COLBY COLLEGE
Department of Mathematics
and Computer Science

CS231 **Allen B. Downey**
Fall 1998

Assignment 5: Stacks

Due: Friday 23 October
Reading: Standish Chapter 6.1 – 6.6.

Check balanced parentheses, braces, brackets

```
import java.io.*;

public class Balance {

  // parseLine traverses the given String, printing all the characters,
  // with a newline at the end of each line

  public static void parseLine (String line) {
    for (int i = 0; i<line.length(); i++) {
      System.out.print (line.charAt(i));
    }
    System.out.print ("\n");
  }

  // checkFile reads each line of the given file and passes it to parseLine

  public static void checkFile (String filename)
                   throws FileNotFoundException, IOException {

    // open the file and create a BufferedReader for it
    BufferedReader in =
      new BufferedReader (
new FileReader (filename));
    String s;

    // use the BufferedReader to get lines from the file
    while (true) {
      s = in.readLine();
      if (s == null) break;
      parseLine (s);
    }
```

```
  }

  public static void main (String[] args)
                     throws FileNotFoundException, IOException {
    checkFile ("Balance.java");
  }
}
```

1. Type in or cut and paste the above program, and run it. It should print out its own source code file.

2. Study the program, noting especially the use of the methods `readLine` and `charAt`. You don't have to worry much about the details of File I/O, but this will give you a sense of what it looks like (and why I have been postponing it).

3. Modify this program so that it prints each line backwards. You don't have to turn in this code, but doing it will help you understand the code you are starting with.

Make a plan

1. Write a paragraph or two explaining how you would use the Stack ADT to make sure that the parentheses, braces, and brackets in the program are balanced.

2. Once you have an algorithm, make a plan that details at least three intermediate stages between where the program is now and where it is going. Each stage should be a small modification of its predecessor, it should use a new feature or add a new function, and it should be testable.

Implement it!

1. After we have discussed the planning stage in class, go ahead and implement it.

Print backwards

1. Write a paragraph explaining how you would use the Stack ADT to write an iterative version of `printBackwards`.

2. Do it.