Algorithms Assignment 1

Christina Clements

Question 2

I uploaded the code I wrote into GitHub as well.

```
public class commonSubstring {
  public static String findCommonString(String text1, String text2) {
    int maxLength = 0;
    int startIndex = 0;

    for (int i = 0; i < text1.length(); i++) {
      for (int j = 0; j < text2.length(); j++) {
        int length = 0;
        while (i + length < text1.length() && j + length < text2.length() &&
            text1.charAt(i + length) == text2.charAt(j + length)) {
          length++;
        }
        if (length > maxLength) {
          maxLength = length;
          startIndex = i;
        }
      }
    }

    if (maxLength > 0) {
      return text1.substring(startIndex, startIndex + maxLength);
    } else {
      return "";
    }
  }
```

Question 6

Problem 1:

Time Complexity Big-O: O(m*n) Big-Ω: Ω(m*n) where m = length of text1, n= length of text2

The time complexity is O(m*n) because the nested for loops will run m*n amount of times, where m = length of text1, n= length of text2. The variables n and m are independent.

Space Complexity Big-O: O(m*n) Big-Ω: Ω(m*n) where m = length of text1, n= length of text2

The space complexity is O(m*n) because of the 2D array being created with a size (m+1)*(n+1), which simplifies to m*n without constants. The variables n and m are independent.

Problem 2

Time Complexity Big-O: O(m*n*min(m,n)) Big-Ω: Ω(m*n) where m = length of text1, n= length of text2

The time complexity is O(m*n*min(m,n)) because the nested for loops will run m*n amount of times, where m = length of text1, n= length of text2. There is also a while loop that run min(m,n) times.

Space Complexity Big-O: O(min(m,n)) where m = length of text1, n= length of text2, the maxLength is the minimum of the two variables

Big-Ω: Ω(1) is no common substring

Problem 3