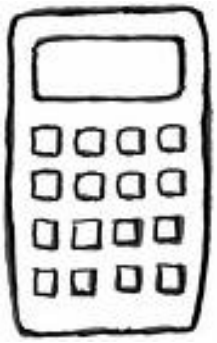
CAS CS 112 – Spring 2008, Assignment 3

due at 10:00 pm on Thursday, February 28



In this assignment you will write a programmable calculator. The calculator will have an assign function which takes a variable name and an assignment: either an integer or postfix expression. The calculator will also have an evaluate function which takes a postfix expression and converts it into an integer, and then returns that integer. Postfix expressions will consist of single character variables, arithmetic operations (+ , - , * , /), and integers between 0 and 9 inclusive. You can see example use of the calculator below.

```
Calculator c = new Calculator();
c.assign("x=5");
c.assign("y=x7+");
int first = c.evaluate("xy*");
c.assign("z=57*");
c.assign("x=x8+y *");
int sec = c.evaluate("xy+");
int third = c.evaluate("z");
//This code will set first to 60, then it will set sec to 168, and third to 35.
```

Linked list

(25 Points) You will create a linked list data structure to store variables. The linked list will consist of variable nodes; these will contain the variable name as a char and its value as an int. Students should create this structure in a separate class named **VList**.

Iterator

(10 points) **VList** should implement the **Iterable** interface. Feel free to use your Iterator when looking up the values of variables.

Stack

(25 Points) You will create a stack to evaluate postfix expressions. The stack will be completely dynamic, that is the underlying data structure will be a linked list. You should only need one stack to perform an evaluation. The stack should also be its own class named **CStack**.

Calculator

(30 Points) Finally, you will create a calculator class named **Calculator** which contains a constructor and two public methods **void assign(string assgnmnt)** and **int evaluate(string expr)** as described above.

Handling bad input

(10 points) If your calculator encounters a malformed postfix expression or an undefined variable, it must throw a **malformedPostfixException** or **undefinedVariableException**, respectively. We will show you how to define your own exceptions by extending the Exception class in lab.

Submissions

Please submit all files necessary for compilation, including the files named above and any additional files that you used. You do not need to submit a test file or a main method.