CMP3005 Analysis of Algorithms Term Project Report

Project Purpose: The goal of this project is to find missing parts of the given sentence from large text.

Implementation

Programming Language: Java

Libraries: We use standart Java libraries.

```
1 import java.io.BufferedReader;
2 import java.io.FileInputStream;
3 import java.io.FileNotFoundException;
4 import java.io.InputStream;
5 import java.io.InputStreamReader;
6 import java.util.ArrayList;
7
```

We used BufferedReader, FileInputStream, InputStream and InputStreamReader for read from text files.

We used ArrayList library to get the questions and put them into arraylist.

Methods Used: We used similarly brute force string search method.

Function search takes smaller text , larger text in this order as a parameter.

One outer for for-loop and inner while loop.Outer for loop begins from 0 to larger text size – smaller text size. For loop increase the index in the larger text. Outer for loop begins from j=0 to smaller text size and each iteration it checks whether the chars at that index is matching with larger text. If its match increase j. Outside of the while

loop we control if j=smaller text length, we return the beginning index of the wanted word. If we didn't find the sentence, we return -1. We used first if for speed up the algorithm. This if checks if the first index of the statement does not match with larger text's first index we passed to other for iteration by continue keyword.

```
    public static int search(String statement, String showScript) {
        for(int i=0;i<showScript.length()-statement.length();i++) {
            int j = 0;
            if(showScript.charAt(i)!=statement.charAt(0)) {
                continue;
            }
            while (j < statement.length() && (statement.charAt(j) == showScript.charAt(i+j))) {
                j++;
            }
            if(j == statement.length()) {
                return i+statement.length();
            }
        }
        return -1;
    }

}
```

```
ArrayList<String> statement=new ArrayList<String>();
String showScript = "";
int i = 0;
try {
    InputStream is = new FileInputStream("statements.txt");
    BufferedReader buf = new BufferedReader(new InputStreamReader(is));

    InputStream is1 = new FileInputStream("the_truman_show_script.txt");
    BufferedReader buf1 = new BufferedReader(new InputStreamReader(is1));
    try {
        statement.add(buf.readLine());
        while(statement.get(i) != null) {
            statement.add(buf.readLine());
            i++;
        }
        showScript = buf1.readLine();
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
} catch (FileNotFoundException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

In this code we get from text line by line and put smaller text into arraylist. We put larger text into a string. We used try catch for if there is a error about reading file we catch the error.

```
String[] arr1 = {statement.get(j).substring(0, statement.get(j).indexOf("_")),statement.get(j).substring(statement.ge
arr = statement.get(j).split("_", 0);
if(arr1[1].length()>arr1[0].length() ) {
    index = search(arr1[1], showScript);
    stoplength= index-(arr1[1].length());
    if(index == -1) {
        System.out.println("Statement NOT found\n");
        continue;
    }
    result = "";
    String result2 = "";
    for(int m=stoplength-1;m>0;m-- ) {
        if(showScript.charAt(m) ==' ') {
            break;
        }


        result+=String.valueOf(showScript.charAt(m));

    }
    for(int a=result.length()-1;a>=0;a--) {
        result2 += result.charAt(a);
    }
```

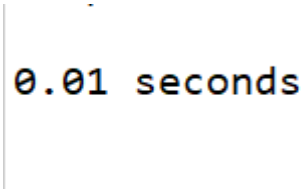
We create a string arrays and we assign into statement's subsitrings by using subsitring and split methods. The algorithm puts the subsitring before " ___" to array's 0 index and after "___" of the sentence is putted array 1 index. There are 2 situations here. First one if after the "___" is larger than before we used substring method because split method does not split both parts rigth. We used a for loop to find the result. Loop is started just after "___" and continues until it sees the space. We managed to add chars reverse order using simple for loop.

```
else {
    //arr = statement.get(j).split("_", 0);
stoplength = arr[0].length()+3;
index = search(arr[0], showScript);

if(index == -1) {
    System.out.println("Statement NOT found\n");
    continue;
}
result = "";
for (int k = index; k < showScript.length(); k++) {
    if (showScript.charAt(k) == statement.get(j).charAt(stoplength)) {
        break;
    }
    result += String.valueOf(showScript.charAt(k));
}
```

If the after ”___” statement is bigger code enter else situation.We create stopword integer which keeps the index of after ”___” . In the else there is for loop and it begins before ”___” and continues until stopwords index which is next word’ beginning index. Each iteration puts chars into result.

<u>The Average Speed Of Our Algorithm:</u>



```
else if(arr1[1].length()==0){

    index2 = search(arr[0], showScript);

    if(index2 == -1) {
        System.out.println("Statement NOT found\n");
        continue;
    }
    result = "";
    for (int k = index2; k < showScript.length(); k++) {
        if (showScript.charAt(k) == ' ') {
            break;
        }
        result += String.valueOf(showScript.charAt(k));
    }

    System.out.println("Original statement: " + statement.get(j));
    System.out.println("Completed statement: " + arr[0] + result+"\n");

}
```

If “___” is in the end of the statement we changed the stop character to ‘ ’(space). The code will add all characters until it sees the space.

- Buğra Mert Ayar (1730080)
- Anıl Şülekoğlu (1602414)