Sheetal Parikh
Programming Assignment 3 - Summary

(a) Give an efficient algorithm that takes strings s, x, and y and decides if s is an interleaving of x and y. Derive the computational complexity of your algorithm.

In order to ensure a better time complexity and avoid having to continuously compare all the characters of all three strings, we can use dynamic programming to solve the problem. We will use a 2D boolean array to keep track of the parts of the strings we have traversed and move from the end of the strings to the start of the strings. Through this method, we will be able to recursively compare subsequent characters of the strings. As we will be traversing the lengths of string s and string y and comparing them to string s, I believe the complexity of the algorithm will be $O(nm)$. The $n * m$ represents the length of the string x,(which will be represented by n), and length of string y (which will be represented by m).

Pseudocode:

```
Boolean Function CheckifInterweaving(String x, String y, String s)
    If x = null or y = null or s = null
        return false
    If length of x and length of y and length of s = 0
        return true
    Boolean 2D Array $A[][] = [length of x][length of y]$
    $A[0][0] = true$
    for i to length of x when y is empty
        if current character at x matches s
        $A[i][0] = true$
    for j to length of y when x is empty
        if current character at y matches s
        $A[0][i] = true$
    for i to length of x when y is not empty
        for j to length of y when x is not empty
            if current character at x matches s
                $A[i][j] = A[i-1][j]$
            if current character at y matches s
                $A[i][j] = A[i][j]$ or $A[i][j-1]$
    return $A[length of x][length of y]$
```

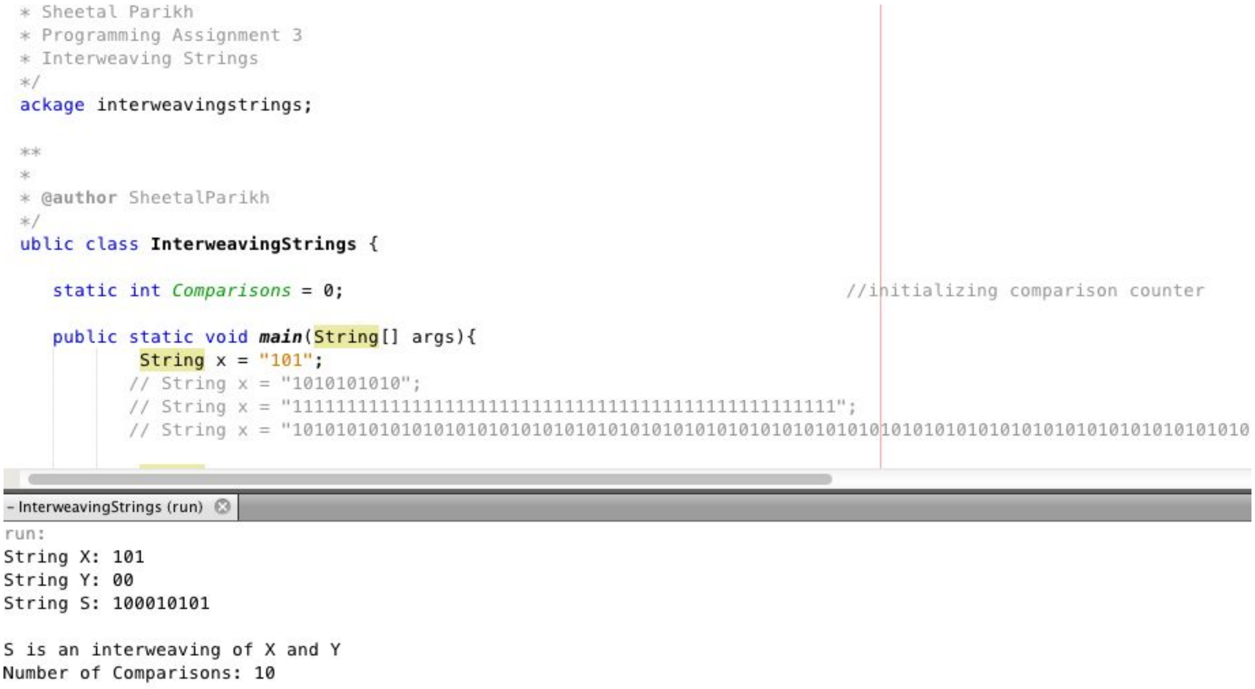
(b) Implement your algorithm above and test its run time to verify your analysis. Remember that CPU time is not a valid measure for testing run time. You must use something such as the number of comparisons.

As stated in part a, we assumed that the time complexity of the algorithm would be $O(nm)$. We get $n*m$ based on n, the length of string x and m, the length of string y. I ran 4 different tests based on different lengths of strings x, y, and s. The length of string s was a minimum of the length of string x plus the length of string y. I implemented a count of comparisons as the method of testing the complexity. I counted one comparison when any of the strings were compared to each other. (If the comparison count was implemented correctly) As seen below, the number of comparisons in the results, as assumed, correlated with the change in the length of x and y.

Results & Screenshots:

Output1: x.length = 3
y.length = 2
s. Length = 9

Predicted # of comparisons = 3*2 = 6
Actual # of comparisons = 10



```
* Sheetal Parikh
* Programming Assignment 3
* Interweaving Strings
*/
ackage interweavingstrings;

**
*
* @author SheetalParikh
*/
ublic class InterweavingStrings {

   static int Comparisons = 0;                                              //initializing comparison counter

   public static void main(String[] args){
           String x = "101";
          // String x = "1010101010";
          // String x = "11111111111111111111111111111111111111111111111111";
          // String x = "1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010
```

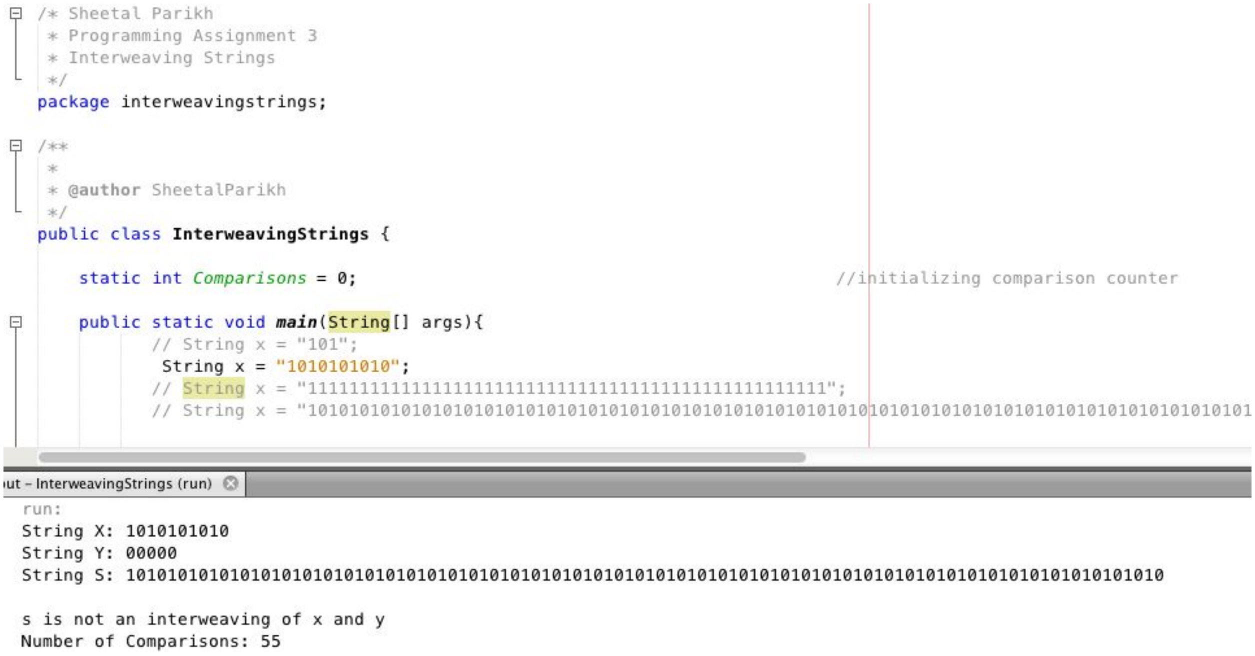
- InterweavingStrings (run)

```
run:
String X: 101
String Y: 00
String S: 100010101

S is an interweaving of X and Y
Number of Comparisons: 10
```

****

Output2:   x.length = 10
y.length = 5
s. Length = 50

Predicted # of comparisons = 5*10 = 50
Actual # of comparisons = 55

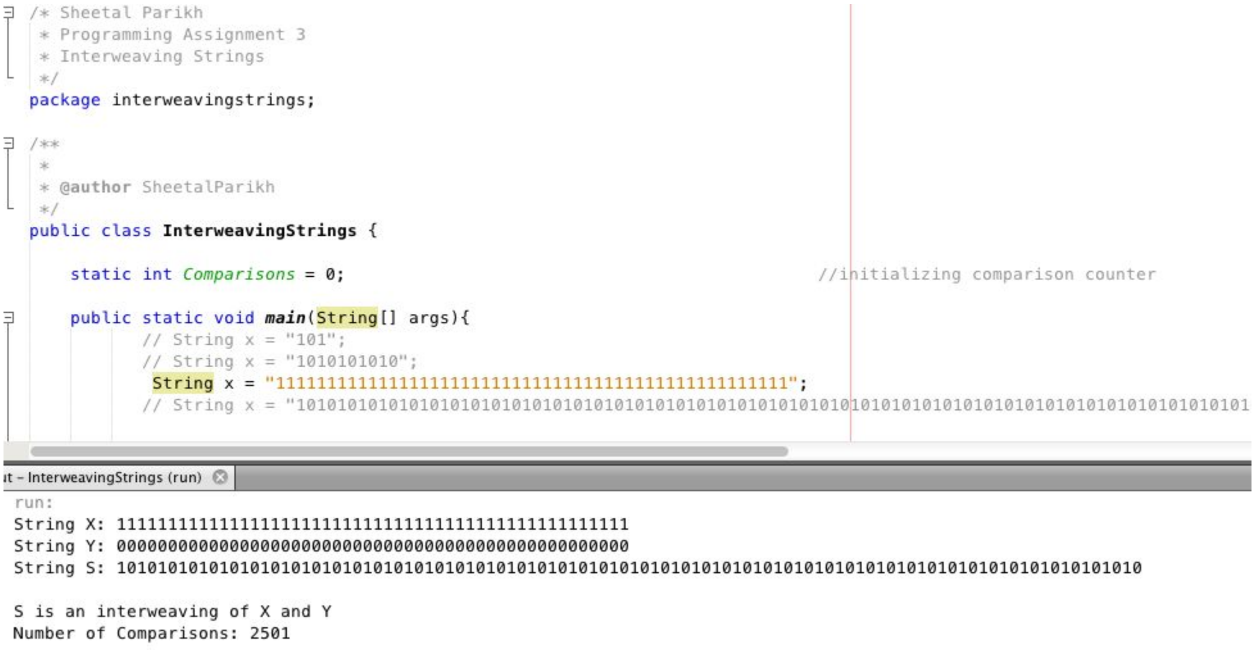


```
/* Sheetal Parikh
 * Programming Assignment 3
 * Interweaving Strings
 */
package interweavingstrings;

/**
 *
 * @author SheetalParikh
 */
public class InterweavingStrings {

    static int Comparisons = 0;                                              //initializing comparison counter

    public static void main(String[] args){
           // String x = "101";
            String x = "1010101010";
           // String x = "11111111111111111111111111111111111111111111111111";
           // String x = "1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
```

```
run:
String X: 1010101010
String Y: 00000
String S: 1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010

s is not an interweaving of x and y
Number of Comparisons: 55
```

Output3:   x.length = 50
y.length = 50
s. Length = 100

Predicted # of comparisons = 50*50 = 2500
Actual # of comparisons = 2501

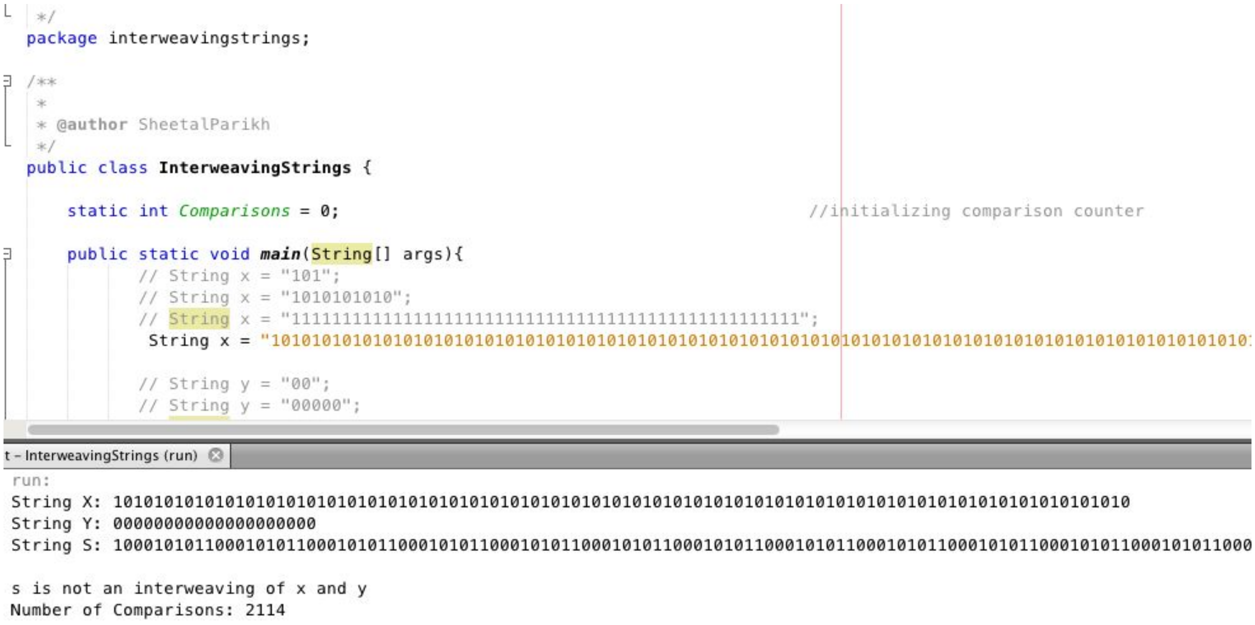


```
/* Sheetal Parikh
 * Programming Assignment 3
 * Interweaving Strings
 */
package interweavingstrings;

/**
 *
 * @author SheetalParikh
 */
public class InterweavingStrings {

    static int Comparisons = 0;                                          //initializing comparison counter

    public static void main(String[] args){
            // String x = "101";
            // String x = "1010101010";
             String x = "11111111111111111111111111111111111111111111111111";
            // String x = "101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
```

```
run:
String X: 11111111111111111111111111111111111111111111111111
String Y: 00000000000000000000000000000000000000000000000000
String S: 1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010

S is an interweaving of X and Y
Number of Comparisons: 2501
```

Output4:   x.length = 100
y.length = 20
s. Length = 126

Predicted # of comparisons = 100*20 = 2000
Actual # of comparisons = 2114



```
 */
package interweavingstrings;

/**
 *
 * @author SheetalParikh
 */
public class InterweavingStrings {

    static int Comparisons = 0;                                          //initializing comparison counter

    public static void main(String[] args){
            // String x = "101";
            // String x = "1010101010";
            // String x = "11111111111111111111111111111111111111111111111111";
             String x = "10101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010

            // String y = "00";
            // String y = "00000";
```

```
run:
String X: 1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010
String Y: 00000000000000000000
String S: 1000101011000101011000101011000101011000101011000101011000101011000101011000101011000101011000101011000101011000

s is not an interweaving of x and y
Number of Comparisons: 2114
```