Intro Python Programing

We learn 5 concepts + 1 OOP

- Variable + Basic calculation
- Data Types
- Data Structure
- Control Flow
- Function

↳ 1 cell hidden

Variable + Basic calculation

```python
## + , - , * , /

print(1+1)
print(2*2)
print(100/2)
```

```
2
4
50.0
```

```python
## // : floor division ปัดเศษทิ้งไป
print(100//2)
print(5//2)
```

```
50
2
```

```python
## force result to float
3/1.0
```

```
3.0
```

```python
## % : modulo ดูเศษที่เหลือ
5%2
```

```
1
```

```python
## ** : power
5**2
```

```
25
```

```python
## variable
x = 100
y = 200
print(x+y)
```

```
300
```

```python
## naming variable
## not start with number
## use lower case , _ between word (snake case)
income = 50000
expense = 36000
saving = income - expense
print(saving)

## snake case
income_y2025 = 40000
```

```
14000
```

```python
## remove variable
del saving
```

```python
print(saving)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipython-input-2901794190.py in <cell line: 0>()
----> 1 print(saving)

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

Data Types

- int
- float
- str
- bool

```
## dynamic type
name = "toy" # str
text = "I'm loving McDonald" # single quote in double quote
age = 37 # int
gpa = 3.41 # float
fav_lang = "Python"
movie_lover = True # bool

print(name, text, age, gpa, fav_lang, movie_lover)
```

```
toy I'm loving McDonald 37 3.41 Python True
```

```
## check type of variable
type(name)
```

```
str
```

```
## type hint
name: str = "Kasidis Toy"
type(name)
```

```
str
```

```
## convert type
str(100) # change int to str
```

```
'100'
```

```
int(False) #change bool to int
```

```
0
```

```
# int() , float() , str() , bool()
bool(0)
```

```
False
```

```
## f-string template
name = "toy"
age = 37
lang = "R"

text = f"Hi! My name is {name}. I'm {age} yearsold. My favorite language is {lang}"
print(text)
```

```
Hi! My name is toy. I'm 37 yearsold. My favorite language is R
```

Data Structure

- list []
- tuple ()
- dictionary {}
- set

↳ 82 cells hidden

Control Flow

- if
- for
- while

```python
## if
score = 82

if score >= 80:
    print("passed")
else:
    print("failed")
```

```
passed
```

```python
## multiple grades
score = 75

if score >= 80:
    print("A")
elif score >= 70:
    print("B")
elif score >= 60:
    print("C")
else:
    print("D")
```

```
B
```

```python
## get input from user

username = input("what's your name: ") ## input() is string only
age = int(input("How old are you: "))
```

```
what's your name: toy
How old are you: 37
```

```python
print(username)
```

```
toy
```

```python
## multiple conditions
## night club
has_id = True
age_over_18 = False
VIP_customer = False

if has_id and (age_over_18 or VIP_customer):
    print("Welcome to the night club")
else:
    print("Please come next time")
```

```
Please come next time
```

```python
## for & while
items = ["egg", "milk", "bread"]

for item in items:
    print(item.upper())
```

```
EGG
MILK
BREAD
```

```python
items = ["egg", "milk", "bread"]

for item in items:
    if item == "milk":
        print("CP Meiji")
    else:
        print(item)
```

```
egg
CP Meiji
bread
```

```python
## list comprehension

## want to change all items to Upper case : Normal Way
result = []

for item in items:
    result.append(item.upper())

result
```

```
['EGG', 'MILK', 'BREAD']
```

```python
items
```

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

```python
## list comprehension

## want to change all items to Upper case : list comprehension Way
[item.upper() for item in items]
```

```
['EGG', 'MILK', 'BREAD']
```

```python
## while loop
while True:
    user_input = input("Play on: Y/N ")

    if user_input == "N":
        print("Good Bye")
        break
    else:
        print("Gooooo!")

```

```
Play on: Y/N Y
Gooooo!
Play on: Y/N Y
Gooooo!
Play on: Y/N N
Good Bye
```

```python
## while another example
count = 0

while count < 5:
    print("hi!")
    count += 1
```

Function

```
def function_name ():
    pass
```

```python
## create our first function

def greeting():
    """
    say hi to our friends
    """
    print("hello")
```

```python
greeting()  ## call function
```

```
hello
```

```python
## function can take inputs
def greeting2(name):
    """
    say hi to our friends
    """
    print(f"hello {name}")
```

```python
greeting2("toy")
```

```
hello toy
```

```python
## new function cube()

def cube(base, power):
    return base**power
```

```python
result = cube(5,4)

result
```

```
625
```

```python
## function can return multiple values
## using tuple unpacking
def random_fun(base):
    return base**2, base**3, base**4
```

```python
x, y, z = random_fun(5)

print(x, y, z, x+y+z)
```

```
25 125 625 775
```

```python
greeting2("jisoo")
```

```
hello jisoo
```

```python
cube(10,3)
```

```
1000
```

```python
## function is reusable
```

```python
## build my own chatbot

def chat():
    print("Hello!")

    orders = {"hawaiian":0, "cheese":0, "salad":0}
    bill = # total money (spending)

    while True:
        user_name = input("what's your name: ")
        print(f"Welcome to our Pizza restaurant {user_name}")
        ## other inputs...

        ## end process
        if user_input == "Check bill":
            ## calculate total money spending
            print("Thank you very much!")
            break

```

OOP

Object Oriented Programming

```python
## create Class
class Employee():
    """ A new employee
    """
    def __init__(self, name, age, city):
        self.name = name ## attribute
        self.age = age
        self.city = city

    def working(self):  ## method for Employee()
        print("I am working now!")
```

```python
toy = Employee("Kasidis", 37, "Bangkok")
```

```python
type(toy)
```

```
Employee
def __init__(name, age, city)

A new employee
```

```
toy.age
```

```
37
```

```
toy.working()
```

```
I am working now!
```