

```

```
def mean_score(scores) :
  return sum(scores) / len(scores)
```

```
# เรียกใช้งาน mean_score
scores = [ 88,90,87,85]

print(len(scores) , sum(scores) ,
      min(scores), max(scores),
      mean_score(scores))

```

```
4 350 85 90 87.5
```

pop ( )

```
# Remove ตัวสุดท้ายทิ้ว
print(shopping_items)
shopping_items.pop()
shopping_items
```

```
['milk', 'mango', 'longan', 'egg']
['milk', 'mango', 'longan']
```

```
# append
shopping_items.append("egg")
```

remove ("...")

```
# remove ตัวอื่น ไม่ใช่ตัวสุดท้าย
shopping_items.remove("mango")
shopping_items
```

```
['milk', 'longan', 'egg']
```

insert (ตำแหน่ง , "...")

บันทึกสำเร็จแล้ว ×

```
# insert()
shopping_items.insert(1,"vetgetable")
shopping_items
```

```
['milk', 'vetgetable', 'longan', 'egg']
```

```
# list [ ] + list [ ]

items1 = [ "egg" , " milk"]
item2 = [ " banana" , " bread"]
items1 + item2
```

```
['egg', ' milk', ' banana', ' bread']
```

Tuple : เก็บค่าที่ไม่ต้องการเปลี่ยนค่า

```
## tuple =inmutable > no update ค่า
```

```
# tuple ()
tup_item = ("egg" , " banana","peppsi","egg")
```

```
tup_item.count("egg")

```

```
2
```

```
s1 = ("id1" , "12345")
s2 = ("id2" , "44253")
user_pw = (s1,s2)

print(user_pw)
```

```
(('id1', '12345'), ('id2', '44253'))
```

```
# tuple unpacking กระจายค่า
username , password = s1
s1
```

```
('id1', '12345')
```

```
# Unpack 3 varliable

name age gpa = ("kook" 44 3.75 )
```

บันทึกสำเร็จแล้ว ×

```
kook 44 3.75
```

Dictionary

```
# dictionay > mutable

course = {
    "name" : " data science bootcamp",
    "duration" : "4 months",
    "skill " : [ "google sheet" , "dashboard","SQL","R","python" ],
    "student" : 200

}
```

```
course
```

```
{'name': ' data science bootcamp',
 'duration': '4 months',
 'skill ': ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
 'student': 200}
```

```
# ดึงค่าข้อมูลด้วย - - >Key
course["student"]
```

```
200
```

```
# เพิ่ม key ใหม่
course["start time"] = "9am"
course
```

```
{'name': ' data science bootcamp',
 'duration': '4 months',
 'skill ': ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
 'student': 200,
 'start time': '9am'}
```

```
# ลบ key ทิ้ง
del course["start time"]
course
```

```
{'name': ' data science bootcamp',
 'duration': '4 months',
 'skill ': ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
 'student': 200}
```

บันทึกสำเร็จแล้ว ×

```
course["student"] = 300
course
```

```
{'name': ' data science bootcamp',
 'duration': '4 months',
 'skill ': ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
 'student': 300}
```

```
# ดึง subset ออกมา
course["skill "] [-3:]

```

```
['SQL', 'R', 'python']
```

```
list(course.keys())  # อยากรู้ key => column
```

```
['name', 'duration', 'skill ', 'student']
```

```
course.values()
```

```
dict_values([' data science bootcamp', '4 months', ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
200])
```

```
list(course.values())
```

```
[' data science bootcamp',
 '4 months',
 ['google sheet', 'dashboard', 'SQL', 'R', 'python'],
 200]
```

```
# ส่งค่าเป็นคู่ dictionary
```

```
course.items()
```

```
dict_items([('name', ' data science bootcamp'), ('duration', '4 months'), ('skill ', ['google sheet',
'dashboard', 'SQL', 'R', 'python']), ('student', 300)])
```

```
list(course.items())
```

```
[('name', ' data science bootcamp'),
 ('duration', '4 months'),
 ('skill ', ['google sheet', 'dashboard', 'SQL', 'R', 'python']),
 ('student', 300)]
```

```
# get
course.get("student")
```

บันทึกสำเร็จแล้ว ×

```
#----------------------------
```

```
# set {unique}

```

```
course = ["python","R" , "python","SQL"]
course
```

```
['python', 'R', 'python', 'SQL']
```

```
set(course) # ไม่นับซ้ำ
```

```
{'R', 'SQL', 'python'}
```

```
# dictionay , list -> mutable
# tuple , string  - > immutable อัพเดทค่าไม่ได้
```

```
########################

```

Control flow

1. if

2. for

3. while

IF

```
# if

score = 125
if score >= 120 :
  print ( " pass " )
else :
  print ( " fail " )


```

บันทึกสำเร็จแล้ว ×

```
# creat function grate

def grade (score) :
    if score >= 120 :
        print ( " pass " )
    else :
        print ( " fail " )


```

```
# test function

grade (125)
```

```
pass
```

```

def grade (score) :
    if score >= 120 :
        return " pass "
    else :
        return " failed "

## return +> ฝากค่าใหม่ชื่อ result ได้
```

```
result = grade(144)
print(result)
```

```
pass
```

```

 def grade (score) :
     if score >= 120 :
         return " Excellent "
     elif score >= 100 :
         return " Good"
     elif score >= 80 :
         return " Okey"
     else :
         return " failed "

```

บันทึกสำเร็จแล้ว ×

```
print(result)
```

```
Okey
```

```
# Used And , Or

# course == data science , score >= 80 passed
# course == english , score >= 70 passed
```

```
def grade ( course , score ) :
    if course == "english" and score >=70 :
        return "passed"
    elif course == "data sciencr" and score >= 80 :
         return "passed"
    else :
        return " failed "


```

```
result = grade("english",80)
print(result)
```

```
passed
```

for

```
# For
# if score >= 80  , passed


scores = [88 , 90 , 75]

for score in scores :
    print(score)

```

```
88
90
75
```

บันทึกสำเร็จแล้ว ×

```

# update score
scores = [88 , 90 , 75]
```

```

new_score = []   # new_score บรรทัดล่าง เอามาใส่ อัพเดท ตรงนี้
for score in scores :
    new_score.append(score-2)

print( new_score)
```

```
[86, 88, 73]
```

```

```

```
def grading_all(scores) :
    new_score = []
    for score in scores :
     new_score.append(score+2)
    return new_score

```

```
grading_all( [ 75 , 99 , 85 , 84 ] )

```

```
[77, 101, 87, 86]
```

```

```

```
#---------
```

```
# List comprehention
```

```
scores = [88 , 90 , 75]
```

```
for s in scores :
    print (s *2)
```

```
176
180
150
```

บันทึกสำเร็จแล้ว ×

```
[ s *2 for s in scores ]

# for s in scores :
#     print (s *2)

# ctr + /
```

```
[176, 180, 150]
```

```
new_s = [ s *2   for s in scores     ]
new_s
```

```
[176, 180, 150]
```

```
friend = [ "toy" , "kook", "nu" , "nok" , "mee"]
for f in friend :
    print (f.upper())

```

```
TOY
KOOK
NU
NOK
MEE
```

```
[ f.upper()   for f in friend  ]

# for f in friend :
#     print (f.upper())
```

```
['TOY', 'KOOK', 'NU', 'NOK', 'MEE']
```

```

```

While

```
#-------------------
```

```
# while loop
count =0

while count < 5 :
[illegible]
    count = count + 1 # count+=1

```

บันทึกสำเร็จแล้ว ×

```
hello
hello
hello
```

```
hello
hello
```

input

```
# chatbot
user_name = input("What is your name?  ")

```

```
What is your name?  kook
```

```
user_name
```

```
'kook'
```

```
def chatbot():
    fruits = []
    while True :
        fruit = input("what fruit do you want.....? ")                    # user input
        fruits.append(fruit)                                               # sent to upper
        if fruit == "exit" :
            return fruits
```

```

```

```
chatbot()

```

```
what fruit do you want.....? hooo
what fruit do you want.....? hid
what fruit do you want.....? dd
what fruit do you want.....? ddd
what fruit do you want.....? exit
['hooo', 'hid', 'dd', 'ddd', 'exit']
```

```
# HW01 : ordrer pissa
# HW02 : pow ying shup

```

บันทึกสำเร็จแล้ว ×

```
ou ?   "))
```

```
 How old are you ?   55
```

```
type ( age)
```

```
int
```

```
###------------------------------------------------------------------------------------------##
```

```
# OOP  => Object Oriented Programming

# dog class
```

```
class Dog :
   pass
```

```
dog = Dog()
print (dog)
```

```
<__main__.Dog object at 0x7f0435bc47f0>
```

```
class Dog:
    def __init__(self,name) :
        self.name = name

```

```
dog1 = Dog("mila")
dog2 = Dog("joo")
dog3 = Dog("lijo")
```

```
print ( dog1.name,
          dog2.name,
          dog3.name)
```

```
mila joo lijo
```

บันทึกสำเร็จแล้ว ×

```
                                                    :
        self.name = name
        self.age = age
```

```
dog1 = Dog("mila" , 2)
dog2 = Dog("joo" , 3)
```

```
dog3 = Dog("lijo" , 5)
```

```
print ( dog1.name, dog1.age,
        dog2.name,dog2.age,
        dog3.name,dog3.age)

```

```
mila 2 joo 3 lijo 5
```

```
class Employee:
    def __init__(self, id , name , dept , pos):
        self.id = id
        self.name = name
        self.dept = dept
        self.pos = pos
    def hello(self) :
        print ( f"hello! my name is {self.name}")
    def work_hour (self , hours) :
        print (f" {self.name} work for {hours} hours ")


```

```
emp1 = Employee(1,"kook","Finance","CFO")
```

```
emp1.hello()
```

```
hello! my name is kook
```

```
emp1.work_hour(5)
```

บันทึกสำเร็จแล้ว ×

```
print(emp1.name,emp1.pos)
```

```
kook CFO
```

```
emp1.hello()

```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-170-5a7203eb3fbf> in <cell line: 1>()
----> 1 emp1.hello()

<ipython-input-166-e88e1734276d> in hello(self)
      6         self.pos = pos
      7     def hello(self) :
----> 8         Print(f"hello! my name is {self.name}")

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

SEARCH STACK OVERFLOW

แบบมีค่าใช้จ่าย - ยกเลิกสัญญาที่นี่

บันทึกสำเร็จแล้ว ✕

เสร็จสมบูรณ์เมื่อ 16:07 ● ✕