בס"ד

Exercise Nr. 1 in Python

1. Write a Python program which gets three positive integer (and/or float) numbers from the keyboard, into three variables called a, b, and c.
   Using an appropriate Boolean function, the program will check if all those three numbers can be the lengths of the three sides of a triangle. That is, that all the following inequalities need to hold:

   $$a+b>c \ ,a+c>b \ ,c+b>a$$

   If those three inequalities hold, the program will print the following message:

   "they are correct triangle sides' lengths"

   Otherwise, the program will print the following message:

   "they are in error"

   Try the program with the following input values:
   - a = 4 ,b = 3 ,c = 2
   - a = 20 ,b = 3 ,c = 2

2. In the supplied Python program file MenuForTar1.py you will find an example program which implements a textual menu that allows the user to calculate the surface of geometric shapes, in accordance with the menu's options. The calculation of each one of those geometric shapes is done by using appropriate functions.
   You need to add options to the menu in order to allow the user to choose additional 2D and 3D shapes from the menu; for example: sphere, cone, square pyramid, etc. For each one of them, you will have to write an appropriate function; in the case of a 2D shape, the function will calculate its surface, and in the case of a 3D shape, the function will calculate its volume.
   Try the program with the following input values:
   - In the case of a sphere: r = 3.0
   - In the case of a cone: r = 2.5 , h = 4.7
   - In the case of a square pyramid: a = 2.0 , h = 5.2

   Note: go to the Internet (or a high school geometry textbook) in order to find the appropriate formulas, that you will have to implement in Python.

3. Solve the following three paragraphs:
    a. Write a Python program which will execute the following:
        1. gets four values from the keyboard, and checks, using an appropriate Boolean function, if they are numbers (integer and/or float).
        2. chooses, by using an appropriate function, the two middle numbers (they are greater than the minimum and less then the maximum).
        3. prints, by using an appropriate function, those two numbers.

        Write two versions of this program, one version that uses the sort() (or perhaps the sorted()) method upon the list of those four numbers, and a second version that uses no one of those methods.
        Try the program with the following input values: 100,20,35,40.
    b. Write a generalization of the program you wrote above, such that it will execute the following:
        1. gets from the keyboard, a tuple of any length, and checks, using an appropriate Boolean function, if it is a tuple and all its elements are numbers (integer and/or float).
        2. chooses, by using an appropriate function, the two middle numbers, relative to all the other numbers in the input tuple.
        3. prints, by using an appropriate function, those two numbers.

        Write two versions of this program, one version that uses the sort() (or perhaps the sorted()) method upon the list of those four numbers, and a second version that uses no one of those methods.
        Try the program with the following input values: (100, 20, 35, 40, 67, 32).
    c. Write an additional generalization of the program you wrote above, such that it will execute the following:
        1. gets from the keyboard, a tuple of any length, and checks, using an appropriate Boolean function, if it is a tuple that contains values of any data type, and generates a new tuple that contains only the numbers which were in the input tuple.
        2. executes what you already implemented in paragraphs 2 and 3 of the paragraph b' above.

        Try the program with the following input values:
        (100, [20, 35, 'abc'], 40, "my test", 67, 32, 15, 34)
4. Let's assume that positive integer numbers are represented by strings containing only "0" and "1".
   Write the following four functions, called shiftL, shiftR, shiftCL, shiftCR, which receive two parameters: a string called binNr that contains a binary number, and a positive integer N which determines how many digits need to be moved (or shifted). Each one of those functions returns a new binary number which is the result of that shift.

The function shiftL moves the digits of the binary number N places to the left. The function shftR will move the digits of the binary number N places to the right. The function shiftCL moves the digits of the binary number N places to the left, but in a circular manner (those digits that "overflow out" get in from the other side of the binary number).

For example:

```
>>> shiftL("110001110",2)
"000111000"
>>> shiftR("110001110",2)
"001000111"
>>> shiftCL("110001110",2)
"000111011"
>>> shiftCR("1100011",2)
"101100011"
```

Idea for a simple solution: use slices instead of loops.

5. Write a Python program which gets from the keyboard, a list that contains integer numbers, strings, tuples, and lists; for example:

   [1,2,'a',(11,2,'b'),[22,'c'],(33,),['d'],'e']

   The program will do the following (by using appropriate functions)

   a. Creates a list whose elements are the elements of all the tuples nested inside the input list.
   b. Creates a tuple whose elements are the elements of all the lists nested inside the input list.
   c. Creates a tuple whose elements are the numbers contained in the input list, which are not contained in any of the nested lists or tuples.
   d. Creates a list whose elements are the strings contained in the input list, which are not contained in any of the nested lists or tuples.

   The program will print the values returned by all the functions that implement what is required by paragraphs a' to d' above. For example, if the input list is [1,2,'a',(11,2,'b'),[22,'c'],(33,),['d'],'e'], the output printed by the program will be:

   ```
   [11,2,'b',33]
   (22,'c','d')
   ['a','e']
   (1,)
   ```

   Implement all the required functions by using appropriate for loops.

   The program that you will write must be written in a way that it will be possible to run it as a program and also imported as a module.

6. Write a Python program which gets from the keyboard, a list that contains integer numbers, strings, tuples, and lists. Write appropriate functions which will allow the program to count the number of tuples, lists, numbers, and strings, contained in the input list. The program will print the results returned by those functions. Use a dictionary in order to count the different kinds of data contained in the input list, such that the data type name will be the key and the appropriate counter will be the value bound to that key. For example, if we will use the following list:

   L = [1,2,'a',(11,2,'b'),[22,'c'],(33,),['d'],'e']

   The dictionary will look like the following:

   D = {list : 2, int : 2, float : 0, str : 2, tuple : 2}

   The required output printed by the program, for this example will be:

   The contains 2 tuples, 2 lists, 2 numbers, and 2 strings.

   Implement all those functions by using appropriate for loops.

   The program that you will write must be written in a way that it will be possible to be ran as a program and also imported as a module.

7. Write a Python program that receives from the keyboard, the name of some text file, reads it from its beginning to its end, counts the number of occurrences of every word in the text, and prints every word and its number of occurrences, each in a different line.

   You are free to define every function that you think is necessary to write the program. Use the example file shakespeare.txt, provided to you with this exercises sheet.

   Some suggestions for the implementation of your solution:

   a. Use a dictionary in order to store the counters of all the words found in the text.
   b. Use the split method, defined upon strings, in order to decompose every line of text, and transform it into a list of the words contained in the line.
   c. Be aware that not only spaces may be separator characters between words, in the text.

8. Write a Python program which creates a tuple containing N numbers (3 <= N <= 9) which will be randomly chosen from the closed interval [1,9]. The program will <u>not</u> print that tuple (it is advisable that during program development, the program will print it, in order to do your programming task easier).

   The program will do the following:

   a. It will ask the user to enter N integer numbers in the closed interval [1,9] (duplications are allowed), as his guess of the N numbers that the program randomly generated.
   b. It will store the N numbers of the guess in a tuple. Be aware that the places where the numbers are in the tuple is significant.
   c. It will call a function called guessTest which receives to parameters: the first one needs to be the tuple of numbers which were randomly chosen, and the second one should be the user guess. That function will check the correctness of the guess, by value and place, against the tuple of the randomly chosen numbers. That function will return a tuple containing all the guessed numbers which occur in their correct place, and the string "X" in all the places where the guess was not correct.

d. The program will print the tuple returned by the function guessTest, and the percent of guess success.
e. The user will play all the number of times that he will wish. The guesses game will stop when the user will enter the number -1, or when his guess will be completely correct.
   As soon as the program's loop stops, it will print the tuple containing the randomly chosen numbers, which was created at the beginning of the game, and the highest success percent among all the guesses.