Ryan Coslove rmc326

CS111 Trees

Assignment 5

Problem 1

1.

C Q S A X B T //

A Q S C X B T // C Swaps with A; 7 comparisons

A B S C X Q T // Q Swaps with B; 6 comparisons

A B C S X Q T // S Swaps with C; 5 comparisons

A B C Q X S T // S Swaps with Q; 4 comparisons

A B C Q S X T // X Swaps with S; 3 comparisons

A B C Q S T X // T swaps with X; 2 comparisons

2.

1 C Q S A X B T // No Swap

2 C Q S A X B T // No swap

3 A C Q S X B T // A swapped with S, A with Q, A with C

4 A C Q S X B T // No swap`

5 A B C Q S X T // B swapped with X, B with S, B with Q, B with C

6 A B C Q S T X // T swapped with X

Problem 2

1. READ array1

READ array2

SET n equal to the total amount of objects in array1

SET m equal to the total amount of objects in array2

SET commonElement to FALSE

FOR i equal to 0, WHILE i is less than n, INCREMENT i by 1

FOR j equal to 0, WHILE j is less than n, INCREMENT j by 1

```
                IF arr1[i] operand IS EQUAL TO arr2[j] operand
                    SET commonElement equal to TRUE
                    BREAK
                ENDIF
            END FOR
        END FOR
        ENDIF
        IF commonElement EQUALS false
            DISPLAY “Array 1 and Array 2 share nothing in common”
        ELSE
            DISPLAY “Array 1 and Array 2 have some similarity”
        ENDIF
```

2. Factors that influence the running time: “n” and “m”, both of which are the array sizes of “array1” and “array2” respectively\.

3.

READ array1	1
READ array2	1
SET n equal to the total amount of objects in array1	1
SET m equal to the total amount of objects in array2	1
SET commonElement to FALSE	1
FOR i equal to 0, WHILE i is less than n, INCREMENT i by 1	n + 1 times
FOR j equal to 0, WHILE j is less than n, INCREMENT j by 1	m + 1 times
IF arr1[i] operand IS EQUAL TO arr2[j] operand	m*n times
SET commonElement equal to TRUE	1 time
BREAK	
ENDIF	
END FOR	
END FOR	
ENDIF	

IF commonElement EQUALS false 1 time

DISPLAY “Array 1 and Array 2 share nothing in common” 1 time

ELSE

DISPLAY “Array 1 and Array 2 have some similarity” 1 time

ENDIF

4. mn + m + n + 11

5. Best case: n O(1)

6. Worst case: n O(m*n)

b.

```
SET duplicateChar equal to 0;
SET repeats(char, String)
SET count equal to 0
    FOR index equal to 0, WHILE index is less than original.length(String), INCREMENT index
        IF String[index] == char
            COMPUTE count plus 1
            RETURN count
        ENDIF
    FOR i equal to 0, WHILE i is less than original.length(String), INCREMENT i by 1
        SET count equal to repeats(String[i], String)
        IF count is greater than 1
            COMPUTE repeats += count
        END IF
    ENDFOR
    IF repeats == 0
    DISPLAY “The string doesn’t have any duplicate characters”
    ELSE
    DISPLAY “Total duplicates are:” + repeats
```

2. Factors: length of the String, n

3. Each step takes one operation except the for function which will take as many steps as the length of the string.

SET duplicateChar equal to 0; 1

SET repeats(char, String) 1

SET count equal to 0 1

FOR index equal to 0, WHILE index is less than original.length(String), INCREMENT index n

IF String[index] == char 1

COMPUTE count plus 1 1

RETURN count 1

ENDIF

FOR i equal to 0, WHILE i is less than original.length(String), INCREMENT i by 1 n

SET count equal to repeats(String[i], String) 1

IF count is greater than 1 1

COMPUTE repeats += count 1

END IF

ENDFOR

IF repeats == 0 1

DISPLAY “The string doesn’t have any duplicate characters” 1

ELSE

DISPLAY “Total duplicates are:” + repeats 1

4. n^2 + 12 The algorithm takes n*n or n^2 operations

5. Best case: n O(n^2)

6. Worst case: n O(n^2)

c.

c.

READ 2DArray

READ charInput

SET count EQUAL to 0

SET Boolean fullRow to false

FOR each row in the 2DArray

FOR each character in that row of 2DArray

IF the character at the current index is equal to charInput

ADD 1 to count

ENDIF

ENDFOR

IF count is equal to the length of this row

SET fullRow to True

ENDIF

ENDFOR

IF fullRow EQUALS true

DISPLAY “There is a row where all the chars are ” + charInput

ELSE

DISPLAY “There is a row where all the chars are ” + charInput

ENDIF

2. Factors: length of the 2D array and the length of each array within the 2D array

3.

READ 2DArray	1
READ charInput	1
SET count EQUAL to 0	1

SET Boolean fullRow to false 1

FOR each row in the 2DArray n

FOR each character in that row of 2DArray m

IF the character at the current index is equal to charInput 1

ADD 1 to count 1

ENDIF

ENDFOR

IF count is equal to the length of this row 1

SET fullRow to True 1

ENDIF

ENDFOR

IF fullRow EQUALS true 1

DISPLAY “There is a row where all the chars are ” + charInput 1

ELSE

DISPLAY “There is a row where all the chars are ” + charInput 1

ENDIF

4. n*m + 11

5. Best case is if the first element in each row is not ‘x’ and loop breaks, so best case is big O(m)

6. Worse case is no elements is ‘x’ in the rows so the big O(m*n)