National University of Computer and Emerging Sciences, Lahore Campus

Course:	Design & Analysis of Algorithms	Course Code:	CS-2009
Program:	BS (Computer/Data Science)	Semester:	Fall 2023
Duration:	60 Minutes	Total Marks:	19
Paper Date:	2-Oct-22	Section:	ALL
Exam:	Midterm 1	Page(s):	6
Name		Roll Number	

Instruction/Notes: Solve it on question paper

Question	1	2	3	4/5	Total
Marks	/4	/2	/8	/5	/19

Q1) Multiple Choice Questions [4 Marks]

a) Which of the following sort algorithms are stable?

i. Quick Sort
ii. Merge Sort
iii. Insertion Sort
iv. Count Sort

b) Which of the following sort algorithms are guaranteed to be O(n log n) even in the worst case?

i. Quick Sort
ii. Merge Sort
iii. Insertion Sort

c) 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 5 0 6 8 13 9 10

Which statement is correct?

i. The pivot could be either the 6 or the 8.
ii. The pivot could be the 6, but it is not the 8.
iii. The pivot is not the 6, but it could be the 8.
iv. Neither the 6 nor the 8 is the pivot.

d) $f(n) = 30*2^n + 15*4^n + 3*16^n$

which of the following statements are true about f(N)

i. $f(n) = O(4^{2n})$
ii. $f(n) = O(8^{n+2})$
iii. $f(n) = O(2^{4n})$
iv. $f(n) = O(4^{n+4})$

Q2) What is asymptotic time complexity of following code. n is size of input. [2 Marks]

```
for(i=1; i<=n; i++)
{
   for(j=1; j<=n; j=j*2)
   {
      Arr[i] = Arr[i] + (Arr[j]*0.1)
    }
 }
```

Q3) [8 Marks] There are 2 sorted arrays **A** and **B** of size n each. Write an algorithm to find the median of the array obtained after merging the above 2 arrays (i.e. array of length 2n).

Input : ar1[] = {1, 12, 15, 26, 38}
ar2[] = {2, 13, 17, 30, 45}
Output : 16

Explanation :
After merging two arrays, we get
{1, 2, 12, 13, 15. 17, 26, 30, 38, 45}
Middle two elements are 15 and 17
Average of middle elements is (15 + 17)/2
which is equal to 16

(a) [4 Marks] Write an algorithm to solve the above problem that takes O(n) time.

(b) [4 Marks] Write an algorithm to solve the above problem that takes O(lg n).
Hint: You do not need to merge the two arrays. Use divide and conquer approach.

Question 4 is only for Sections BDS-(5A, 5B, 5C), BCS-(5C, 5D, 5E), BSR-5A

Q4) (a) Solve the following recurrence and write time complexity in asymptotic notation. Show all working [3 Marks]

$T(n) = 3T(n/3) + n^2$

Q4) **(b)** Consider the following algorithm

```
splitData(Arr[], left, right)
{
   if(left < right)
   {
      splitData(Arr, left, (left+right)/2)
      splitData(Arr, (left+right)/2 + 1, right)
      for(i=left; i<=right; i++)
      {
         for(j=1; j<=i; j++)
         {
            Arr[i] = Arr[i] + (Arr[j]*0.1)
         }
      }
   }
}
```

Give the recurrence for the worst-case running time of above algorithm. Only write the recurrence, you do not need to solve it [2 Marks]

Question 5 is only for Sections BCS-5A, BCS-5B

Q5) (a) [3 Marks] Prove that following statement is valid
$3n^2 + 29\ n = O\ (n^2)$
Prove by providing the value of c and n_0.

Q5) (b)[2 Marks]

i. How many comparisons will be made to sort the array arr={1,5,3,8,2} using counting sort?
   a) 5
   b) 7
   c) 9
   d) 0

ii. Which of the following sorting techniques is most efficient if the range of input data is significantly greater than a number of elements to be sorted?
   a) selection sort
   b) bubble sort
   c) counting sort
   d) merge sort