CS1083
Assignment #7

Daniyal Khan
3765942

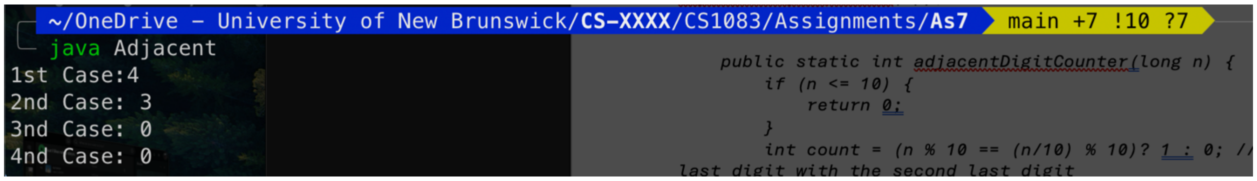
Adjacent.java:

```
public class Adjacent {
    public static int t = 0;
    public static void main (String[] args) {
        long n = 955133366;
        System.out.println("1st Case:" +
adjacentDigitCounter(n));
        long m = 112233;
        System.out.println("2nd Case: " +
adjacentDigitCounter(m));
        long b = 0;
        System.out.println("3nd Case: " +
adjacentDigitCounter(b));
        long v = 10;
        System.out.println("4nd Case: " +
adjacentDigitCounter(v));
    }

    public static int adjacentDigitCounter(long n) {
        if (n <= 10) {
            return 0;
        }
        int count = (n % 10 == (n/10) % 10)? 1 : 0; // comparing
last digit with the second last digit
        return count + adjacentDigitCounter(n/10); // chopping
off the last digit
    }

    /*
        if (n % 10 == (n/10)%10) {
            return 1 + adjacentDigitCounter(n/10);
        } else {
            return 0 + adjacentDigitCounter(n/10);
        }
     */
}
```

CaeserCipher.java:

```
import java.io.*;
import java.util.Scanner;

public class CaeserCipher {
    public static void main(String[] args) {
        try {
            File fileIn = new File("Msg.txt");
            Scanner scan = new Scanner(fileIn);
            scan.useDelimiter(",");
            PrintWriter pw = new PrintWriter("Encrypted.txt");

            while (scan.hasNextLine()) {
                int key = Integer.parseInt(scan.next().trim());
                String msg = scan.nextLine().trim();
                if (msg.startsWith(",")) {
                    msg = msg.substring(1).trim();
                }
                pw.println(encrypt(msg, key));
            }
            scan.close();
            pw.close();
        } catch (NumberFormatException nfe) {
            System.out.println(nfe.getMessage());
        }  catch (FileNotFoundException fnfe) {
            System.out.println(fnfe.getMessage());
        }
    }

    public static String encrypt(String msg, int key) {
        if(msg.length() == 0) {
            return "";
        }
        char currentChar = msg.charAt(0);
        if((key >= 0 && key <= 25) && (currentChar >= 'A' &&
currentChar <= 'Z')) {
            if ((currentChar+key) > 'Z'){ // wrap around if it's
greater than Z
                currentChar = (char)((msg.charAt(0) + key) -
26);
            } else {
                currentChar = (char)(msg.charAt(0) + key);
            }
        }
        return currentChar + encrypt(msg.substring(1), key);
    }

    /* Iterative
    public static String lencrpt(String msg, int key) {
        String array[] = new String[msg.length()];

        for (int i =0; i < msg.length(); i++) {
            char currentChar = msg.charAt(i);
            if ((key >= 0 && key <= 25) && (currentChar >= 'A'
&& currentChar <= 'Z')) {
                if ((currentChar + key) > 'Z') {
                    array[i] = "" + (char)((currentChar + key) -
26);
                } else {
                    array[i] = "" + (char)(currentChar + key);
                }
            } else {
                array[i] = currentChar + "";
            }
        }

        return String.join("", array);
    }
    */
}
```

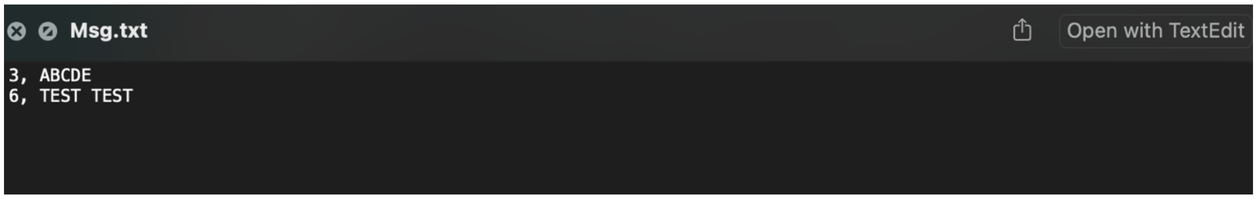
Output for Adjacent.java:



```
~/OneDrive - University of New Brunswick/CS-XXXX/CS1083/Assignments/As7  main +7 !10 ?7
 java Adjacent
1st Case:4
2nd Case: 3
3nd Case: 0
4nd Case: 0
```

```
    public static int adjacentDigitCounter(long n) {
        if (n <= 10) {
            return 0;
        }
        int count = (n % 10 == (n/10) % 10)? 1 : 0; //
last digit with the second last digit
```

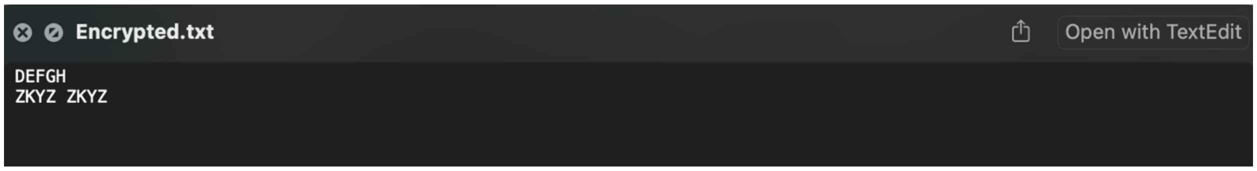
Output for CaeserCipher.java:



Msg.txt

Open with TextEdit

```
3, ABCDE
6, TEST TEST
```



Encrypted.txt

Open with TextEdit

```
DEFGH
ZKYZ ZKYZ
```