QUIZ 7

Due Date : 08.12.2023 - Friday (23:00)
Advisor : R.A. Görkem AKYILDIZ
Programming Language : Python 3.6.8

CASIO
SCIENTIFIC CALCULATOR fx-82ES
NATURAL DISPLAY
M STO RCL STAT CMPLX MAT VCT FIX SCI Math ▼▲ Disp
Math ERROR
[AC] :Cancel
[◀][▶]:Goto

1 Introduction

In this quiz, you are expected to gain some basic experience on exception mechanism of Python and exceptional situations in general. So far most of your tasks were like a sandbox and they were not containing any exceptional cases as their main aim were making you gain some basic experience on Python without minding edge cases, but from now on, as you have learnt about exceptions and exceptional cases, you must mind about them as real life scenarios are not as regular as the tasks that assigned to you so far. So, indeed, it is very crucial practice to think in exceptions and edge cases.

2 Implementation

For this quiz, you are expected to implement a calculator that reads the input file line by line, prints the line to the output and prints either the result or the error according to that calculation. You will deal with four operations (addition, subtraction, multiplication, and division) and exceptional cases about them. The exceptions that must be checked are as follows:

- Checking if program executed with exactly two command line arguments, if it is not, you must printout following error **to the terminal console** and **terminate the program**:
  ERROR: This program needs two command line arguments to run, where first one is the input file and the second one is the output file!
  Sample run command is as follows: python3 calculator.py input.txt output.txt
  Program is going to terminate!
- Checking if the given input file exists, if it does not, you must printout following error **to the terminal console** and **terminate the program**:
  ERROR: There is either no such a file namely <INPUT> or this program does not have permission to read it!
  Program is going to terminate!

  (Note that <INPUT> corresponds to the name of the input file that is given as argument and it must be replaced with the name of the input file while printing.)
- Checking if the line format is correct, if it is not, you must printout following error **to the output file** and continue with the next line:
  ERROR: Line format is erroneous!
- Checking if the first operand is a number, if it is not, you must printout following error **to the output file** and continue with the next line:
  ERROR: First operand is not a number!
- Checking if the second operand is a number, if it is not, you must printout following error **to the output file** and continue with the next line:
  ERROR: Second operand is not a number!
- Checking if the operator is one of the four operations (+, -, *, /), if it is not, you must printout following error **to the output file** and continue with the next line:
  ERROR: There is no such an operator!
- If everything is correct, you must printout the result with an equal sign before it.

Note that you must printout exactly one error per case, the priority order is as follows:

- Checking command line arguments.
- Checking if the given input file exists.
- Skipping empty lines and trimming (actually it is not an error but you must do it before checking any kind of errors after checking if program can successfully boot itself with input and output files).
- Checking the line format.
- Checking if the first operand is number.
- Checking if the second operand is number.
- Checking if the operator is valid.

3 Definition of Input

Each line is assumed to contain one operation with only and exactly two operands and one operator; and its format is as follows:

"<OPERAND1><SPACE><OPERATOR><SPACE><OPERAND2>"

For example "3 + 5" (excluding quote marks).

If format of trimmed version of a line does not matches with the given format, that line must be assumed as an erroneous line, but note that, that line may contain any whitespace characters before or after the operation, you must trim them, you may use strip function for this purpose. Moreover, if a line is empty or if it only contains whitespace characters, it will not be assumed as an erroneous line, you must just skip it.

4 Definition of Output

For each line at the input (excluding the skipped ones due to being empty) you must follow the following format:

Trimmed version of the line
=<RESULT> if the line is not erroneous or error if otherwise.

Say that the current line at the input is "3 + 5", the output will be as follows:

```
3 + 5
=8.00
```

Say that the current line at the input is "3 a 5", the output will be as follows:

```
3 a 5
ERROR: There is no such an operator!
```

Note that you must printout exactly two digits of the fraction part even if the fraction part is less than or more than two digits, the result must be rounded if it contains more than two fraction digits.

5 Restrictions

- Your code must be able to execute on our department's developer server (dev.cs.hacettepe.edu.tr).
- You must obey given submit hierarchy and get score (1 point) from the submit system.
- **You must benefit from exceptions.**
- Your code must be clean, do not forget that main method is just a driver method that means it is just for making your code fragments run, not for using them as a main container, create functions in necessary situations but use them as required.
- You must use comments for this project and you must give brief information about the challenging parts of your code. Do not over comment as it is against clean code approach. Design your comments so that they make your code fully understandable and not excessive for others.

- You can benefit from Internet sources for inspiration but do not use any code that does not belong to you.
- You can discuss high-level (design) problems with your friends but do not share any code or implementation with anybody.
- Do not miss the submission deadline.
- Source code readability is a great of importance. Thus, write READABLE SOURCE CODE, comments, and clear MAIN function. This expectation will be graded as “clean code”.
- Use UNDERSTANDABLE names to your variables, classes, and functions regardless of the length. The names of classes, attributes and methods should obey Python naming convention. This expectation will be graded as “coding standards”.
- You can ask your questions through course’s Piazza group, and you are supposed to be aware of everything discussed in the Piazza group. General discussion of the problem is allowed, but **DO NOT SHARE** answers, algorithms, source codes and reports.
- All quizzes must be original, individual work. Duplicate or very similar quizzes are both going to be considered as cheating.

6 Execution and Test

Your code must be executed under **Python 3.6.8** at **dev.cs.hacettepe.edu.tr**. If your code does not run at department’s developer server during the testing stage, then you will be graded as 0 for code part even if it works on your own machine.
Sample run command is as follows:

- python3 calculator.py input.txt output.txt

7 Grading

Task	**Point**
Cases with No Exceptions	10
Checking Command Line Arguments	15
Checking Input File	15
Skipping Empty Lines and Trimming	15
Checking Line Format	15
Checking First Operand	10
Checking Second Operand	10
Checking Operator	10
Total	100

Note that you must score one at the submit system, otherwise 20% of your grade will be deducted, moreover, you must implement a main function otherwise 10% of your grade will be deducted! There may also be other point deductions if you do not obey the given rules, such as if you do not use exceptions as necessary.

8 Submit Format

File hierarchy must be zipped before submitted (Not .rar, only not compressed .zip files because the system just supports .zip files).

```
- b<StudentID>.zip
     - calculator.py
```