National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	**CS302**
Degree Program:	**BS(CS)**	**Semester:**	**Spring 2020**
Exam Duration:	**60 Minutes**	**Total Marks:**	**40**
Paper Date:	**Feb 24, 2020**	**Weight**	**15**
Section:	**ALL**	**Page(s):**	**4**
Exam Type:	**Mid Term 1 Exam**		

Student : Name: _____ **Roll No.** _____ **Section:** _____

Instruction/Notes: Attempt the examination on the question paper and write concise answers. You can use extra sheet for rough work. Do not attach extra sheets used for rough with the question paper. Do not use pencil and red ink. Don't fill the table titled Questions/Marks.

Question	1	2	3	4	5	Total
Marks	/15	/5	/5	/10	/5	/40

Q1) Suppose the higher management of FAST-NU is impressed by your problem solving skills and want you to add a new feature in flex that can answer the following query efficiently i.e. it should work in O(1): For a given class how many students earned grades between grade X and grade Y inclusive? Where grade X and Y can be any of these grades: {A+, A, A-, B+, B, B-, C+, C, C-, D+, D, F}. If you want you can preprocess the data beforehand so that you can answer the query within desired time bounds. **[15 Marks]**

Do you think you need to preprocess the data? If yes then explain your idea in 3-4 lines and then give the preprocessing steps (pseudo code) in details. Otherwise justify your answer.

Yes preprocessing is required. If for each grade g, we count the number of students having grade less or equal to g then we can answer this query in O(1) time. Below are steps
Make an array ***count*** of size 12 where grade F maps to index 0 and grade D maps to index 1 and so on
Define an O(1) function map that takes grade as input and returns the corresponding number e.g. for F it should return 0.
Initialize array count to zero
For i=1 to n
    count[map(student_grade)]++
For i=1 to 11
    count[i] = count[i] + count[i-1]

How much time will be required to preprocess the data if any? Give your answer in terms of n where n is number of students in the class.

O(n)

How will you answer the query in O(1)

Given grade X and Y max = maximum(map(X),map(Y)) and min = minimum(map(X),map(Y))
If(min>0)
    Total students = count[max] – count[min-1]
else
   Total students = count[max]

Q2) Suppose we are sorting an array of eight integers using Quick sort, and we have just finished the first partitioning with the array looking like this:

2 6 3 8 11 19 13 14 **[5 Marks]**

Which statement is correct?

a) The pivot could be either the 8 or the 11.
b) The pivot could be the 8, but it is not the 11.
c) The pivot is not the 8, but it could be the 11.
d) Neither the 8 nor the 11 is the pivot.

(a)

Q3) What is the asymptotic complexity of following algorithms when the input array is completely sorted in the desired order? **[5 Marks]**

1. Insertion sort
2. Quick sort(not randomized quick sort)
3. Merge Sort

1. **O(n)**
2. **$O(n^2)$**
3. **O(nlgn)**

Q4) Consider the following piece of code. Here, initially, | right - left | = n. Assume that n is large.

```
Function1(A, left, right)
          IF left < right
          {
                    k <- ((2*left)+right)/3

                    IF(rand()%2)
                    {
                              Function1(A, left, k)
                    }ELSE{
                              Function1(A, k, right)
                    }

                    Function2(A, left, right) //O(n) method
          }
```

i) Write a recurrence for the worst case and best case of Function1 **[5 Marks]**

Best Case:
T(n) = T(n/3) + O(n)
Worst Case:
T(n) = T(2n/3) +O(n)

ii) Solve the recurrence of worst case to find big-Oh bound (as tight as possible). Show complete working. **[5 Marks]**

T(n) = T(2n/3) +cn = T(4n/9) + c2n/3 + cn =

$$T(n) = T\left(\frac{2n}{3}\right) + cn = T\left(\frac{4n}{3}\right) + \frac{2cn}{3} + cn = T\left(\left(\frac{2}{3}\right)^i n\right) + cn\sum_{k=0}^{i-1}(2/3)^k$$

$$For\ \left(\frac{2}{3}\right)^i n = 1 \rightarrow i = log_{3/2}n$$

$$T(n) = T(1) + cn \sum_{k=0}^{log_{3/2}n-1} (2/3)^k \leq T(1) + cn\sum_{k=0}^{\infty}\left(\frac{2}{3}\right)^k = O(n)$$

Q5) Your friend makes the claim that the running time of their algorithm is both O(n) and O(nlgn). Are there any circumstances under which your friends claim may be true? **[5 Marks]**

If the running time of an algorithm is O(n) then it is also O(nlgn) (loose upper bound)