ICT1002 Programming Fundamentals

Lab 3

Topics:

1. Functional Abstraction
2. Functions and modules
3. Higher Order Functions

Warmup exercises:

The following lab assignment requires the use of all topics discussed so far in the module. You may wish to practice some of the concepts with simple exercises before attempting the lab assignment. You are not required to include these exercises in your submission, though you may wish to do so, to help you in the lab test.

1. Define one function to calculate the division of given two numbers, return the result:

2. Define one function to print out all the element for one given list – the input argument is one list and nothing needs to be returned.

3. Evaluate the following code and figure out the output and explain why?

```
def printMax(a, b):
    if a > b:
        print(a, 'is maximum')
    elseif a == b:
        print(a, 'is equal to', b)
    else:
        print(b, 'is maximum')
```

Evaluate the following expressions

printMax(3,4)	printMax(3)
printMax(3,3)	printMax(3,4,5)
printMax(4,4)	printMax('charlie', 'hello')

4. Evaluate what is the output of the following code and understand why

```
def say(message, times = 2):
    print(message * times)

say('Hello')
say('World', 5)
```

5. Evaluate the following code and understand why

```
def func(a, b=5, c=10):
    print('a is', a, 'and b is', b, 'and c is', c)

func(3, 7)
func(25, c = 24)
func(c = 50, a = 100)
```

6. Evaluate what is the output of the following program and explain why?

```
def func(a, b, names):
     a = a+10
     b = b+20
     names[0] = 12
     names[1] = 18
     return a,b

x,y=10,30
fruits=['apple', 'orange', 'banana']
num1, num2=func(x,y,fruits)
print num1, num2
for fruit in fruits:
     print fruit
```

7. According to lab2 Task 1, write one function to decide whether one year is a leap year or not. If it is one leap year, return true. Otherwise, return false.

8. The following examples of writing higher order functions may help you better understand the basics covered by the lecture. Please evaluate the following code, and explain and understand why the output is that:

```
>>> def func(x):
        return x+5

>>> func(20)
```

Ans:

```
>>> def double(x):
        return x*2

>>> map(double, [10,20,30,40])
[20, 40, 60, 80]
```

Ans:

```
>>> sorted(abs, [100,-800,400,-200,50])
```

Ans: If there is an error, then fix it.

```
def printWelcome():
        return 'Welcome:'
def messager(func,str1):
        print func(),str1
```

a.)
```
messager(printWelcome, 'Python')
```

b.)
```
def increment(x):
        return x+100
def double(x):
        return x*2
def getBonus(func, salary):
        bonus = 1000
        if func(salary) > 5000:
                return func(salary)+ bonus*2
        else:
                return func(salary)+bonus

print getBonus(increment, 3000)
print getBonus(double, 3000)
print getBonus(increment, 6000)
print getBonus(double, 6000)
```

Lab Assignment:

To help you better practice, you need to perform a set of tasks in one auto-grading system, CodeDr. CodeDr will provide you immediate feedback of your program, such that you will know the issue of your program. Below is the link for you to access the system:

http://172. 27. 54. 87/codeDr/public

PS. The codeDR system is only accessible via the ICT network. You can access it from outside via the ICT VPN. The ICT VPN account confidential should have been sent to your sit email. If you have not or

have any issue using the ICT VPN, please contact ICT program professional officer: Arthur Loo at weeyeong.loo@singaporetech.edu.sg and Thiru at Thiru@singaporetech.edu.sg.

For your CodeDr account, please get it from your lab instructor in the lab. Please remember your user name and password for the future usage of the system. And, you are not allowed to change the password.

In this lab, you need to finish three tasks in CodeDr system, including the Lab3_Task1, Lab3_Task2 and Lab3_Task3.

P.S. CodeDr system will provide you immediate feedback about the correctness of your program. You can view the feedback of your system by clicking the view detail beside your grade. CodeDr adopts a test cases-based approach to check your program. Your grade is depending on the number of test cases that you program can pass. Note that to train you to have a good programming practice, you have to write your program strictly according to the requirement of the tasks, including your input and output format. If there is any difference (even one more or less space), your program will fail on the test cases. SO TRAIN YOURSELF TO BE AN EXACT THINKER!

To help you practice, you are allowed to do multiple attempts for each task. Enjoy your learning!

You can use either the Python packaged IDLE (Python GUI) or PyCharm to create the Python program. To create one new program, you can create a new file by clicking the NEW FILE under the FILE menu. See below figure.