CS201 – Spring 2017-2018 - Sabancı University

Homework #4: A Simple Calculator

Due March 28, Wednesday, 19:00 (Sharp Deadline)

Motivation

The aim of this homework is to practice on string member functions and loops. Of course, you will need to use the topics that we have seen before such as if statements, void functions and non-void functions, as required. we will be automatically grading your following homeworks as GradeChecker does, so it is very important to satisfy the exact same output with the sample runs. You can utilize GradeChecker (http://sky.sabanciuniv.edu:8080/GradeChecker/) to check whether your code is working in the expected way. To be able to use GradeCheker, you should upload all of your files used in the homework (your_main_cpp file, strutils.h and strutils.cpp). Additionally, you should submit all of your files to SuCourse (your main cpp file, strutils.h and strutils.cpp) **without zipping them**. **Name of your main cpp file should be in the expected format** ("sucourseusername_lastname_name_hwnumber.cpp", check the submission procedure of the homework).

Description

In this homework, you will write a C++ program that works as a simple calculator that performs only addition, subtraction, multiplication and division operations with many operands. Your program will ask the user to enter the expression that they want to be calculated. After receiving the input, the program will perform the necessary input checks. If there is an error in the expression, an appropriate message will be displayed and the user will be asked to enter another expression. If there are no errors in the input expression, then the calculated result will be displayed and the program will ask for another expression until the user enters a termination command. Termination commands are "QUIT", "quit", "EXIT", "exit", "END" or "end".

After a termination command, your program will close. You may (and you should) check the "Sample Runs" below.

VERY IMPORTANT!

Your programs will be compiled, executed and evaluated automatically; therefore you should definitely follow the rules for prompts, inputs and outputs. See **Sample Runs** section for some examples.

- **Order of inputs and outputs** must be in the abovementioned format.
- **Prompts before inputs and outputs** must be **exactly the same** with examples.

Following these rules is crucial for grading, otherwise our software will not be able to process your outputs and you will lose some grades in the best scenario.

IMPORTANT!

If your code does not compile, you will get **zero**. Please be careful about this and double check your code before submission.

Inputs

The input string is an expression that contains the operator symbol and/or the operands. The operators can be one of the following:

Operator	Arithmetic or Math Library	Example
+	addition	43+5-3
-	subtraction	223-8+56
*	multiplication	45*1/9
/	division	56/9*100

Simplification1: To simplify the handling of precedence rules, only addition and subtraction can be used together in the same expression; only multiplication and division together can be used in the same expression. That is, multiplication and division will **NOT** exist in an expression that has addition or subtraction and vice versa.

Simplification2: You may assume that there is no sign character in the expression. Whenever you see a "+" or "-"; then it means that it is an addition or subtraction operator.

You may also assume that all operands will be entered as non-negative integers and your program should not do input check for negative or non-integer operands. However, the result of the expression can still be a negative or non-integer value.

There could be as many operands as user enters. See sample runs for some examples.

Input Check

The expression string can only have digits and the operators given in the above table. If the user enters an expression string with an invalid character in it, then you should display an error message saying "**Invalid entry"**

If a valid operator is found in the expression string, there should be left and right operands for each operator; otherwise, you should display an error message saying "**Invalid entry**". However, if the expression only contains a non-negative integer like "110", then it is a valid entry and the result of this expression is itself. The format of the operands are not important, you MAY ASSUME they are non-negative integers.

Your program should not exit when an invalid expression is entered, it should just ignore that invalid expression after giving the error message and continue with the next prompt "**Please enter an expression to calculate:** "
See sample runs for some examples.

You MAY ASSUME that there will be no spaces in the string input of the expression.

Processing, Program Flow and Output

Your program starts with a prompt "**Please enter an expression to calculate:** " and expects one input string. After that input string is calculated and the result is displayed to the user, your program displays the same prompt "**Please enter an expression to calculate:** " again and again until the user enters one of the following strings to tell the program to exit:

- "QUIT" or "quit"
- "EXIT" or "exit"
- "END" or "end"

The result of the operation should be displayed on the next line as output. There is one exception for the processing: if you try to divide any number by 0, then you should prompt **"You cannot divide by 0"**

Please see sample runs for examples.

Parsing the expression operation to extract the operator and operands require using some of the string member functions (like **`length, find, rfind, substr, at,`** etc.) that we covered in class. You can use **atof** function for converting a string operand to a double value; **atoi** function for converting a string operand to an integer value. You may utilize one of these functions (or both depending on your algorithm). Note that these functions are given in **strutils** class, you should use these files in your project.

No abrupt program termination please!

You may want to stop the execution of the program at a specific place in the program. Although there are ways of doing this in C++, it is not a good programming practice to abruptly stop the execution in the middle of the program. Therefore, your program flow should continue until the end of the main function and finish there.

Sample Runs

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You should follow the input order in these examples and the prompts your program will display must be **exactly the same** as in the following examples.

Sample Run 1

```
Please enter the expression to calculate: 12+4
16

Please enter the expression to calculate: 17-5+200
212

Please enter the expression to calculate: 6/2*3
9

Please enter the expression to calculate: exit
GOOD BYE
```

Sample Run 2

```
Please enter the expression to calculate: -3+3
Invalid entry

Please enter the expression to calculate: 40+
Invalid entry

Please enter the expression to calculate: +
Invalid entry

Please enter the expression to calculate: ++
Invalid entry

Please enter the expression to calculate: 40++9
Invalid entry

Please enter the expression to calculate: 40+9-
Invalid entry

Please enter the expression to calculate: 40+9-3
46

Please enter the expression to calculate: QUIT
GOOD BYE
```

Sample Run 3

```
Please enter the expression to calculate: 3-15+4-200
-208

Please enter the expression to calculate: 2*2*2*2*2*2*2*2*2*2
1024

Please enter the expression to calculate: 1024/2/2/2/2*10
640

Please enter the expression to calculate: 44/5
8.8

Please enter the expression to calculate: END
GOOD BYE
```

Sample Run 4

```
Please enter the expression to calculate: 100
100

Please enter the expression to calculate: -100
Invalid entry

Please enter the expression to calculate: 100-0+55
155

Please enter the expression to calculate: 0
0

Please enter the expression to calculate: exit
GOOD BYE
```

Sample Run 5

```
Please enter the expression to calculate: 275*765
210375

Please enter the expression to calculate: 100/0
You cannot divide by 0

Please enter the expression to calculate: 234*4*56/0*4*56
You cannot divide by 0

Please enter the expression to calculate: 100+1A12
Invalid entry

Please enter the expression to calculate: END
GOOD BYE
```

General Rules and Guidelines about Homeworks

The following rules and guidelines will be applicable to all homeworks, unless otherwise noted.

How to get help?

You may ask questions to TAs (Teaching Assistants) of CS201. Office hours of TAs are at the class website. Recitations will partially be dedicated to clarify the issues related to homework, so it is to your benefit to attend recitations.

What and Where to Submit

Please see the detailed instructions below/in the next page. The submission steps will get natural/easy for later homeworks.

Grading and Objections

Careful about the full-automatic grading: Your programs will be graded using an automated system. Therefore you should follow the guidelines about input and output order; moreover you should also

use same prompts as given in the Sample Runs. Otherwise automated grading process will fail for your homework, and you may get a zero, or in the best scenario you will lose points.

Grading:

- Late penalty is 10% off of the full grade and only one late day is allowed.
- **Having a correct program is necessary, but not sufficient to get the full grade. Comments, indentation, meaningful and understandable identifier names, informative introduction and prompts, and especially proper use of required functions, unnecessarily long program (which is bad) and unnecessary code duplications (which is also bad) will also affect your grade.**
- Please submit your own work only (even if it is not working). It is really easy to find out "similar" programs!
- For detailed rules and course policy on plagiarism, please check out http://myweb.sabanciuniv.edu/gulsend/su_current_courses/cs-201-spring-2008/plagiarism/ and keep in mind that

Plagiarism will not be tolerated!

Grade announcements: Grades will be posted in SUCourse, and you will get an Announcement at the same time. You will find the grading policy and test cases in that announcement.

Grade objections: It is your right to object to your grade if you think there is a problem, but before making an objection please try the steps below and if you still think there is a problem, contact the TA that graded your homework from the email address provided in the announcement.

- Check the comment section in the homework tab to see the problem with your homework.
- Download the files you submitted to SUCourse and try to compile it.
- Check the test cases in the announcement and try them with your code.
- Compare your results with the given results in the announcement.

What and where to submit (IMPORTANT)

Submissions guidelines are below. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade.

Add your name to the program: It is a good practice to write your name and last name somewhere in the beginning program (as a comment line of course). Do not use Turkish characters anywhere in your program (not even in the comments).

Name your submission file:

- Use only English alphabet lowercase letters, digits or underscore in the file names. Do not use blank, Turkish characters or any other special symbols or characters.
- Name your cpp file that contains your program as follows.

"**sucourseusername_lastname_name_hwnumber.cpp**"

- Your SUCourse user name is actually your SUNet user name which is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse user name is cago, name is Çağlayan, and last name is Özbugsızkodyazaroğlu, then the file name must be:

  cago_ozbugsizkodyazaroglu_caglayan_hw4.cpp

- Do not add any other character or phrase to the file name.
- Make sure that this file is the latest version of your homework program.
- You have to submit all files that are used in your homework (ago_ozbugsizkodyazaroglu_caglayan_hw4.cpp, strutils.h, strutils.cpp) **without zipping them**.

Submission:

- Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.).
  1) Click on "Assignments" at CS201 SUCourse (not the CS201 web site).
  2) Click Homework 4 in the assignments list.
  3) Click on "Add Attachments" button.
  4) Click on "Browse" button and select all file used in homework.
  5) Now, you have to see your file(s) in the "Items to attach" list.
  6) Click on "Continue" button.
  7) Click on "Submit" button. We cannot see your homework if you do not perform this step even if you upload your file.

Resubmission:

- After submission, you will be able to take your homework back and resubmit. In order to resubmit, follow the following steps.
  1) Click on "Assignments" at CS201 SUCourse.
  2) Click Homework 4 in the assignments list.
  3) Click on "Re-submit" button.
  4) Click on "Add/remove Attachments" button
  5) Remove the existing files by clicking on "remove" link. This step is very important. If you do not delete the old files, we receive both files and the old one may be graded.
  6) Click on "Browse" button and select the new files that you want to resubmit.
  7) Now, you have to see your new files file in the "Items to attach" list.
  8) Click on "Continue" button.
  9) Click on "Submit" button. We cannot see your homework if you do not perform this step even if you upload your file.

Successful submission is one of the requirements of the homework. If, for some reason, you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!
Ece Egemen, İnanç Arın and Gülşen Demiröz