Lab4

Functions and Modules

https://www.tutorialspoint.com/python
http://marcuscode.com/lang/python/functions

SECTION 1: FUNCSIONS

1. Defining a function

```
def printme( str ):
   "This prints a passed string into this function"
   print (str)
   return
```

<<Try to understand the function elements>>

Calling function

```
printme("This is first call")
printme("Again second")
```

<<fill your result>>

2. Pass by reference

```
# Function definition is here
def changeme( mylist ):
   "This changes a passed list into this function"
   print ("Values inside the function: ", mylist)

   mylist[2]=50
   print ("Values inside the function after change: ", mylist)
   return

# Now you can call changeme function
mylist = [10,20,30]
changeme( mylist )
print ("Values outside the function: ", mylist)
```

<<fill your result>>

One more example where argument is being passed by reference and the reference is being overwritten inside the called function.

```
# Function definition is here
def changeme( mylist ):
   "This changes a passed list into this function"
   # This would assign new reference in mylist
   mylist = [1,2,3,4]
```

```
    print ("Values inside the function: ", mylist)
    return

# Now you can call changeme function
mylist = [10,20,30]
changeme( mylist )
print ("Values outside the function: ", mylist)
```

<<fill your result>>

Exercise: Let's try to pass on of the primitive datatype (e.g. Integer, Float, String, Boolean) and see that it is passed by reference or pass by value

<<fill your result>>

3. Arguments

Required arguments

```
# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print (str)
   return

# Now you can call printme function
printme()
```

<<fill your result>>

<<fix calling printme() to avoid the error>>

Keyword arguments

```
# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print (str)
   return

# Now you can call printme function
printme( str = "My string")
```

<<fill your result>>

```
# Function definition is here
def printinfo( name, age ):
   "This prints a passed info into this function"
   print ("Name: ", name)
   print ("Age ", age)
   return
```

```
# Now you can call printinfo function
printinfo( age = 50, name = "miki" )
```

<<fill your result>>

Default arguments

```
# Function definition is here
def printinfo( name, age = 35 ):
   "This prints a passed info into this function"
   print ("Name: ", name)
   print ("Age ", age)
   return

# Now you can call printinfo function
printinfo( age = 50, name = "miki" )
printinfo( name = "miki" )
```

<<fill your result>>

```
def show_info(name, salary = 84360, lang = "Python"):
    print('Name: %s' % name)
    print('Salary: %d' % salary)
    print('Language: %s' % lang)
    print()
```

```
# calling function
show_info('Mateo')
show_info('Mateo', 105000)
show_info('Danny', 120000, 'Java')
```

<<fill your result>>

```
def create_button(id, color = '#ffffff', text = 'Button', size = 16):
    print('Button ID: %d' % id)
    print('Attributes:')
    print('Color: %s' % color)
    print('Text: %s' % text)
    print('Size: %d px' % size)
    print()

create_button(10)
create_button(11, color = '#4286f4', text = 'Sign up')
create_button(id = 12, color = '#323f54', size = 24)
create_button(color = '#1cb22b', text = 'Log in', size = 32, id = 13)
```

<<fill your result>>

Variable-length arguments

```
# Function definition is here
def printinfo( arg1, *vartuple ):
   "This prints a variable passed arguments"
   print ("Output is: ")
   print (arg1)

   for var in vartuple:
      print (var)
   return

# Now you can call printinfo function
printinfo( 10 )
printinfo( 70, 60, 50 )
```

<<fill your result>>

<<what is the value of arg1 in printinfo(70, 60, 50)>>

<<what is the value of *vartuple in printinfo(70, 60, 50)>>

4. Return statement

```
# Function definition is here
def sum( arg1, arg2 ):
   # Add both the parameters and return them."
   total = arg1 + arg2
   print ("Inside the function : ", total)
   return total

# Now you can call sum function
total = sum( 10, 20 )
print ("Outside the function : ", total )
```

<<fill your result>>

5. Lambda function

```
# Function definition is here
sum = lambda arg1, arg2: arg1 + arg2

# Now you can call sum as a function
print ("Value of total : ", sum( 10, 20 ))
print ("Value of total : ", sum( 20, 20 ))
```

<<fill your result>>

```
f = lambda x: x + 1
print(f(2))
print(f(8))
```

<<fill your result>>

```
g = lambda a, b: (a + b) / 2
print(g(3, 5))
print(g(10, 33))
```

<<fill your result>>

Built-in functions for lambda: filter() and map()

```
numbers = [2, 15, 5, 7, 10, 3, 28, 30]
print(list(filter(lambda x: x % 5 == 0, numbers)))
print(list(map(lambda x: x * 2, numbers)))
```

<<fill your result>>

True or False

```
is_even = lambda n : n % 2 == 0
```

<<fill your result>>

Using Lambda with map (changing the sequence with new value)

```
numbers = [1, 2, 3, 4, 5, 7, 8]
doubled = map(lambda n: n * 2, numbers)
print(list(doubled))
```

<<fill your result>>

Exercise: Describe the different between filter and map

<<fill your answer>>

6. Global and local variables

```
total = 0   # This is global variable.
# Function definition is here
def sum( arg1, arg2 ):
   # Add both the parameters and return them."
   total = arg1 + arg2; # Here total is local variable.
   print ("Inside the function local total : ", total)
   return total

# Now you can call sum function
sum( 10, 20 )
print ("Outside the function global total : ", total )
```

<<fill your result>>

7. Example

```
def count_vowel(str):
    vowel = 0
    for c in str:
        if c in ('A', 'E', 'I', 'O', 'U', 'a', 'e', 'i', 'o', 'u'):
            vowel = vowel + 1
    return vowel
```

<<fill your result>>

```
def area(width, height):
    c = width * height
    return c
```

<<fill your result>>

SECTION 2: MODULES

1. Create a file name "support.py" and write the following code

```
def print_func( par ):
   print("Hello : ", par)
   return
```

Note: create a new file using VScode

Test the module

```
# Import module support
import support

# Now you can call defined function that module as follows
support.print_func("Zara")
```

<<fill your result>>

The from...import Statement
Import specific function, create a file name 'fb.py' with the following code

```
# Fibonacci numbers module

def fib(n): # return Fibonacci series up to n
   result = []
   a, b = 0, 1
   while b < n:
      result.append(b)
      a, b = b, a + b
   return result
```

```
>>> from fb import fib
>>> fib(100)
```

<<fill your result>>

2. Variables and Scope

```
Money = 2000
def AddMoney():
   # Uncomment the following line to fix the code:
   # global Money
   Money = Money + 1

print (Money)
```

```
AddMoney()
print (Money)
```

<<fill your result>>

Now uncomment the line "global Money"

<<fill your result>>

3. See built-in modules

```
# Import built-in module math
import math

content = dir(math)
print (content)
```

<<fill your result>>

```
# Import built-in module math
import fb

content = dir(fb)
print (content)
```

<<fill your result>>

Compare to the previous result

4. Locals and Globals

In your python shell

```
money = 100
def fib(n): # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    print(locals())
    print(globals())
fib(money)
```

<<fill your result>>

SECTION 3: I/O

1. Open and close file

```
# Open a file
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
print ("Closed or not : ", fo.closed)
print ("Opening mode : ", fo.mode)
fo.close()
```

<<fill your result>>

2. Write a file

```
# Open a file
fo = open("foo.txt", "w")
fo.write( "Python is a great language.\nYeah its great!!\n")

# Close opend file
fo.close()
```

<<fill your result>>

3. Read a file

```
# Open a file
fo = open("foo.txt", "r+")
str = fo.read(10)
print ("Read String is : ", str)

# Close opened file
fo.close()
```

<<fill your result>>

Exercise: Try all the mode to open the file. Use your own example to explain the different of each mode. You must display the result with explanation.

<<fill your result>>

<<fill your answer>>