Project Documentation
CSCI 174

Codeeval Challenge:
String Substitution

Submitted By:

Preeti Aggarwal

(109087257)

Submitted To:

Dr. David Ruby

PROBLEM STATEMENT:

Given a string S, and a list of strings of positive length, F1, R1, F2, R2,..., FN, RN. We have to proceed to find in order the occurrences (left-to-right) of Fi in S and replace them with Ri. All strings are over alphabet {0, 1}.

INPUT:

The program should accept as its first argument a path to a filename. Each line in this file is one test case. Each test case will contain a string, then a semicolon and then a list of comma separated strings.

OUTPUT:

For each line of input, print out the string after substitutions have been made.

CONSTRAINTS:

Searching should consider only contiguous pieces of S.

An iteration of the algorithm should not write over any previous replacement by the algorithm.

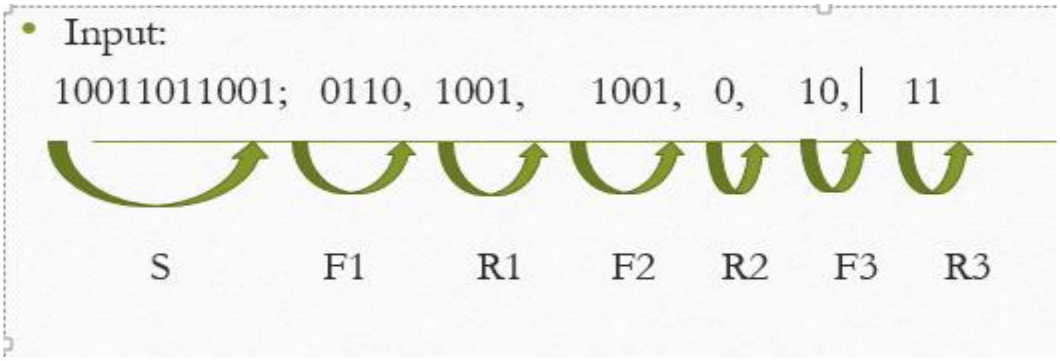
SAMPLE INPUT:

10011011001; 0110, 1001, 1001, 0, 10,11

SAMPLE OUTPUT:

11100110

DESCRIPTIVE EXAMPLE:



- Input:

10011011001; 0110, 1001, 1001, 0, 10,| 11

TRANSITIONS:

10011011001 => 10100111001 **[replacing 0110 with 1001**] => 10100110 **[replacing 1001 with 0]** => 11100110 **[replacing 10 with 11]**. So the answer is **11100110**

APPROACH:

1. Separates String S from the input string by taking the string before the ';'

2. Separates FN and RN from the input string by taking the string after the ';' and alternately storing them values split by ',' into their proper array

3. Then implement a function that takes String S and searches within it a string that matches FN. Matching string is replaced by string RN and modified final string is returned

4. To track the matching string a variable is used that serves as the marker where strings start to match. Finds a portion of S that matches with FN and restarts search at marker if mismatches at any point

STEPS TO SOLVE THE PROBLEM:

Step 1: First the file is read line by line. Each line contains a string S, and a list of strings of positive length, F1, R1, F2, R2,..., FN, RN.

Step 2: Then implemented a function that Separates String S from the input string by taking the string before the ';' using the Inbuilt ‘string split’ function.

Step 3: Similarly by using string split function, Separates FN and RN from the input string by taking the string after the ';' and alternately storing them values split by ',' into their proper array named as fn and Rn.

Step 4: After that implemented findfinalString Function that takes String S and searches within it a string that matches FN. Matching string is replaced by string RN and modified final string is returned.

Step 5: Inside findfinalString Function, first put the characters in String S into an array, characters of FN into an array then used a variable as marker an with while loop finds a portion of S that matches with FN and restarts search at marker if mismatches at any point

CODE EXPLAINATION:

Implementation: Java

There are descriptive comments in the code below:

```
 ******************   Source Code   ***************************


import java.io.*;

import java.util.*;


public class Main

{


        private static String[] F;            // Array of strings that needs to match

        private static String[] R;           //  Array of string to be replaced with


 /********       Function to read a file *******************/


       private Scanner openFile(String file)

       {
            Scanner x = null;
            try
            {
                x = new Scanner(new File(file));
            }
            catch(Exception e)
            {
                    System.out.println("Error reading file.");
            }
            return x;
        }

  /********      Function to read a line in a file *******************/

        private String readLine(Scanner x)
        {
            String line = x.next();
            return line;
        }



 /********      Function to close a file *******************/

        private void closeFile(Scanner x)
        {
            x.close();
        }
```

/* Function to separates String S from the input string by taking the string before the ';' */

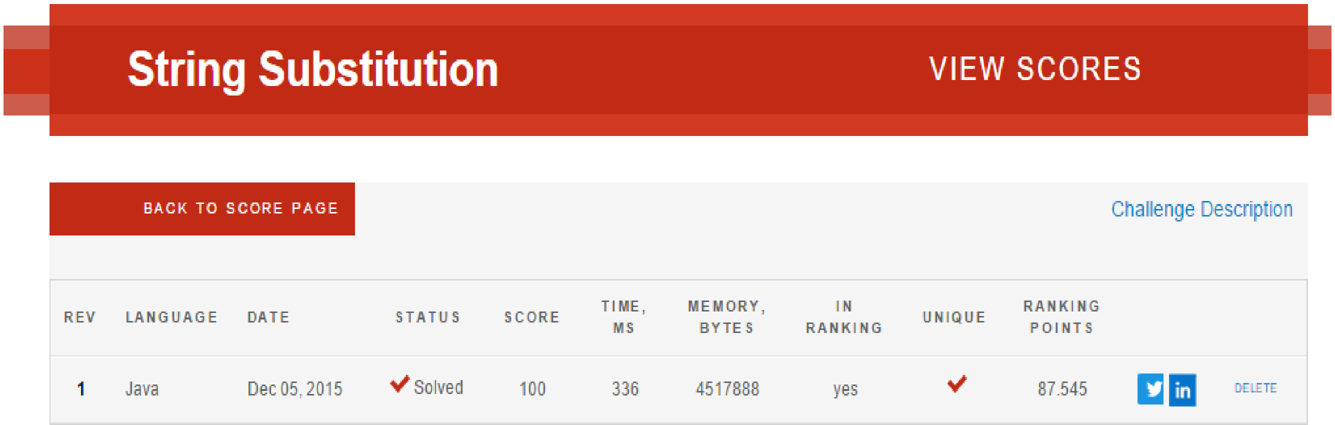
```
    public String extractString(String tc)
    {
        String[] parts = tc.split(";");
        String line = parts[0];
        return line;
    }
```

/** Separates FN and RN from the input string by taking the string after the ';' and alternately storing them values split by ',' into their proper array   ********/**

```
    public void extractxN(String tc)
    {
        String[] parts = tc.split(";");
        String[] nlist = parts[1].split(",");
        int nLen = nlist.length;
        int i = 0;
        String[] fn = new String[nLen/2];
        String[] rn = new String[nLen/2];
        for(int j = 0; j < nlist.length; j = j+2)
        {
            fn[i] = nlist[j];
            rn[i] = nlist[j+1];
            i++;
        }
        F = fn;
        R = rn;
    }


/******   Function that takes String S and searches within it a string that matches FN.
Matching string is replaced by string RN And modified final string is returned  **** */


    private static String findFinalString(String S)
    {
        String finalString = "";
        int strLen = S.length();

        //Puts the characters in String S into an array
        String[] strArray = new String[strLen];
        for (int i = 0; i < strLen; i++)
        {
            strArray[i] = "" + S.charAt(i);
        }

        int sizeFN = F.length;

        //Puts the characters of FN into an array
        for (int i = 0; i < sizeFN; i++)
        {
            int fLen = F[i].length();
            String[] fArray = new String[fLen];
            for (int j = 0; j < fLen; j++)
            {
                fArray[j] = "" + F[i].charAt(j);
            }
            //Initializes finalString to blank
            finalString = "";


            int j = 0;
            int k = 0;
           //Variable m serves as the marker where strings start to match
            int m = 0;


  /****  Finds a portion of S that matches with FN and restarts search at marker if
mismatches at any point *****/
            while (j < fLen && k < strLen)
            {
                if(fArray[j].equals(strArray[k]))
                {
                    if(j == 0)
                    {
                        m = k;
                    }
                    j++;
                    k++;
                }
                else
                {
                    if(j == 0)
                    {
                        m = k;
                    }
                    j = 0;
                    k = m + 1;
                }

       /****  Replaces FN for RN by replacing the first array of the matching portion
       of S with RN and turning the rest of the portion blank  ****/
                if(j == fLen)
                {
                    strArray[m] = "r" + R[i];
                    for(k = 1; k < fLen; k++)
                    {
                        strArray[k + m] = "r";
                    }
                    j = 0;
                    k = m + fLen;
                }
            }
        }
        for(int i = 0; i < strLen; i++)
        {
            strArray[i] = strArray[i].replace("r", "");
            finalString = finalString + strArray[i];
        }
        return finalString;
    }

    public static void main(String[] args)
    {

        Main m = new Main();
```

```
		String S = null;
		Scanner x = m.openFile(args[0]);       // call to openline function defined above
		while(x.hasNext())
		{
			String line = m.readLine(x);          // call to readline fun defined above
			S = m.extractString(line);         // call to extractstring fun defined above
			m.extractxN(line);    // separate fi and Ri strings with fun defined above

			String finalString = findFinalString(S);     // returns the final string
			System.out.println(finalString);               // prints final string

		}
		m.closeFile(x);
	}
}
```

Codeeval Score:



REV	LANGUAGE	DATE	STATUS	SCORE	TIME, MS	MEMORY, BYTES	IN RANKING	UNIQUE	RANKING POINTS		
1	Java	Dec 05, 2015	✓ Solved	100	336	4517888	yes	✓	87.545		DELETE