Assignment 5: Expressions and Queues

CS230
Fall 2000

Allen B. Downey
Computer Science Department

Due: Wednesday 26 October

The purpose of this assignment is to use a Stack and implement a Queue.

Postfix calculator

Write a program that prompts the user for a mathematical expression in postfix and that evaluates the expression and prints the result.

The following steps are my suggestion for a program development plan.

1. Write a program that prompts the user for input and prints the input string, over and over, until the user types "quit". Here's a starter:

```
public static void inputLoop () throws IOException {
    BufferedReader stdin =
        new BufferedReader (new InputStreamReader (System.in));

    while (true) {
        System.out.print ("=>");       // print a prompt
        String s = stdin.readLine();   // get input
        if (s == null) break;
        // check if s is "quit"
        // print s
    }
}
```

2. Identify helper methods you think will be useful, and write and debug them in isolation. Suggestions: `isOperator`, `isOperand`, `parseExpression`, `performOperation`.

3. We know we want to push `int` values onto the stack and pop them off, which means we will have to use a wrapper class. Make sure you know how to do that, and test those operations in isolation. Maybe make them helper methods.

4. Write a version of `evaluate` that only handles one kind of operator (like addition). Test it in isolation.

5. Connect your evaluator to your input/output loop.

6. Add the other operations.

7. Once you have code that works, you might want to evaluate the structural design. How should you divide the code into classes? What instance variables should the classes have? What parameters should be passed around?

8. In addition to making the design elegant, you should also make the code bulletproof, meaning that is does not cause an exception under any circumstances, even if the user types something weird.

Circular buffer

1. Write an implementation of a `Queue` using a circular buffer.
2. Again, make your code bulletproof, so that it is provably impossible for an exception to occur under any circumstances.