Module 3: Deep Dive – Functions, Sorting, Errors and Exception Handling, Regular Expressions and Packages

Assignment

Version 2.0

1. Build an interactive application which should simulate a Quiz contest. The following questions might be asked as input from user:

Choose level (easy, intermediate, and hard): → 3 modes of difficulty and user should input one of these choices.

Please give us the number of question you want to attempt: → No of questions thrown should be the number entered through this prompt.

Specify the question type (multiplication:M, addition:A, subtraction:S, division:D): → One of these operations to be performed.

If the answer is right or wrong, appropriate messages should be printed and move to next question if attempt count is not exceeded.

Hint: Random utility can be used to change complexity of questions.

The program should ask if the user wants to continue even after attempting the number of questions specified and accordingly should loop or terminate.

Sample:

```
Choose level (easy, intermediate, and hard): easy

Please give us the number of question you want to attempt: 3

Specify the question type (multiplication:M, addition:A, subtraction:S, division:D):D

What's 6 divided by 3?

2

That's right -- well done

What's 10 divided by 2?

5

That's right -- well done

What's 18 divided by 3?

6

That's right -- well done

Continue or exit (Continue:C, Exit: E): E
```

2. Write a recursive function to compute x raised to the power of n.

3. Sort the list using lambda function mylist = [["john", 1, "a"], ["larry", 0, "b"]]. Sort the list by second item 1 and 0.

4. Sort the list using operator.itemgetter function mylist = [["john", 1, "a"], ["larry", 0, "b"]]. Sort the list by second item 1 and 0.