ECE2036 **Fall Semester, 2012**

Project 1 – Simple Calculator

Assigned: Aug 24, 2012 *Due: Sept 4, 2012 11:59pm*

In this assignment, we will create a calculator that performs simple arithmetic operations on real numbers. The calculator must support addition, subtraction, multiplication and division.

There are three basic requirements for the calculator.

1. If the input line has two operands and one operator, the calculator should perform the specified operation and print the result. Further, the result should be saved for use in requirement 2 below. An example is:

```
12 + 23          (input by user)
    = 35         (output by the program)
```

2. If the input line starts with an operator followed by a single operand, the result from the prior operation is used as the left–side operand. An example is:

```
12.5 + 23.1      (input by user)
    = 35.6       (output by the program)
+ 55.1           (input by user)
    = 90.7       (output by the program)
```

3. Finally, a single operand with no operator simply outputs the value of the single operand and stores it as the prior result.

```
12.5                (input by user)
    = 12.5          (output by the program)
+ 55                (input by user)
    = 67.5          (output by the program)
```

Your program must be written in C++ and implement the functionality described above. To parse the input strings for your complex calculator, a parser is provided. The parser takes an input line as input, and breaks it into substrings separated by numeric operators (+, -, * and /). Documentation of how to call the parser is included in the provided source code. Thus the input:

```
2 + 3
```

would be parsed as two substrings, the first is `2` with a "+" delimiter, and the second is the substring `3` with a zero character delimeter.. The provided skeleton program `SimpleCalculator-Skeleton.cc` demonstrates how to call the parser and prints out the parsed strings as debugging information. You should remove the debug code before finishing your program. Finally, the code given in `string-parse.cc` has an additional useful function, `ToDouble` will convert the string represenation of a floating point number to type `double`.

A sample session is shown below:

```
1.123
 = 1.123
+4.234
 = 5.357
1 - 100
 = -99
154.234 * 200.2
 = 30877.6
/3.0
 = 10292.5
0
 = 0
+ 5.4
 = 5.4
/0.0
 = inf
0.0 / 0.0
 = nan
543 * 43
 = 23349
57.0
 = 57
123.123 + 456.456
 = 579.579
/10.0
 = 57.9579
```

Copying the Project Skeletons

1. Log into `jinx-login.cc` using `ssh` and your prism log-in name.
2. Copy the files from the ECE2036 user account using the following command:

   ```
   /usr/bin/rsync -avu /nethome/ECE2036/SimpleCalculator .
   ```

   Be sure to notice the period at the end of the above command.
3. Change your working directory to `SimpleCalculator`

   ```
   cd SimpleCalculator
   ```

4. Copy the provided `SimpleCalculator-skeleton.cc` to `SimpleCalculator.cc` as follows:

   ```
   cp SimpleCalculator-skeleton.cc SimpleCalculator.cc
   ```

5. Then edit `SimpleCalculator.cc` to implement the calculator.
6. Compile your code using `make` as follows:

   ```
   make
   ```

7. Once you have gotten the calculator program compiled and ready to test, you can just run it interactively:

   ```
   ./SimpleCalculator
   ```

   and type in any calculator commands for debugging. Or you can use the "canned" inputs in input.txt as follows:

   ```
   ./SimpleCalculator < input.txt
   ```

   The expected output matching "input.txt" is found in "output.txt".

Resources

1. `SimpleCalculator-skeleton.cc` is a starting point for your program.
2. `Makefile` is a file used by the `make` command to build the calculator program.
3. `string-parse.h` and `string-parse.cc` are provided for you and are used to "parse" the calculator input into substrings. There are comments in both files describing their usage, plus we will discuss these in class.
4. `input.txt` is a set of inputs that the TA will use to "test" your program.
5. `output.txt` is the matching set of outputs for `input.txt`.

Turning in your Project. The system administrator for the jinx cluster has created a script that you are to use to turn in your project. The script is called `riley-turnin` and is found in `/usr/local/bin`, which should be in the search path for everyone. From your **home directory** (not the MatrixCalculator subdirectory), enter:

`riley-turnin SimpleCalculator.`

This automatically copies everything in your `SimpleCalculator` directory to a place that I can access (and grade) it.