

Introduction to Programming with Python

Weekly Programming Assignment – Week 5

All solution files [**Exercises 1 – 5**] must be submitted at CodeGrade enabled Link for grading.
The solution code for **exercise 6** must be uploaded in Moodle as “**ex6_week5.txt**” submission

All solutions must be uploaded on or before 13th October 2021 at 11:59 PM

Exercise 1

Write a *procedure* that accepts any 3 numbers (which can be integer or float) as arguments and print the smallest one. Use the main program given here to test your code.

```
10 #mainprogram
11 smallest(45,-24,89.90)
12 smallest(100,20,2)
13
```

Shell ×

```
Python 3.7.9 (bundled)
>>> %Run Ex1_week5.py
-24
2
```

Exercise 2

1. Write a *function* called **isPrime()** with one parameter that accpets any integer as an arguement and **return** whether it is a prime number or not. Prime number is a natural number which is greater than one. Notably, it can be divided by 1 and itself. That is, the number which can not be divided by other integers. *Example: 7 → it can be divided by 1 and itself.* Your submitted code must contain main program which accepts user input and use **isPrime(n)** to **check** and print the returned result (you may require if statement in the main program as well). The sample run is here:

```
Python 3.7.9 (bundled)
>>> %cd 'Z:\Python 2021_Fall\Fa
>>> %Run ex3_p.py
Enter an integer(>=2):4
It is not a prime number
>>> %Run ex3_p.py
Enter an integer(>=2):121
It is not a prime number
>>> %Run ex3_p.py
Enter an integer(>=2):131
It is prime number
```

Exercise 3

Write a program that contains two subprograms:

- isValidTriangle(a,b,c)** that checks the sides of triangle given is valid to form a triangle and return a Boolean value (True or False).
- areaOfTriangle(a,b,c)** which print the area of the triangle provided the isValidTriangle(a,b,c) returns True. The formula for area of triangle is

$$s = (side1 + side2 + side3)/2;$$

$$area = \sqrt{s(s - side1)(s - side2)(s - side3)}$$

Your main program should prompt the user to enter 3 values (sides) as input and call declared subprogram(s) based on user input. The sample run is here.

```
>>> %Run Ex3_week5.py
Side1:3
Side2:4
Side3:5
area: 6.0

>>> %Run Ex3_week5.py
Side1:2
Side2:3
Side3:4
area: 2.9047375096555625

>>> %Run Ex3_week5.py
Side1:2
Side2:3
Side3:5
Can't form a triangle
.
```

Hint: Refer the sample program that call another subprogram and returns Boolean value to proceed for area of the circle (see below). It is given for demonstration purposes only.

```

Thonny - Z:\Python 2021_Fall\Fall 2021_CT60A0203\Week 5\Weekly program
File Edit View Run Tools Help

Ex3_week5_sample.py x
1  def isPositive(x):
2      if x>0:
3          return True
4      else:
5          return False
6  def areaCircle(r):
7      if isPositive(r):
8          print(3.14*(r**2))
9      else:
10         print("r value must be positive")
11
12 #main program
13 a= int(input("Enter the radius:"))
14 areaCircle(a)

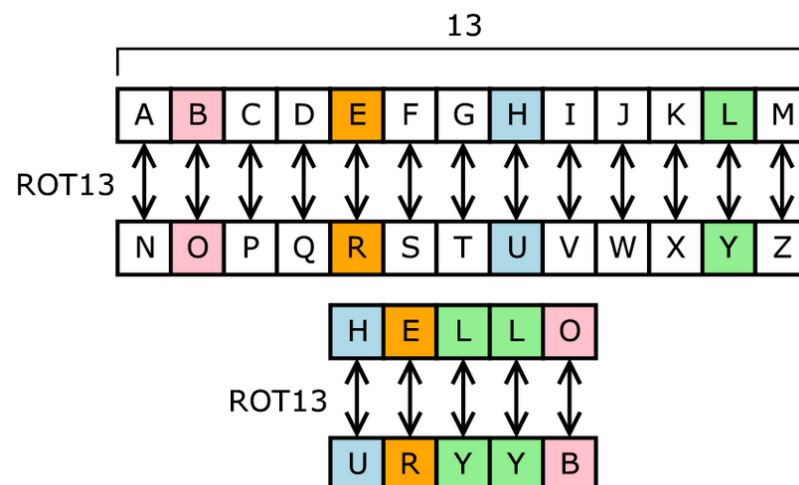
Shell x
Python 3.7.9 (bundled)
>>> %Run Ex3_week5_sample.py
Enter the radius:0
r value must be positive
>>> %Run Ex3_week5_sample.py
Enter the radius:3
28.26
>>> %Run Ex3_week5_sample.py
Enter the radius:-2
r value must be positive

```

Exercise 4

ROT-13 is a very simple cryptographic algorithm, which substitutes each letter (from A to Z) in a string with a letter 13 places further along in the alphabet (Example: A is substituted with N, K with X and T with G) Figure given below demonstrates the use of the algorithm. Note, that ROT-13 is its own inverse, meaning that the same algorithm can be used to code and decode a message (or, `rot13(rot13(string)) == string`).

Write a *function* **rot13(string)**, which returns the string given as an argument in ROT-13 decoded form. You can assume, that the string contains lowercase letters from 'a' to 'z' only.



The sample run is here. The input given here is for demonstration purposes only. Your code will be checked with different types of input

```
9
10 #main program
11 s = input("Enter the string to encrypt:")
12 print(rot13(s))
13
14
```

Shell x

Python 3.7.9 (bundled)

```
>>> %Run Ex4_week5.py

Enter the string to encrypt:hello
uryyb

>>> %Run Ex4_week5.py

Enter the string to encrypt:welcome
jrypbzr

>>> |
```

Exercise 5

String B is said to be a permutation of string A if an equal number of each character in A can be found in B and vice versa. Hence, B is said to be a rearrangement of A. These string permutations are sometimes called *anagrams*.

For example:

- "ABBA", "BAAB" and "ABAB" are all permutations of string "AABB"
- "elvis" and "lives" are permutations of string "ilves"

Write a *procedure* called **anagramAB()** that with two parameters namely A and B, and proceeds to print whether B is a permutation of A. To confirm this, the following conditions need to be met:

- There is an equal number of characters in A and B (or the length of A is equal to length of B)
- For every character found in A, there is an equal number of same characters in B.

The main program should prompt the user to ask for inputs (string A and B) and invoke the **anagramAB()** subprogram. The sample run is here.

```
Enter string A :FFA
Enter string B :AFF
AFF is a permutation of FFA

>>> %Run Ex5_week5.py

Enter string A :CCDEA
Enter string B :ADCDD
ADCDD is not a permutation of CCDEA

>>> %Run Ex5_week5.py

Enter string A :ELBOW
Enter string B :BOWEL
BOWEL is a permutation of ELBOW

>>> |
```

Note: A solution should not use sorted() -function to arrange the letters in order and compare those. You must use **nested loops and if statements**.

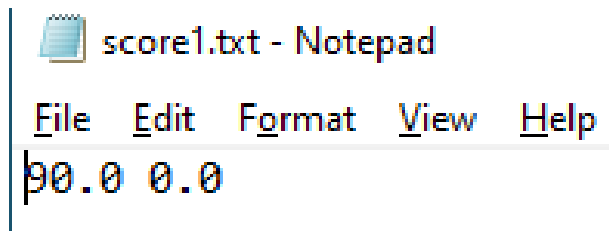
Hint: count() function for string may be useful.

Exercise 6: self-study-based exercise: Will be graded at tutorial session.

Write a program that reads a text file called “**score.txt**” (attached with this exercise) contains students final scores. Then print the lowest, highest score of from that list. The identified lowest and highest score should be written in a new file called “**score1.txt**”. You should use loops and if statements to find the lowest and highest scores. The sample run is here. You are **expected to use subprograms to get full score** (20 marks). Program with no functions may receive at the maximum of 80% of full score for this exercise. That is 15 marks.

```
>>> %Run Ex6_week5.py
The highest score is: 90.0
The lowest score is: 0.0
```

Score1.txt file contains (should be created and written via your code only)



Refer the example code for creating, writing and reading the contents in the file given here + read the **chapter 5 of prescribed textbook** for more details.

```
1 f1 = open("myfile.txt","w") # Creating a file --> myfile.txt
2 f1.write("74") # writing content to the file
3 f1.close() # closing the file
4
5 f2 = open("myfile.txt","r") # open the file in readmode
6 content = f2.readlines() # read the contents
7 print(content) # displaying the read contents
8
```

Exercise / task Number	Codegrade link_Moodle for file solution files upload	Points / Marks
1	Exercise1_Week 5	10
2	Exercise2_Week 5	10
3	Exercise3_Week 5	10
4	Exercise4_Week 5	10
5	Exercise5_Week 5	10
6*	Exercise6_Week 5	20
*Exercise 6 will be checked manually at tutorial session.		