219116/117 Programming II

Lab 3

Answer question 1 in a file named StudentID_Firstname_lab3_ans.pdf, where StudentID is your KU ID and Firstname is your given name

1. **What Would Python Display?**

Analyze pieces of Python code below and figure out what would Python display? Then, verify your answer by running the code in Python interactive mode (python3 -i) If you got it wrong, put a short note explaining what you misunderstood.

```
>>> lambda x: x  # A lambda expression with one parameter x
```

Nothing - <function <lambda> at 0x00000216918EF4C0>

```
>>> a = lambda x: x  # Assigning the lambda function to the name a
>>> a(5)
```

5-5

```
>>> (lambda: 3)()  # Using a lambda expression as an operator in a call exp.
```

3-3

```
>>> b = lambda x: lambda: x  # Lambdas can return other lambdas!
>>> c = b(88)
>>> c
```

88 - <function <lambda>.<locals>.<lambda> at 0x039A8580>

```
>>> c()
```

88 - 88

```
>>> d = lambda f: f(4)  # They can have functions as arguments as well.
>>> def square(x):
...     return x * x
>>> d(square)
```

16 - 16

```
>>> x = None
>>> x
>>> lambda x: x
```

Nothing - <function <lambda> at 0x03228580>

```
>>> z = 3
>>> e = lambda x: lambda y: lambda: x + y + z
>>> e(0)(1)()
```

4 - 4

```
>>> f = lambda z: x + z
>>> f(3)
```
Error - TypeError: unsupported operand type(s) for +: 'NoneType' and 'int'

```
>>> higher_order_lambda = lambda f: lambda x: f(x)
>>> g = lambda x: x * x
>>> higher_order_lambda(2)(g)  # Which argument belongs to which function call?
```
Error - TypeError: 'int' object is not callable

```
>>> higher_order_lambda(g)(2)
```
4 - 4

```
>>> call_thrice = lambda f: lambda x: f(f(f(x)))
>>> call_thrice(lambda y: y + 1)(0)
```
3 - 3

```
>>> print_lambda = lambda z: print(z)   # When is the return expression of a lambda expression executed?
>>> print_lambda
```
Error - <function <lambda> at 0x01A286E8>

```
>>> one_thousand = print_lambda(1000)
```
None - 1000

```
>>> one_thousand
```
1000 - None

```
>>> def even(f):
...     def odd(x):
...         if x < 0:
...             return f(-x)
...         return f(x)
...     return odd
>>> steven = lambda x: x
>>> stewart = even(steven)
>>> stewart
```
Nothing - <function even.<locals>.odd at 0x01748730>

```
>>> stewart(61)
```
61 - 61

```
>>> stewart(-4)
```
4 - 4

```
>>> def cake():
...    print('beets')
...    def pie():
```

```
...         print('sweets')
...         return 'cake'
...     return pie
>>> chocolate = cake()
```
beels

```
>>> chocolate
```
<function> cake. .... - <function cake.<locals>.pie at 0x01418730>

```
>>> chocolate()
```
sweets - sweets

```
>>> more_chocolate, more_cake = chocolate(), cake
```
sweets - sweets

```
>>> more_chocolate
```
sweets - cake

```
>>> def snake(x, y):
...     if cake == more_cake:
...         return chocolate
...     else:
...         return x + y
>>> snake(10, 20)
```
<function>cake. ... - <function cake.<locals>.pie at 0x03408778>

```
>>> snake(10, 20)()
```
sweets - sweets
cake

```
>>> cake = 'cake'
>>> snake(10, 20)
```
30 - 30

Do not proceed to the next stage until you fully understand the above code and its behavior. Ask the instructor or the TAs if you are confused about any part of the code.

2. lab3.py

Complete the missing code in lab3.py and make sure that it passes all the test cases. **You must use higher-order functions or lambda expressions to get credit for this problem.**

Submission:

- ***Create StudentID_Firstname_lab3 folder, where StudentID is your KU ID and Firstname is your given name***
- ***Put the files to submit, StudentID_Firstname_lab3_ans.pdf and lab3.py, into this folder***
- ***Zip the folder and submit the zip file to the course's Google Classroom before the due date***