```
//Class: CSE 330

// Term: Spring 2014

// Instructor: George M. Georgiou

// Name: Seth Lemanek &

// Lab 6

// Title: Postfix.cpp & Stack.h


#include "stack.h"

#include <iostream>


// class postfixCalc

//  simulates the behavior of a reverse notation calculator



class postfixCalc {

 public:

    enum binaryOperator {plus, minus, multiply, divide};
    int currentMemory  () { return data.top(); }
    void pushOperand   (int value) { data.push (value); }
    void doOperator    (binaryOperator theOp);

 protected:

    Stack<int> data;
};

void postfixCalc::doOperator(binaryOperator theOp)
    // perform a binary opertion on stack values
{

    int right = data.top();
```

```
    data.pop();
    int left = data.top();
    data.pop();
    int result;

    switch(theOp) {  //do the operation

        case plus:
            result = left + right;
            break;

        case minus:
            result = left - right;
            break;

        case multiply:
            result = left * right;
            break;

        case divide:
            result = left / right;
            break;

        }
    data.push(result);//put result in stack for future use

}


int main()
{

    int intval;
    postfixCalc calculator;
    char e;
    cout << "Enter your postfix expression: ¥n";

    while(cin >> e)
    {
        switch(e) {

                case '0': case '1': case '2': case '3': case '4':
                case '5': case '6': case '7': case '8': case '9':
```

```
                cin.putback(e);
                cin >> intval;
                calculator.pushOperand(intval);
                break;

            case '+':
                calculator.doOperator(postfixCalc::plus);
                break;

            case '-':
                calculator.doOperator(postfixCalc::minus);
                break;

            case '*':
                calculator.doOperator(postfixCalc::multiply);
                break;

            case '/':
                calculator.doOperator(postfixCalc::divide);
                break;

            case 'p':

                cout << calculator.currentMemory() << '¥n'; // output current data

                break;

            case 'q':
                return 0; // quit calculator

            }
    }
}


//stack.h

#ifndef STACK_H
#define STACK_H
// Stack.h -- a stack implemented as an adapter (of vector or list or ...)
#include <list>
using namespace std;
```

```
//Use the following line for STL containers.
template <class T, template <class T, class = allocator<T> > class Container = list>
//template <class T, template <class T> class Container = list>
class Stack
{
public:
//We don't need a constructor or destructor because the Container has/should have one
//Stack(): container() { }
//~Stack() { ~container(); }
bool empty() const { return c.empty(); }
unsigned int size() const { return c.size(); }
void push(const T & x) { c.push_back(x); }
void pop() { c.pop_back(); }
T & top() { return c.back(); }
private:
Container<T> c;
};
#endif
```

```
Script started on Tue 13 May 2014 10:07:10 AM PDT
#]0;004470530@jb358-
15:/students/csci/004470530/cse330/lab06##[?1034h[004470530@jb358-15 lab06]$
./calc

Enter your postfix expression:

34+

Segmentation fault

#]0;004470530@jb358-15:/students/csci/004470530/cse330/lab06#[004470530@jb358-15
lab06]$ ./calc#######[Kcal##[K##[K##[K./calc

Enter your postfix expression:

34+# ## # 4 + p

7

3 2 + 6 * 8 2 + 5 * - p

-20

q

#]0;004470530@jb358-15:/students/csci/004470530/cse330/lab06#[004470530@jb358-15
lab06]$ exit

exit
```

Script done on Tue 13 May 2014 10:11:57 AM PDT