Processing and displaying user inputs part 2

Object Oriented Programming - Programming Coursework #2 (9 pts)

Due Date : May 14, 2020 5pm

1 Task

The objective of this coursework is to gain a hand-on experience on processing numerical values and visualizing the corresponding outputs. The students have to implement three programs. The first program (Program 1) takes from the keyboard, two (integer) numbers, say a and b, in $\{0, \ldots, 9\}$ and prints the sum of all integer numbers that are larger than or equal to a and smaller than an or equal to b (3 pts). The second program (Program 2) takes a positive real number and prints out the square root of that number (3 pts). The third program (Program 3) visualizes three real-valued random numbers that lie in-between 0 and 1, each separated by space (see Sec. 2). The submitted codes have to satisfy the compilation requirements (Sec. 3.1): If the codes do not meet these requirements, the students will receive 0 point. We will test Program 1 and Program 2 with five different user (keyboard) input combinations. If the submitted program generates correct answers for all five input instances, the students will receive 3 points per program (i.e., 9 points for three programs). For Program 3, we will execute it five times and asses the respective program outputs.

2 Program specifications

For Program 1 and Program 2, the input will be provided by the users using the keyboard. After taking the respective inputs, Program 1 will print out (i.e., display in the monitor) an integer number, while Program 2 will print a real number.

For program 1, if a is larger than b, the output should be zero. When the user executes Program 3 multiple times, the visualized random number triples should be different across these runs. For instance if Program 3 prints 0.840188 0.394383 0.783099 in the first run, and prints the same triples in the second to fifth run, the student will receive only 0.6 pts out of 3 pts (only the first run will be counted).

For the results of program 2 and program 3, please visualize 6 digits after the decimal points by rounding the respective values at the 7-th point after the decimal point.

2.1 Sample inputs and outputs

Program 1 input: The user will type two numbers separated by a space and then type 'Return' ('Enter').

```
2 5
```

Program 1 output to be visualized in the monitor:

```
14
```

Note that $14 = 2 + 3 + 4 + 5$.

Program 1 input: The user will type two integers separated by a space and then type 'Return' ('Enter').

```
7 3
```

Program 1 output to be visualized in the monitor:

```
0
```

Note the output is zero since the first number (7) is larger than the second one (3).

Program 1 input: The user will type two numbers separated by a space and then type 'Return' ('Enter').

```
5 5
```

Program 1 output to be visualized in the monitor:

```
5
```

Program 2 input: The user will type a positive real number and then type 'Return' ('Enter').

```
2.1
```

Program 2 output to be visualized in the monitor:

```
1.449138
```

There's no user input for Program 3.
Example Program 3 outputs to be visualized in the monitor for the first run:

```
0.840188 0.394383 0.783099
```

Example Program 3 outputs to be visualized in the monitor in the second run:

```
0.493565 0.313249 0.967869
```

For this CW, your programs do not have to handle exceptions caused by incorrect input data types. For instance, for Program 1 the user wouldn't provide numbers that lie outside $\{0, \ldots, 9\}$

```
-1 11
```

3 What to hand in for assessment

Please submit the source codes for Program 1, Program 2, and Program 3 and a brief report (a single report for all programs) via Black Board. Comments in the codes and the report should be written in English. The report should provide code comments explaining the algorithms. If the submission does not include a report, the maximum possible points will be kept at 40% of the full points. Please format the submission as a single zip file 'StudentID_YourName.zip' that includes the files of the source codes and the report,

e.g.,
20201111_KwangInKim.zip
- Program1.cpp
- Program2.cpp
- Program3.cpp
- Report.pdf

The submitted source codes should contain 'Program1.cpp', 'Program2.cpp', and 'Program3.cpp' files each including the id and name of the submitting student.

In "Program1.cpp"

```
//StudentId YourName
#include <iostream>
Your code here...
```

3.1 Compilation requirements and instruction

Each submitted code needs to be compilable by the GNU C++ compiler and in the Linux environment. Specifically, they should be **compilable and executable in UNI06 server**. Students can verify the correct compilation and execution of their codes by following the instructions provided in 'howtocompile.pdf' document accompanying the CW1 announcement.