National University of Computer and Emerging Sciences

Lahore Campus

Design and Analysis of Algorithms (CS2009)

Date: April 6, 2024

Sessional-II Exam

Total Time (Hrs): 1
Total Marks: 20
Total Questions: 3

Course Instructor(s)

Dr. Saira Karim
Dr. Aasim Qureshi
Ms. Abeeda Akram
Mr. Usama Hassan Alvi
Mr. Sajid Ali Kazmi
Mr. Iteza Muzaffar

20L-0933
Roll No

4G
Section

Student Signature

Do not write below this line

Attempt all the questions.

CLO #1: Design algorithms using different algorithms design techniques i.e. Brute Force, Divide and Conquer, Dynamic Programming, Greedy Algorithms and apply them to solve problems in the domain of the program

Q1: Given an array A of n numbers in the range of [-n, n], devise an algorithm that finds the number of distinct pairs (i, j) such that j>i and A[i] = A[j]. Your algorithm must work in O(n) time.

[marks 10]

Sample Input:
[1, 2, 1, 2, 3, 1, 4]
Sample Output: 4
Sample Input:
[-1, -1, -1, -1, 4]
Sample Output: 6

CLO #1: Design algorithms using different algorithms design techniques i.e. Brute Force, Divide and Conquer, Dynamic Programming, Greedy Algorithms and apply them to solve problems in the domain of the program

Q2: Catalan numbers form a fascinating sequence of natural numbers that occur in various counting problems and have many applications in combinatorial mathematics. The first few Catalan numbers for n = 0, 1, 2, 3, 4, 5, 6, 7 … are 1, 1, 2, 5, 14, 42, 132, 429, …
Following recurrence is used for calculation of the nth Catalan number.

$$C_0 = 1, C_1 = 1 \text{ and } C_n = \sum_{i=0}^{n-1} C_i C_{n-1-i} \quad for\ n > 2$$

0271

Consider the following recursive solution for calculation of the nth Catalan.

```
int catalan(n)
{
  if (n <= 1)
    return 1
  num = 0
  for i = 0 to n
    num = num + (catalan(i) * catalan(n - i - 1))
  return num
}
```

Write the pseudocode for the bottom-up DP solution of this problem. Also provide the time complexity of your solution. **[marks 5]**

CLO #1: Design algorithms using different algorithms design techniques i.e. Brute Force, Divide and Conquer, Dynamic Programming, Greedy Algorithms and apply them to solve problems in the domain of the program

Q3: Given N algorithm design problems in an exam with a total exam time of T minutes, each problem has a certain number of checkpoints and marks of a problem are equally distributed for each checkpoint. You are not sure whether you will be able to attempt all the questions or even all the checkpoints of a certain problem. The goal is to maximize the score within the given time constraint. You are given the information of total exam time (T), total number of questions (N), marks and expected time for each problem in arrays M[1...N] and E_Time[1...N] respectively. A student devised an algorithm to maximize the score by prioritizing the problems with maximum marks using a greedy approach. Do you think this greedy approach will always provide an optimal solution. If not, then provide a counter example where this approach will fail. In case of a counter example, you are supposed to follow the same pattern of sample input i.e., clearly mention all the information and a proper justification is required which shows that why this greedy approach will fail to provide an optimal solution. **[marks 5]**

Sample Input:

T = 60 minutes
N = 5
Marks: {30, 25, 20, 15, 10}
Expected Time: {25, 20, 15, 30, 20}
Max Achievable Score = 75