CS201 – Fall 2022-2023
Homework 2
– CS201 Course – Math Expression Checker–
Due November 9th, Wednesday, 23:55 (Sharp Deadline)

Introduction

The aim of this homework is to practice on functions, strings and nested if-else statements. Your aim will be to create a basic **math expression checker**. Imagine that the user is preparing questions to put in an elementary math book, so the program is checking their correctness.

Description

The program will check for the correctness of mathematical expressions entered by the user. The math expressions will consist of basic addition or subtraction only. The question itself as well as the answer will be given by the user as input, as in:

5 - 7 = -2

where 5 - 7 will be referred to as the left-hand side (lhs) and the right-hand-side will be referred to as the right-hand side (rhs).

Your program will get and process a total of **4** math expressions from the user. In addition, there will be **1** joker option, for a question that is answered incorrectly. The user may earn a maximum of 100 points and a minimum of 0.

Once your program gets a question and its answer as the input, it will parse the value obtained from the user, do the respective calculation and check:

a) if the syntax is correct and

b) if the given answer is correct.

If the math syntax and the answer is correct, then the user will gain 25 points.

If the question syntax is wrong (e.g. + 5 7 = -2) the user will lose 10 points.

If the syntax is correct but the answer is wrong, (s)he will receive a message and a chance to re-enter the expression.

Users will have **1** joker option. For the first wrong math expression, the user will be asked if s/he wants to use their joker option and enter the expression again. If the

user's answer is case-insensitive "yes", the user gets a second chance to enter the previous wrong expression one last time. If the new expression;

- is not the same as the previous given question,
- is answered wrong again,
- is with wrong syntax,

the user loses 10 points.

The user has only one try for the joker case, but does not have to use the joker option.

The format of the inputs and how to process them are described below in detail.

You **have to** write user-defined functions to implement your program, you can **not** use only the *main* function. The user-defined functions that you are expected to implement are also described below in detail.

Your homework will be automatically graded using GradeChecker, so it is very important to satisfy the exact same output given in the sample runs. You can utilize GradeChecker (http://learnt.sabanciuniv.edu/GradeChecker/) to check whether your implementation is working in the expected way. To be able to use GradeChecker, you should upload all of your files used in the homework. Additionally, you should submit all of your files to SUCourse **without zipping** them. **Just a reminder, you will see a character ¶ which refers to a newline in your expected output.**

The name of your main source (cpp) file should be in the expected format: "SUCourseUsername_HWnumber.cpp" (all lowercase letters). Please check the submission procedures of the homework, which are listed at the end of this document.

Inputs, Flow of the Program, Functions and Outputs

The program to be implemented has **three (3)** essential sections, and you have to implement **each section in a different user-defined function**. Here are the details for these three (3) sections:

- ***Input Check*:** You should take math expression input as a parameter and check the correctness of the input syntax according to the instructions below. If an expression has an incorrect math syntax, then your program should display a warning message and the user will lose points.

A correct syntax is a *simplified arithmetic syntax* like the following:

`(sign)number1 operator (sign)number2 = (sign)number3`

- An operation can only have one **`operator`**, and this operator can only be addition (+) or subtraction (-). Your program should not accept any other operators as correct, and there has to be exactly one operator.
- The answer should also be given within this input, meaning that there has to be an equality sign ("**=**") in the input. This equality sign should be in the correct place; in other words, there should be **only** a number after it and there should be an expression before it.
- The signs are optional, they may or may not exist. That's why it is written as **`(sign)`** in the format given above. They can only be the plus (+) and minus (-) signs in a math expression input. There can be **at most** 1 (one) sign before a number.
- Your program should also check the validity of the operands of the question (i.e. numeric values of the expression) and the answer as well. Each operand and the result (i.e. **`number1`**, **`number2`**, and **`number3`**) must be integers, thus they should only consist of digits.
- Math expression input can be like ***`5-2=3`*** or ***`5000-25=4075`***, you should not make assumptions on the size of the operands (they can have as many digits as any integer).

In summary, the expression can only consist of digits, and "+", "-" , "=" symbols. Also, they should all be in the correct place. *However, you should assume that there will be* ***no spaces within the input****.*

- ***Question Parse*****:** You should take the math expression input as a parameter and should find the parts of the expression based on the following directions. Note that the math expression input by the user will be of type `string`. Your program should then determine (i) the **`operator`**, (ii) the answer part of the input (**`number3`**), (iii) the left side of the operand (**`number1`**), and (iv) the right-hand side of the operand (**`number2`**). You may use ***`string member functions`*** for these purposes.
  Your program should also convert the numbers into **`int`** type, and you can use the ***`atoi`*** function from the ***`strutils`*** library for this purpose.

- ***Check Result*****:** You should take rhs, lhs, operator, and answer variables which you found in the previous section as parameters. Your program should evaluate the arithmetic expression (either addition or subtraction) given

within a valid math expression input, find the result of the respective calculation and check the answer given by the user. Users will gain 0 or more points as described in the third paragraph in the Description part. However, your program should also check the final point and set it to 0 if the total points is negative.

You **must** implement these 3 essential sections described above as **user-defined functions**! You can also implement more user-defined functions if you need them. In other words, you may prefer to use a single user-defined function to handle each task, or you may also apply decomposition and implement multiple user-defined functions to handle each one of these essential sections.

In short, **your program should have at least 4 functions**:

1. Main function,
2. A function to check the input format,
3. A function to parse the math expression inputs, and
4. A function to calculate the answer and check its correctness.

Additionally, you are the one who will decide how to integrate these functions and trace the points earned by the user to complete your solution.

Please refer to the "Sample Runs" section for some examples and further details.

IMPORTANT!

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

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 mentioned format.

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

Sample Runs

Below, we provide some sample runs of the program you will develop. The *italic* and **bold** phrases are inputs taken from the user. You have to display the required information in the same order and with the same words and characters as below.

Sample Run 1

```
Please enter question #1 and its answer: 15+5=20
Correct answer! You got 25 points for this math expression.

Please enter question #2 and its answer: 198--3=201
Correct answer! You got 25 points for this math expression.

Please enter question #3 and its answer: 30-53=-22
Wrong!
Would you like to use your joker option to correct the answer? YES
Please enter the expression again: 30-53=-23
Correct answer! You got 25 points for this joker option.

Please enter question #4 and its answer: 0-0=0
Correct answer! You got 25 points for this math expression.

End of the program. You got 100 points in total.
```

Sample Run 2

```
Please enter question #1 and its answer: 48+12=60

Correct expression! You got 25 points for this math expression.

Please enter question #2 and its answer: 146-89=48
Wrong!
Would you like to use your joker option to correct the answer? No
Sorry! The answer should have been: 57. You got 0 points for this math
expression.

Please enter question #3 and its answer: 12-5=-13
Wrong!
Would you like to use your joker option to correct the answer? no
Sorry! The answer should have been: 7. You got 0 points for this math
expression.

Please enter question #4 and its answer: 56+28=84
Correct answer! You got 25 points for this math expression.

End of the program. You got 50 points in total.
```

Sample Run 3

Please enter question #1 and its answer: ***145*1=145***
Wrong input format! You got -10 penalty points for this math expression.

Please enter question #2 and its answer: ***145-1=144***
Correct answer! You got 25 points for this math expression.

Please enter question #3 and its answer: ***-45+-36=-24***
Wrong! Would you like to use your joker option to correct the answer? ***Yes***
Please enter the expression again: ***-45+-36=-81***
Correct answer! You got 25 points for this joker option.

Please enter question #4 and its answer: ***67--3=64***
Sorry! The answer should have been: 70. You got 0 points for this math expression.

End of the program. You got 40 points in total.

Sample Run 4

Please enter question #1 and its answer: ***5%2=1***
Wrong input format! You got -10 penalty points for this math expression.

Please enter question #2 and its answer: ***12-5=-13***
Wrong! Would you like to use your joker option to correct the answer? ***Yes***
Please enter the expression again: ***5=2+3***
Wrong input format! You got -10 penalty points for this joker option.

Please enter question #3 and its answer: ***98--2=102***
Sorry! The answer should have been: 100. You got 0 points for this math expression.

Please enter question #4 and its answer: ***10-1264=-523***
Sorry! The answer should have been: -1254. You got 0 points for this math expression.

End of the program. You got 0 points in total.

Sample Run 5

Please enter question #1 and its answer: ***45-39=7***
Wrong! Would you like to use your joker option to correct the answer?
No
Sorry! The answer should have been: 6. You got 0 points for this math expression.

Please enter question #2 and its answer: ***29-8=22***
Wrong! Would you like to use your joker option to correct the answer?
YeS
Please enter the expression again: ***29*7=22***
Wrong input format! You got -10 penalty points for this joker option.

Please enter question #3 and its answer: ***29-8=21***
Correct answer! You got 25 points for this math expression.

Please enter question #4 and its answer: ***29-7=21***
Sorry! The answer should have been: 22. You got 0 points for this math expression.

End of the program. You got 15 points in total.

Sample Run 6

Please enter question #1 and its answer: ***48-13=35.0***
Wrong input format! You got -10 penalty points for this math expression.

Please enter question #2 and its answer: ***48-13=33***
Wrong! Would you like to use your joker option to correct the answer?
Yes
Please enter the expression again: ***48-13=36***
Sorry! The answer should have been: 35. You got -10 penalty points for this joker option.

Please enter question #3 and its answer: ***48-13=35***
Correct answer! You got 25 points for this math expression.

Please enter question #4 and its answer: ***15a+25=40***
Wrong input format! You got -10 penalty points for this math expression.

End of the program. You got 0 points in total.

General Rules and Guidelines about Homework

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

How to get help?

You can use GradeChecker (http://learnt.sabanciuniv.edu/GradeChecker/) to check your expected grade. Just a reminder, you will see a character ¶ which refers to a newline in your expected output.

You may ask questions to TAs (Teaching Assistants) or LAs (Learning Assistants) of CS201. Office hours of TAs/LAs can be found at SUCourse.

What and Where to Submit

You should prepare (or at least test) your program using MS Visual Studio 2012 C++ (Windows users) or using XCode (macOS users).

It'd be a good idea to write your name and last name in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts). If your name and last name is "Gülşen Demiröz", and if you want to write it as comment; then you must type it as follows:

// Gulsen Demiroz

Submission guidelines are below. Since the grading process will be automatic, students are expected to strictly follow these guidelines. If you do not follow these guidelines, your grade will be 0.

- Name your submission file as follows:
  - Use only English alphabet letters, digits, dot ('.') 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_hwnumber.cpp"
  - Your SUCourse user name is actually your SUNet username, which is used for checking sabanciuniv emails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name (**use only lowercase letters**) except dot ('.'). For example, if your SUCourse username is "**gulsend**", then the file name should be: **gulsend_hw2.*cpp*** (please only use lowercase letters).
  - Do not add any other character or phrase to the file name.
- Please make sure that this file is the latest version of your homework program.
- Submit your work **through SUCourse only**! You can use GradeChecker only to see if your program can produce the correct outputs both in the correct order and in the correct format. It will not be considered as the official submission. You must submit your work to SUCourse. You will receive no credits if you submit by any other means (email, paper, etc.).
- If you want to resubmit your work, you should first remove the existing file(s). This step is very important as if you don't delete the old files, we receive both files and the old one may be graded.

Grading, Review and Objections

Be careful about the automatic grading: Your programs will be graded using an automated system. Therefore, you should follow the guidelines on the input and output order. Moreover, you should also use the same text as given in the "Sample Runs" section. Otherwise, the automated grading process will fail for your homework, and you may get a zero, or in the best scenario, you will lose points.

Grading:

- There is NO late submission. You need to submit your homework before the deadline. Please be careful that SUCourse time and your computer time may have 1-2 minute differences. You need to take this time difference into consideration.
- 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.
- If your code does not work because of a syntax error, then we cannot grade it; and thus, your grade will be 0.
- Please submit your **own** work only. It is really easy to find "similar" programs!
- Plagiarism will not be tolerated. Please check our plagiarism policy given in the Syllabus.

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 comment section of your announced homework grade or attend the specified objection hour in your grade announcement.

- Check the comment section in the homework tab to see the problem with your homework.
- Download the file 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.

Good Luck!
E. Beyza Çandır & Serhan Yılmaz & CS201 Instructors