

The background is a vibrant purple gradient. It features numerous out-of-focus light circles (bokeh) in shades of white, light purple, and cyan. Overlaid on this are several thin, white circular lines and arcs, some of which have small tick marks, resembling a stylized clock or a technical diagram. A large, dark purple circular area on the right side contains the text.

# PYTHON FUNCTION **FUNCTION #2**

Sirinthorn Cheyasak,  
Asst.Prof.



# Python Functions

# FUNCTION ARGUMENTS

1

Required Arguments

2

Keyword Arguments

3

Default Arguments

4

Variable-length Arguments



# 1

## REQUIRED ARGUMENTS

- Required arguments are the arguments passed to a function in **correct positional order**. Here, the **number of arguments in the function call should match exactly with the function definition**.
- To call the function `printme()`, you definitely need to pass one argument, otherwise it gives a syntax error as follows –

```
def printme(str,num ):
    print (str,",lucky number is",num )

str = "My name is ning"
num = 99
printme(str,num)
```

My name is ning ,lucky number is 99

## 2

# KEYWORD ARGUMENTS

- This allows you to **skip arguments or place them out of order** because the Python interpreter is able to use the keywords provided to match the values with parameters. You can also make keyword calls to the `printinfo()` function in the following ways –

```
def printinfo( name, age ):  
    print ( "Name: ", name)  
    print ( "Age: ", age)  
  
printinfo(age=50, name="ning")
```

```
Name: ning  
Age: 50
```

- A default argument is an **argument that assumes a default value if a value is not provided** in the function call for that argument. The following example gives an idea on default arguments, it prints default age if it is not passed –

```
def printinfo(name, age = 35):  
    print ("Name: ", name)  
    print ("Age ", age)  
  
printinfo(age=50, name="nung-ning")  
printinfo(name="sirinthorn")
```

```
Name:  nung-ning  
Age   50  
Name:  sirinthorn  
Age   35
```



# 4

## VARIABLE-LENGTH ARGUMENTS

In Python, we can pass a variable number of arguments to a function using special symbols. There are two special symbols:

- **\*args (Non-Keyword Arguments)**

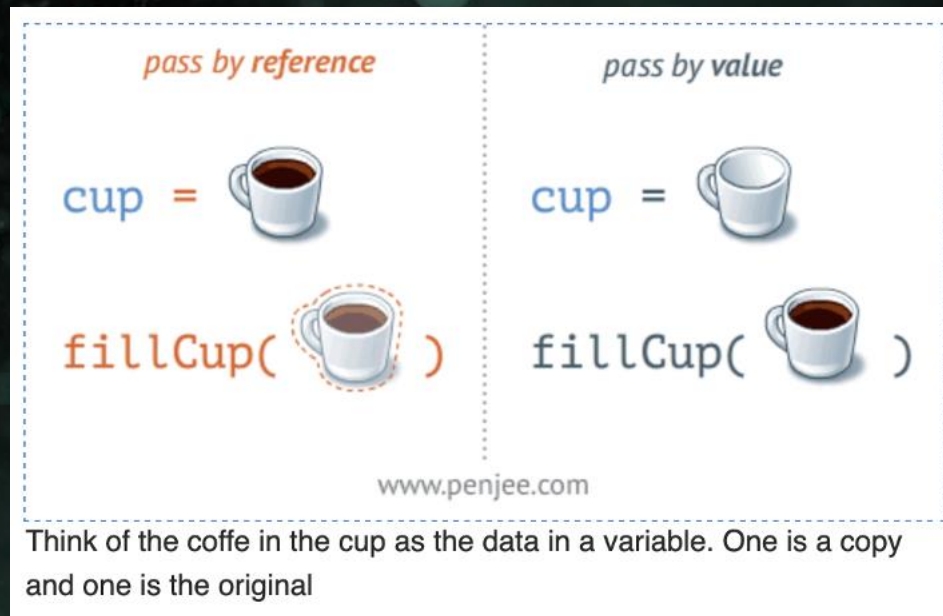
```
def myFun(*argv):  
    for arg in argv:  
        print(arg)
```

```
myFun('Hello', 'Welcome', 'to', 'GeeksforGeeks')
```

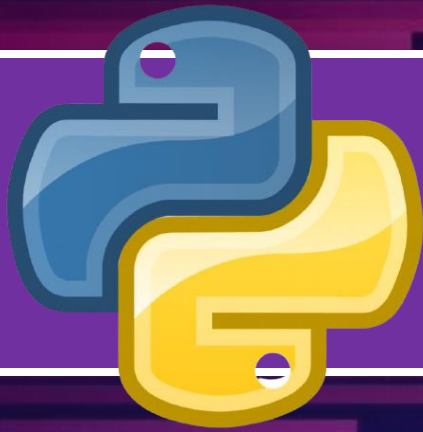
```
Test : ('Hello', 'Welcome', 'to', 'Python')  
Hello  
Welcome  
to  
Python
```

# PASS BY VALUE AND BY REFERENCE

- **Pass by value**, the function receives a copy of the argument objects passed to it by the caller, stored in a new location in memory.
- **Pass by reference**, the box (the variable) is passed directly into the function, and its contents (the object represented by the variable) implicitly come with it. Inside the function context, the argument is essentially a complete alias for the variable passed in by the caller. They are both the exact same box, and therefore also refer to the exact same object in memory.



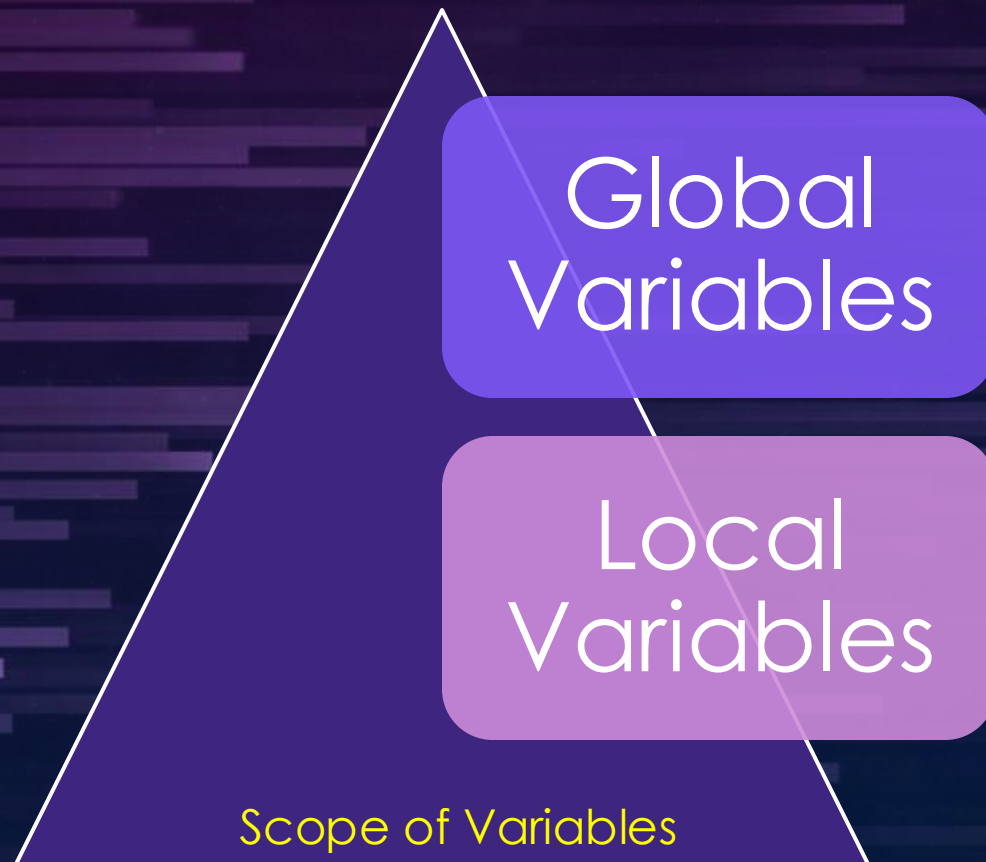




# Scope of Variables

# SCOPE OF VARIABLES

All variables in a program may not be accessible at all locations in that program. This depends on where you have declared a variable.



# SCOPE OF VARIABLES EXAMPLE

```
def function1(n1,n2,n3) :  
    n4,n5,name = 11,22,"Ning"  
    n1 = 10000  
    n2,n3 = 20000, 30000
```

```
def function2(n1,n2,n3) :  
    n4,n5,name = 33,44,"Sirinthorn"  
    n1 = 11111  
    n2,n3 = 22222, 33333
```

```
n1 = 100  
n2 = 200  
n3 = 300  
n4,n5 = 400,500  
name = "Python"  
function1(n1,n2,n3)  
print(n1,n2,n3,n4,n5,name)  
function2(n1,n2,n3)
```



# SCOPE OF VARIABLES EXAMPLE

```
def fn1():  
    print(word)  
  
word = "I love python" #s is global variable.  
fn1()
```

I love python

```
def fn2():  
    word = "So funny" # s is local variable.  
    print(word)  
  
word = "I love COM1"  
fn2()  
print(word)
```

So funny  
I love COM1

# SCOPE OF VARIABLES EXAMPLE

```
def myfunc():
```

```
    print(word)
```

```
    word = "Com1"
```

← global value

```
    print(word)
```

← assigning a new = creating a local variable

```
word = "I love python"
```

```
myfunc()
```

```
print(word)
```

```
-----  
UnboundLocalError                                Traceback (most recent call last)
```

```
<ipython-input-5-4a6c057e1eb2> in <module>()  
      4     print(word)
```

```
      5     word = "I love python"
```

```
----> 6     myfunc()  
      7     print(word)
```

```
<ipython-input-5-4a6c057e1eb2> in myfunc()  
      1     def myfunc() :
```

```
----> 2     print(word)
```

```
      3     word = "Com1"
```

```
      4     print(word)
```


```
      5     word = "I love python"
```

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

# SCOPE OF VARIABLES EXAMPLE

In python, we can use the global variable, we have to use the keyword "**global**", as can be seen in the following example:

```
def myfunc():  
    global word  
    print(word)  
    word = "Me too"  
    print(word)  
  
word = "I love python"  
myfunc()  
print(word)
```



The diagram illustrates the use of the `global` keyword. Two yellow arrows point from the `global word` line in the function definition to two separate white boxes, each containing the text `global value` in red. This indicates that the function is accessing the global variable `word` both when it prints its value and when it reassigns it.

```
I love python  
Me too  
Me too
```



# SCOPE OF VARIABLES EXAMPLE

```
def func1():  
    s = "I am globally not known"  
    print(s)  
  
func1()  
print(s)
```

```
Traceback (most recent call last):  
  File "main.py", line 6, in <module>  
    print (s)  
NameError: name 's' is not defined
```

# ACTIVITY-1 : SCOPE OF VARIABLES

**What is an output of following this program?**

```
def func1(x, y):  
    global a  
    a = 42  
    x,y = y,x  
    b = 33  
    b = 17  
    c = 100  
    print(a,b,x,y)  
  
a,b,x,y = 1,15,3,4  
func1(17,4)  
print(a,b,x,y)
```

# ACTIVITY1-2 : SCOPE OF VARIABLES

**What is an output of following this program?**

```
def fn1(a,b):  
    a = a + 10  
    b = a * 2  
    print(a,b,c,d)  
def fn2(a,b,c):  
    a = b + c  
    b = a + 2  
    c = a + b  
    return(a,b,c)  
a,b,c,d = 10,5,2,4  
fn1(a,b)  
a,b,c = fn2(a,b,c)  
print(a,b,c,d)
```





# Return multiple values

# RETURN WITH MULTIPLE VALUES

In python, we can use the **return()** statement to return **multiple values** back to a main program.

```
def getinfo() :  
    id = input("Enter ID : ")  
    name = input("Enter Name : ")  
    subject = input("Enter Subject : ")  
    score = float(input("Enter total score : "))  
    return(id,name,subject,score)
```

What is a calling statement of getinfo() function?

# RETURN WITH MULTIPLE VALUES

In python, we can use the **return()** statement to return **multiple values** back to a main program.

```
def getinfo() :  
    id = []  
    name = []  
    dept = []  
    gpa = []  
    with open("student.txt", "r") as file :  
        data = file.read().splitlines()  
        for line in data :  
            item = line.split()  
            id.append(item[0])  
            name.append(item[1]+" "+item[2])  
            dept.append(item[3])  
            gpa.append(item[4])  
    return(id, name, dept, gpa)
```

What is a calling statement of getinfo() function?



# ACTIVITY2 OF WEEK14



จงเขียนโปรแกรมแบบฟังก์ชัน กำหนดให้มีการทำงานแยกเป็นฟังก์ชันดังนี้

- ฟังก์ชัน getdata() อ่านข้อมูลจากแป้นพิมพ์ดังตัวอย่าง

- ฟังก์ชัน write2file () แสดงข้อมูลดังตัวอย่างออกแฟ้มข้อมูล โดยแยกตามประเภทของรายการ โดยรายการฝาก (D) แสดงข้อมูลออกแฟ้มข้อมูล deposit.txt ส่วนรายการถอนแสดงออกแฟ้มข้อมูล withdraw.txt

Enter transaction amount : 5

-----  
Detail : Buy art toy

Type(D/W) : w

Total amount : 550

-----  
Detail : From my mom

Type(D/W) : d

Total amount : 5000

-----  
Detail : Popular vote

Type(D/W) : W

Total amount : 200

-----  
Detail : From my sister

Type(D/W) : d

Total amount : 2000

-----  
Detail : Buffet

Type(D/W) : W

Total amount : 299

deposit.txt

1      From my mom      D    5000.00

2      From my sister D    2000.00

3

withdraw.txt

1      Buy art toy      W    550.00

2      Popular vote      W    200.00

3      Buffet      W    299.00

4

จงเขียนโปรแกรมเก็บข้อมูลพนักงาน กำหนดให้มีการทำงานแยกเป็นฟังก์ชันดังนี้

- ฟังก์ชัน `getdata()` อ่านข้อมูลพนักงานจากแฟ้มข้อมูล `profile.txt` ซึ่งมีข้อมูลดังตัวอย่าง พร้อมทั้งคำนวณหารายได้สุทธิของพนักงาน

รายได้สุทธิ = เงินเดือน + (ยอดขาย \* 0.1)

- ฟังก์ชัน `write2file()` แสดงข้อมูลดังตัวอย่างออกแฟ้มข้อมูล โดยแยกตามประเภทพนักงาน โดยพนักงานประจำ (F) แสดงข้อมูลออกแฟ้มข้อมูล `fulltime.txt` พนักงานชั่วคราวแสดงออกแฟ้มข้อมูล `parttime.txt`

profile.txt						
saved						
1	1001	Panee	Sukdee	P	15000	450000
2	1002	Suwit	Pongnonta	P	18000	560000
3	1003	Ratchata	Meemonkon	F	20000	600000
4	1004	Peerada	Ketpong	P	16000	350000
5	1005	Tawee	Rujirote	F	23000	540000
6	1006	Kunya	Wongpisan	P	19000	490000
7						

↑      ↑      ↑      ↑      ↑      ↑

รหัส   ชื่อ   นามสกุล   ประเภท   เงินเดือน   ยอดขาย



profile.txt

saved

1	1001	Panee	Sukdee	P	15000	450000
2	1002	Suwit	Pongnonta	P	18000	560000
3	1003	Ratchata	Meemonkon	F	20000	600000
4	1004	Peerada	Ketpong	P	16000	350000
5	1005	Tawee	Rujirote	F	23000	540000
6	1006	Kunya	Wongpisan	P	19000	490000
7						

Input file

fulltime.txt

1	Ratchata	Meemonkon	20000.00	600000.00	80000.00
2	Tawee	Rujirote	23000.00	540000.00	77000.00

Output file

parttime.txt

1	Panee	Sukdee	15000.00	450000.00	60000.00
2	Suwit	Pongnonta	18000.00	560000.00	74000.00
3	Peerada	Ketpong	16000.00	350000.00	51000.00
4	Kunya	Wongpisan	19000.00	490000.00	68000.00

Output file





**THANK YOU**  
**Q/A**