ITEC 1630 Object-Oriented Programming
Sections M, N, and O, Winter 2007

Assignment 2

Due date:

For all sections M, N, and O of ITEC 1630, the assignment must be submitted by **Monday March 12 at 7pm**.

To submit:

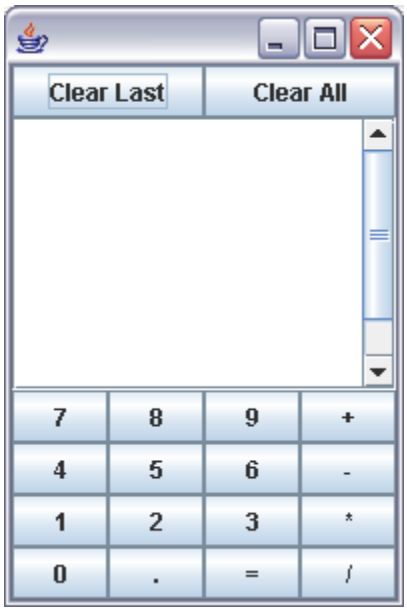
Prepare a single zip file containing all the source files (`.java` files, NOT `.class` files) and the HTML page for running the applet. Submit this file as an attachment to an email sent to: `itec1630@atkinson.yorku.ca`. Put in the subject line your section (M, N, or O), name, student number, and the words "Assignment 2". Please make sure that your section is entered correctly. For example, if your name is Jane Doe, and your section is M, your subject line will be "`M Jane Doe 999999999 Assignment 2`". Your section must be in capital case. Your submission will not reach your TA and will therefore NOT be marked if a wrong section code is used in the subject line. It is highly recommended that you CC yourself on this email. You can email the program any time before the deadline. A printout of the source files and HTML file with the following cover page:

```
/*****************************************************************

ITEC 1630 Assignment 2
Family name:
Given name:
Student number:
Section (M, N, or O):

Marking Template: (-10 marks if this template is not included)

Style (variable naming, indentation, & Layout)  _____/5
Comments                                        _____/5
Proper use of listeners
        (by inspection of source code)          _____/15
Code Compiles?                                  _____(yes/no)
Correct layout  (shape and fixed size)          _____/6
Proper error handling(3 errors)                 _____/15
Successful execution of test cases (18)         _____/54
Total                                           _____/100

According to this template the maximum mark you can get for code that
does not compile is 25/100.  If you do not provide a printout of your
code including this template, the maximum mark is 75/100.

*****************************************************************/
```

is to be put in the drop box on the 3rd floor of TEL building before the deadline.

Details

Write the Java applet which implements a four operation calculator, as shown below:



The calculator will work as follows:

1. The scrollable text area will show each digit as entered.
2. Decimal numbers are allowed, and you must ensure that only one decimal symbol is entered for each number (you decide what the behaviour of the calculator should be if the user breaks this rule, but it must be controlled by your program)
3. When an operator ('+', '-', '/', or '*') is clicked, the operator should be displayed after the last digit entered, and subsequent numbers should be displayed on a new line
4. After entering a sequence of *number operator number* the user can click:
    a. '=', in which case the calculator must display:
        i. the '=' symbol after the last digit of the second *number*
        ii. the result of the operation, on a new line
        iii. anything else entered after the '=' symbol is part of a new calculation, and must be displayed on a separate line

        For example, the user clicks "123.45+456.2=1". The screen should look like this:

        123.45+ ← entered by user
        456.2= ← entered by user
        579.65 ← calculated & displayed by your program
        1 ← beginning of a new calculation, entered by user

b. any new operator, in which case the calculator must assume an implicit '=' and:
    i. display an '=' sign after the last number
    ii. display the result of the previous operation on a new line, followed by the second operator
    iii. treat the result as the first number of a new calculation, and display the new operator right after it
    iv. allow the user to enter a new number

    For example, the user clicks "123.45+456.2/5=". The most reasonable interpretation is that the user wants the first two numbers added, and the sum divided by 5. The screen should look like this:

    123.45+ ← entered by user
    456.2= ← 456.2 entered by user, '=' added by the program
    579.65/ ← 579.65 calculated by program, '/' entered by user
    5= ← new calculation, entered by user
    115.93 ← calculated by your program

5. When the user enters two or more consecutive operators, write an error message in the text area stating that operator 'x' has been ignored. The message should be on a separate line, and everything else following should be on a new line.
6. When the user clicks "Clear Last" the calculator should delete the last character displayed on the screen from the screen. Further behaviour of your calculator should reflect the deletion (for example "12+ClearLast-5" should execute a subtraction, not an addition;). When clicked after '=', Clear Last should have no effect.
7. When the user clicks "Clear All" the calculator should erase everything in the text area and start again with a new calculation.

Other requirements:

1. The layout must match the image above
2. You must combine listeners where possible (hint: you don't need a separate listener for every button)
3. If you decide to issue error messages, they should be displayed in the text area
4. The applet size should be fixed
5. Your applet must protect against illegal operations (such as division by 0)

Please make sure you test at least the following scenarios, using both integer and decimal numbers:

1. Each individual single operation (4 tests)
2. Chained operations, both of the same kind (as in x+y+z) and different kinds (as in x+y-z/u) – 2 tests, plus 1 test with all 4 operators

3. Every error condition (successive operators, division by 0, multiple decimal points) – 3 tests
4. Behaviour of the clear buttons (8 tests), after
    a. digit entry
    b. fisrt operator entry
    c. second operator entry triggering a calculation
    d. ‘=’ symbol