HW3: Calculator

CSE1102 Spring 2015
Jeffrey A. Meunier
University of Connecticut

1. Introduction

This assignment will have you write a program that works like a calculator. The user will be able to enter numbers and select the desired arithmetic operations. The program will repeat until the user chooses to exit the program.

Be aware that this is is a text-mode program that *works* like a calculator, but it is not a graphical program that *looks* like a calculator.

You will develop this program in incremental stages, making sure that it works correctly after each stage. If it does not work, the changes you made from the previous stage will be small enough that you can easily find and correct the error. This is how most programmers write programs: they don't sit down at the computer and type a complete program start to finish. It will have too many errors, and the errors will be too difficult to find and fix.

2. Value

This program is worth a maximum of 20 points. See the grading rubric at the end for a breakdown of the values of different parts of this project.

3. Due Date

This project is due by midnight at the end of the day on **Sunday, February 15, 2015**. A penalty of 20% per day will be deducted from your grade, starting at 12:00:01am. See this link for additional information.

4. Objectives

In the previous assignment I gave you a working program that you had to modify. In this assignment you will write a complete program yourself.

This is a console-based program, meaning there will not be graphics like there were in the previous assignment. You will need to use a **Scanner** to get input from the user, and use **System.out** to display information to the user. We'll come back to graphics soon.

You will use a lot of floating point (**double**) numbers and variables.

You will use the **if** statement and a **while** loop. You will need to read about the **if** statement on your own. It's not hard to use.

There will not be much in the way of design that you have to do. Mostly you will be constructing Java from my directions, and learning more Java syntax.

5. Background

Recall how a calculator works: there is a number that represents the value computed so far. This number starts at 0 and is always displayed on the calculator's screen. Now, let's say that the user enters a number, then the user selects an operation to perform (presses an arithmetic operator button), then enters another number. After the number is entered, the display is updated to indicate the new value. On a real calculator the user normally presses the *equals* button to indicate that he or she is finished entering the number. This program will use the enter key instead of the equal key.

What I find most interesting about this project is that at at fundamental level, any interpreted programming language works the same way as this calculator. In fact, *any computer* works the same way as this calculator. So you're not just writing a calculator program, you're writing a program that is easily capable of being turned into a universal computing machine.

6. Assignment

Read through all of the subsections in this section before you start to make sure you understand how you must proceed with the assignment.

I have you develop this program program in 11 steps.

1. Create a program
2. Create a loop
3. Get the user's choice
4. Add a menu
5. Create accumulator & input variables
6. Handle the user's choice
7. Do addition
8. Do subtraction
9. Multiplication and division
10. Square roots.
11. Clear the accumulator

6.1. Create a new project

Create a new Java project called **CSE1102-03-Calculator** with a standard main method in

a class called **Calculator**. Place comments at the top of the file indicating:

- The name of the project.
- The course number and name.
- Your name.
- Your TA's name and your section number.
- The date you created the project.

6.2. Create a loop

Summary

Most of this program will be inside a **while** loop in the **main** method. This section will just have you type the loop and be sure it works.

Do this

1. Inside the **main** method create a **Scanner** instance in the normal way.
2. After you create the scanner, create a **boolean** variable called **contin** and set it to **true**. This will be used as the *loop control variable*.
3. Next create a **while** loop. The condition of this while loop is simply the **contin** variable (not ***contin == something***, or ***contin != something***, just ***contin***, because it's already a boolean expression). The body of the **while** loop will contain many statements, so type both the opening and the closing braces now. We will start filling in the body of the while loop in the next section.
4. Make sure that you type any remaining closing braces to finish off the program. You now have a complete program.
5. Make sure there are no errors indicated in red. If there are, fix them.
6. Run the program. It should do nothing, and it should do it for a long time. Type **Ctrl-C** to interrupt the program, or press the red square in the console window of Eclipse.
7. A note about formatting: Any time you type an opening brace you should hit Enter and then indent the next line by two additional spaces. Any time you type a closing brace, you should un-indent it by two spaces, so that it lines up vertically with the **while** statement.

```
while(___)
{
  this is correct
}

while(___) {
  this is also correct
}

while(___)
{
this is not correct
}
```

6.3. Get the user's choice

Summary

The program will ask the user to enter an option, which will be an integer. If the option is 0, then the program will exit, otherwise the program will ask the user for another option.

Do this

1. Inside the body of the while loop, create a new integer variable called **opt**. You do not need to assign a value to it yet.
2. For the next statement, display a message to the user that reads `"What is your option? "`. Note that there is a space following the question mark. Also be sure to leave the cursor on the same line.
3. The next statement should read an integer from the keyboard and store it in the **opt** variable.
4. Now check the value of the option variable with an **if** statement (this is all still inside the **while** loop). If the option is equal to 0, then set the value of the **contin** variable to **false**.
5. Otherwise (as the **else** part of the **if** statement) display the message `"Illegal option: "` followed by the value of the **opt** variable.
6. If you run the program now it should look like this:

```
What is your option? 1
Illegal operation: 1
What is your option? 2
Illegal operation: 2
What is your option? 3
Illegal operation: 3
What is your option? 0
```

Note

Don't use any **Scanner** method other than **nextInt** or **nextDouble**.

6.4. Add a menu

Summary

The calculator will ultimately allow addition, subtraction, multiplication, division, and square root. Thus, we should display these options to the user, so that the user knows what options are allowed. This will be done using a series of **println** statements.

Do this

Add a series of **println** statements that displays the following lines below. Simply use one **println** statement for each line. These statements should come right before the statement that displays "What is your choice?"

```
Please choose one of the following options:
1) Addition
2) Subtraction
3) Multiplication
4) Division
5) Square root
6) Clear
0) Exit
What is your option?
```

If you compile and run your program, it should not operate any differently (everything is still an illegal operation). However, there will be a lot more information displayed on the screen.

6.5. Create accumulator and input variables

Summary

The *accumulator* is the fancy name for the variable that holds the intermediate result of a computation. Remember how a calculator works: if you need to calculate 3x2+1, first you clear the calculator so that the accumulator contains 0, then you type 3 and hit x (multiply), so the accumulator contains 3, then you type 2 and hit +, so the accumulator now contains 6, then you type 1 and hit =, so the accumulator contains 7.

Note that the calculator must hold two values simultaneously: the accumulator and the value that the user is entering.

Do this

1. Before the **while** loop inside the main method, declare a new **double** variable called **accumulator**, and define it to be 0.0. This means that the calculator's initial value will be 0.0. Note that this statement can go anywhere inside the main method *before* the **while** loop.
2. On the next line, still outside the while loop, declare a **double** variable called **input**. This variable will hold the number that the user enters that will be added to (or subtracted from, or multiplied by, etc.) the value in the accumulator variable.
3. Inside the while loop display the message `"Accumulator = "` followed by the value of the accumulator variable. This should come before the menu.
4. Compile your program to be sure it does not have any errors. If you run the program, it does not appear to do anything new. That's because even though we have two new variables, we haven't got any new values from the user. The program should still exit when you enter 0.
5. The program output should look like this:

```
Accumulator = 0.0
```

```
Please choose one of the following options:
1) Addition
2) Subtraction
3) Multiplication
4) Division
5) Square root
6) Clear
0) Exit
What is your option?
```

6.6. Handle the user's option

Summary

We let the user enter a value as his or her option, so now we must have the program do something based on that option. Each of the options will cause the program to do something different.

Do this

Modify your **if/else** statement to be an **if/else if** ladder. Each rung in this ladder should check the option variable for one of the numbers 0 to 6. The ladder should look like this:

```
if(opt == 0)
{
 contin = false;
}
else if(opt == 1)   << notice the space between else and if
{
 // do addition
}
```

etc.

Don't forget the curly braces around the bodies of the **else if** clauses, and put simple comments in them like I show here (option 2 is for subtraction, etc.). In the subsequent sections you'll go back and replace the comments with real statements. At the bottom of the ladder is the final else clause that was part of the original **if/else** statement.

If you have been putting the opening brace at the end of the previous line instead of on a line by itself, then continue to do that here.

After completing this section your program should compile and run just as it did before. It still doesn't do anything interesting.

6.7. Do addition

Summary

Now you will replace the "dummy" code inside the addition case of the **if/else** ladder with

code that actually does addition. Since the only number we have right now is the accumulator, you will need to ask the user for a number. This number will be added to the value of the accumulator and then stored back into the accumulator.

Do this

1. Find the place in your **if/else** ladder where choice 1 is handled (this is for addition).
2. You can leave the comment in place.
3. Display a message to the user `"Enter a number: "`
4. Get a number from the keyboard and store it in the **input** variable. Use the **.nextDouble** method found in the scanner.
5. Add the values of the **input** and **accumulator** variables together and store the result back into the **accumulator** variable.
6. Compile and test your program. Now option 1 does something useful. What do you expect should happen now when you run your program? Make sure it works.

6.8. Do subtraction

Write case 2 in the **if/else** ladder so that it is similar to case 1: get a number from the user, subtract it from the accumulator and store it back into the accumulator. Compile & test.

Make sure the answer to 10 – 2 is 8 and not –8.

6.9. Multiplication and division

Add cases 3 and 4 to handle multiplication and division. They work just like cases 1 and 2 for addition and subtraction. Compile & test. Notice what happens if you divide by zero.

6.10. Square roots

Write case 5 so that it does square roots. Do a bit of research using your book or the interwebs to determine how to do square roots in Java (hint: it's called **sqrt**). This case does not need to get input from the user. It does the square root on the value already in the accumulator. Compile & test, but make sure you start with a positive number in the accumulator.

Add an **if/else** statement inside this part of the program that checks to see if the accumulator is negative. If it is not negative, do the square root normally. If it is negative, display the message "Not taking square root of negative number" and leave the accumulator the way it is.

6.11. Clear the accumulator

For case 6 just store the value 0.0 into the accumulator variable. This works just like the

Clear key on a calculator. Compile & test.

Congratulations! Your program is now complete.

7. Development analysis

Notice how I had you develop this program: first we started with the main method, then we created an empty loop, then filled in the loop with a menu and an if/else ladder, then finally filled in the rungs of the ladder one by one. We developed this program iteratively (doing edit-test, edit-test), and working from the outside in. This breaks a large program into manageable pieces. Sometimes this outside-in method is called top-down programming, but that only makes sense when you draw the program modules as boxes on paper or on a white board, and show their relationship in a hierarchy.

8. A sample run

Here is a complete sample run from a working program. You can use it to verify your program. I have omitted the menu information so that you can see the values more easily.

```
Accumulator = 0.0
...
What is your option? 1
10

accumulator = 10.0
...
What is your option? 3
8

accumulator = 80.0
...
What is your option? 5

accumulator = 8.94427 (give or take a few decimal places)
...
What is your option? 2
10

accumulator = -1.05572
...
What is your option? 6

accumulator = 0
...
What is your option? 0
```

9. Report

Create a Microsoft Word or OpenOffice Word file (or some other format that your TA agrees on -- ask him or her if you are not sure). Save the file with a .doc or .docx format.

At the beginning of the document include this information:

Calculator
CSE1102 Project 3, Spring 2015
Your name goes here
The current date goes here
TA: *Your TA's name goes here*
Section: *Your section number goes here*
Instructor: Jeffrey A. Meunier

Be sure to replace the parts that are underlined above.

Now create the following three sections in your document.

1. **Introduction**
   In this section copy & paste the text from the introduction section of this assignment. (It's not plagiarism if you have permission to copy something. I give you permission.)
2. **Output**
   Run your program and copy & paste the output from the command window here. You do not need to write anything. Show that all the operations work, and use numbers different from what I used.
3. **Source code**
   Copy & paste the contents of your Java file here. You do not need to write anything.

10. Submission

Submit the following things things on HuskyCT:

1. Your exported Eclipse project.
2. The report document.

If for some reason you are not able to submit your files on HuskyCT, email your TA before the deadline. Attach your files to the email.

11. Grading Rubric

Your TA will grade your assignment based on these criteria:

- (2 points) The comment block at the beginning of the program file is correct.
- (6 points) The program displays the correct answers.
- (4 points) The program uses the correct calculations to generate the answers.

- (4 points) The program is formatted neatly.
- (4 points) The document contains all the correct information and is formatted neatly.

12. Getting help

Start your project early, because you will probably not be able to get help in the last few hours before the project is due.

- If you need help, post a message on Piazza immediately.
- In second order of priority, send e-mail to your TA.
- Go to the office hours for (preferably) your TA *or any other TA*. I suggest you seeing your own TA first because your TA will know you better. However, don't let that stop you from seeing any TA for help.
- Send e-mail to Jeff.
- Go to Jeff's office hours.