LAB ACTIVITY 4(ii):
Writing Functions In Python

Learning Outcomes:

By the end of this laboratory session, you should be able to:

1. **Construct Python function for returning result using return statement**
2. **Follow rules for local and global scope in function**
3. **Construct Tuples and use it in simple Python program**
4. **Construct Dictionary and use it in simple Python program**

Hardware/Software: Computer, Phyton 3.5 or above.

Activity 4F

Activity Outcome: Creating and calling a simple function

Procedure:

Step 1: Open Code editor and type the code based on the following code :

```
#create a function to calculate total of 'a' in a string
#if there are no 'a', a 'none' message will appear

#1st answer using count function
def findA (s):
    total = 0
    x = s.count('a')
    y = s.count('A')
    total = x+y

    if total == 0:
            print ("none")
    return total

#2nd answer using loop
def findA2 (s):
    n = 0
    for i in s:
            if ((i == 'a') or (i == 'A')):
                    n += 1
    return n

inputdata = input ("Please input a phrase: ")
print ("Total of 'a' or 'A' character in your phrase is: ", findA(inputdata))
print ("Total of 'a' or 'A' character in your phrase is: ", findA2(inputdata))
```

Step 2: Save, compile and run the program. Save the program as `Act4F.py`. Display the output in the area below.

Output:

```
Please input a phrase: sAyA suka maanis
Total of 'a' or 'A' character in your phrase is:  5
Total of 'a' or 'A' character in your phrase is:  5
```

Activity 4G

Activity Outcome: Creating and calling a recursive function.

Procedure:

Step 1: Open code editor and type the following code:

```
#Program to find a factorial of a number using recursive function

def factorial(x):
    """This is a recursive function
    to find the factorial of an integer"""

    if x == 1:
        return 1
    else:
        return (x * factorial(x-1)) # it will recursively call itself

num = 3
print("The factorial of", num, "is", factorial(num))
```

Step 2: Save, compile and run the program. Save the program as `Act4G.py`. Display the output in the area below.

Output:

```
The factorial of 3 is 6
```

Activity 4H

Activity Outcome : Follow rules of local and global scope in function.

Procedures:

Step 1: Open code editor and type the following code:

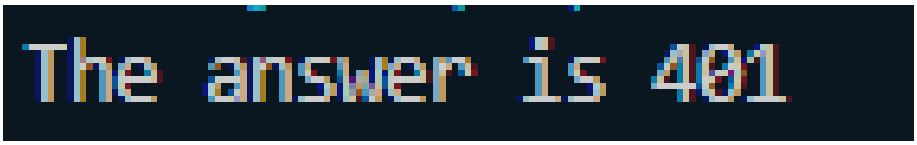
```
# global scope
X = 400            # X and func assigned in module: global

def func(Y):       # Y and Z assigned in function: locals
   # local scope
   Z = X + Y       # X is not assigned, so it's a global
   return Z

func(1)           # func in module: result=401
```

Step 2: Save, compile and run the program. Save the program as `Act4H.py`. Display the output in the area below.

Output:



Activity 4I

Activity Outcome : Follow rules of local and global scope in function (using global variable in a function).
Procedures:
Step 1: Open code editor and type the following code:

```
#First example

y, z = 1, 2        # global variables in module

def all_global():
    global x       # declare globals assigned
    x = y + z      # no need to declare y,z: 3-scope rule
    print ("x in function all_global:",x)

all_global()
print ("x outside function:",x)

#-----------------------------------------
#Second example

def f():
    global s       # declare globals assigned
    print(s)
    s = "Only in spring, but London is great as well!"
    print(s)

s = "I am looking for a course in Paris!"
f()
print(s)
```

Step 2: Save, compile and run the program. Save the program as `Act4I.py`. Display the output in the area below.

Output:

```
x in function all_global: 3
x outside function: 3
I am looking for a course in Paris!
Only in spring, but London is great as well!
Only in spring, but London is great as well!
```

Activity 4J

Activity Outcome : Construct Tuples in Python

Procedures:

Step 1: Open code editor and type the following code:

```
#create a tuple of your wishlist item

myWishlist=("heaven","millionaire","xstudent-employed")  #you can change based on your wish
print (myWishlist)

#access the third item in the tuple
print (myWishlist[2])      #direct access and display

thirdel=myWishlist[2]      #access & store in another variable
print (thirdel)

#change the 2nd element to a new item
wishListNew=list(myWishlist)              #convert the tuple into list first
wishListNew[1]= "married"                 #change the element in the list
myWishlist=tuple(wishListNew)             #convert the list into tuple
print(myWishlist)

#add on a new wishlist to the last element of tuple
wishListNew=list(myWishlist)              #convert the tuple into list first
wishListNew.append("big house")           #add the element in the list using list method
myWishlist=tuple(wishListNew)             #convert the list into tuple
print(myWishlist)

#add on a new wishlist in the third element of tuple
wishListNew1=list(myWishlist)             #convert the tuple into list first
wishListNew1.insert(2,"big family")       #insert the element in the list using list method
myWishlist=tuple(wishListNew1)            #convert the list into tuple
print(myWishlist)

#remove the last element in the tuple
wishListNew2=list(myWishlist)             #convert the tuple into list first
wishListNew2.pop()                        #remove the element in the list using list method
myWishlist=tuple(wishListNew2)            #convert the list into tuple
print(myWishlist)

#delete the tuple
del myWishlist
print (myWishlist)
```

Step 2: Save, compile, and run the program. Save the program as `Act4J.py`. Display the output in the area below.

Output:

```
('Heaven', 'Lots of money', 'Happiness')
Happiness
Happiness
('Heaven', 'Married', 'Happiness')
('Heaven', 'Married', 'Happiness', 'big house')
('Heaven', 'Married', 'big family', 'Happiness', 'big house')
('Heaven', 'Married', 'big family', 'Happiness')

Exception has occurred: NameError ×
name 'myWishlist' is not defined
  File "C:\Users\apitp\Desktop\Diploma Teknologi Maklumat (Teknologi Digital) (DDT)\DFN40323-Programming-Essentials-In-Python\Assignments\Lab Activity 4 (ii)\Act4J.py", line 38, in <module>
```

Activity 4K

Activity Outcome : Construct Dictionary in Python

Procedures:

Step 1: Open code editor and type the following code:

```
#create and display a dictionary
car = {
"brand":"Ford",
"model":"Mustang",
"year":1964
}
print (car)

#access and display a value based on the key
x = car.get("model")              #using get() method
print(x)

x = car ["model"]                 # using key and store in a new variable
print(x)

print(car ["brand"])              # using key and display

#trying to duplicate the dictionary key
thisdict = {
"brand":"Ford",
"model":"Mustang",
"year":1964,
"year":2020
}
print(thisdict)

#add a new item in the dictionary
car["color"] = "red"
print(car)

#update the value
car.update({"year":2020})         #using update() method
print(car)

car["year"] = 2021                #using the key index
print(car)
```

Step 2: Save, compile and run the program. Save the program as `Act4K.py`. Display the output in the area below.

Output:

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
Mustang
Mustang
Ford
{'brand': 'Ford', 'model': 'Mustang', 'year': 2020}
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}
{'brand': 'Ford', 'model': 'Mustang', 'year': 2020, 'color': 'red'}
{'brand': 'Ford', 'model': 'Mustang', 'year': 2021, 'color': 'red'}
```

Activity 4L

Activity Outcome : Construct Dictionary in Python

Procedures:

Step 1: Open code editor and type the following code:

```
#create and display a dictionary
car = {
"brand":"Ford",
"model":"Mustang",
"year":1964
}
print (car)

# display all the items in the dictionary
x = car.items()
print(x)

# display all the values in the dictionary
x = car.values()
print(x) #before the change

car["year"] = 2020
print(x) #after the change

# display all the keys in the dictionary
x = car.keys()
print(x) #before the change

car["color"] = "white"
print(x) #after the change

#print key in dictionary using loop
for c in car:
        print(c)

#print value in dictionary using loop
for d in car:
        print (car[d])
```

Step 2: Save, compile and run the program. Save the program as `Act4L.py`. Display the output in the area below.

Output:

```
dict_items([('brand', 'Ford'), ('model', 'Mustang'), ('year', 1964)])
dict_values(['Ford', 'Mustang', 1964])
dict_values(['Ford', 'Mustang', 2020])
dict_keys(['brand', 'model', 'year'])
dict_keys(['brand', 'model', 'year', 'color'])
brand
model
year
color
Ford
Mustang
2020
```

Activity 4M

Activity Outcome : Construct Dictionary in Python

Procedures:
Step 1: Open code editor and type the following code:

```
#Python program to check whether a given key already exists in a dictionary

d = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

def is_key_present(x):
        if x in d:
                print('Key is present in the dictionary')
        else:
                print('Key is not present in the dictionary')

is_key_present(5)

is_key_present(9)
```

Step 2: Save, compile and run the program. Save the program as `Act4M.py`. Display the output in the area below.

Output:

```
Key is present in the dictionary
Key is not present in the dictionary
```