Programmers Journal - Unit 3 Lesson 1

**Exercise 8.1**

**from pcinput import getInteger**

**def getMultiplicationtable(number):**

**i = 1**

**while i <= 10:**

**result = i \* number**

**print(f"{i} \* {number} is {result}")**

**i += 1**

**num = getInteger("please enter a number: ")**

**getMultiplicationtable(num)**

**Exercise 8.2**

**from pcinput import getString**

**def getLettersincommon(w1, w2):**

**common = " "**

**for letter in w1:**

**if (letter in w2) and (letter not in common):**

**common += letter**

**return len( common )**

**string1 = getString("please enter string1: ")**

**string2 = getString("please enter string2: ")**

**number = getLettersincommon(string1, string2)**

**if number <= 0:**

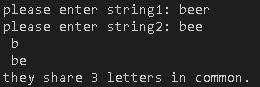
**print("they share no letters in common.")**

**if number == 1:**

**print("they share 1 letter in common.")**

**else:**

**print(f"they share {number} letters in common.")**

****

I was having an error where it was outputting the wrong amount of letters in common, so I changed the function to print the common letters and I noticed that there's a space in front of the common letters and that's being counted toward the amount. So after taking away that space where I defined common, everything seems to be working as intended.

**Exercise 8.3**

**def getGrerory(n):**

**ap = 0**

**for i in range(n):**

**if i%2 == 0:**

**ap += 1/(1 + i\*2)**

**else:**

**ap -= 1/(1 + i\*2)**

**return 4\* ap**

**print(getGrerory(100))**

**Exercise 8.4**

**from pcinput import getFloat**

**from math import sqrt**

**def getQuadratic( a, b, c ):**

**if a == 0:**

**if b == 0:**

**return 0, 0, 0**

**return 1, -c/b, 0**

**discriminant = b \* b - 4\* a \* c**

**if discriminant < 0:**

**return 0, 0, 0**

**elif discriminant == 0:**

**return 1, -b/(2\*a), 0**

**else:**

**return 2, (-b+sqrt(discriminant)) /(2\*a), (-b-sqrt(discriminant)) /(2\*a)**

**num, sol1, sol2 = getQuadratic(getFloat("enter A: "), getFloat("enter B: "), getFloat("enter C: "))**

**if num == 0:**

**print("there are no solutions.")**

**elif num == 1:**

**print("there is 1 solution", sol1)**

**else:**

**print("there is 2 solutions", sol1, sol2)**

**Exercise 8.5**

**from pcinput import getInteger**

**def getNumber(n):**

**while True:**

**num = getInteger(n)**

**if num < 0 or num > 1000:**

**print("please enter a value from 0-1000.")**

**continue**

**return num**

**def main():**

**while True:**

**x = getNumber("enter x (0 for exit): ")**

**if x == 0:**

**break**

**y = getNumber("enter y (0 for exit): ")**

**if y == 0:**

**break**

**if x%y == 0 or y%x == 0:**

**print("Error: numbers cannot be dividers.")**

**continue**

**print("the multipilcation of", x, "and", y, "gives", x\*y)**

**print("Goodbye")**

**if \_\_name\_\_ == '\_\_main\_\_':**

**main()**

**Testing:**

While studying this chapter, I wanted to test out my own function so I created this simple calculator.

**import math**

**from pcinput import getInteger**

**def getAddition():**

**result = (x + y)**

**print(x, "+", y, "is: ", result)**

**def getSubtraction():**

**result = (x - y)**

**print(x, "-", y, "is: ", result)**

**def getMultiply():**

**result = (x \* y)**

**print(x, "times", y, "is: ", result)**

**def getDivision():**

**result = (x / y)**

**print(x, "divided by", y, "is: ", result)**

**while True:**

**data = input("Hello! Would you like the (a)dd, (s)ubtract, (m)ultiply, (d)ivide or (e)xit: ").lower()**

**if data == 'e':**

**break**

**if data in ('a', 's', 'm', 'd'):**

**x = getInteger("please enter number 1 (0 to exit): ")**

**if x == 0:**

**break**

**y = getInteger("please enter number 2 (0 to exit): ")**

**if y == 0:**

**break**

**if data == 'a':**

**getAddition()**

**elif data == 's':**

**getSubtraction()**

**elif data == 'm':**

**getMultiply()**

**elif data == 'd':**

**getDivision()**

**else:**

**print("that is not a valid response.")**

**print("Goodbye!")**

Throughout this lesson, I learned how to use functions better in my code and even create my own functions for different situations. Functions make coding a lot easier and it makes it look neater and easier to understand in my opinion. It really made me think about it and kept me engaged throughout the lesson and made me want to test my own code for making a calculator. I also learned how to use the return() function as well as the main() function to cause a nice flow within the function. I think I need to work on using harder equations in my code such as the quadratic formula as I had a lot of difficulty figuring out how to put it in. I think I had a lot of success in understanding how to make my own functions and how they work.