National University of Computer and Emerging Sciences

Lahore Campus

D & A of Algorithms (CS2009)

Date: Nov 04 2024

Course Instructor(s)

Dr. Maryam, Dr. Asim, Miss Abeeda

Sessional- II Exam

Total Time (Hrs):	**1**
Total Marks:	**15**
Total Questions:	**2**

_____ _____ _____

Roll No Section Student Signature

Instructions: Attempt all questions. Answer in the space provided. Do not attach rough sheets with this exam.

CLO #1: Design algorithms using different algorithm 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

Question 1: [10 Marks]

The longest increasing subsequence (LIS), of any input sequence of numbers a1, . . ., an is a subset of these numbers taken in order, of the form $a_{i1}, a_{i2}, \ldots, a_{ik}$ where $1 \le i1 < i2 < \cdots < ik \le n$, in which the numbers are getting strictly larger. The task is to find the length of the longest increasing subsequence.

For example, the length of the LIS of **5, 2, 8, 6, 3, 6, 9,7, 2** is **4**, and the longest sequence is **2, 3, 6, 9**. Consider the following recursive method that calculates and returns the length of the longest increasing subsequence.

```
//Recursive function, the LIS for the length n sequence is stored in maxLIS
 LIS(A , n, maxLIS)
      if (n==1)
            return 1
      max = 1
      for (i = n-1 down to 1)
            curr = LIS(A, i, maxLIS);
            if (A[n] > A[i] And curr + 1 > max)
                  max = curr + 1

      if (max > maxLIS)
            maxLIS = max
      return max
```

a) Which of the following best describes the time complexity class of the above function? Choose from the following options **[Marks:1]**

i. Exponential
ii. Quadratic
iii. Logarithmic
iv. Linear

b) Write the Dynamic programming **bottom-up (iterative)** algorithm of the LIS problem described above. (Hint: Convert the recursive code given above into iterative dynamic programming solution by using the idea of memoization) **[Marks: 5]**

c) Provide the time complexity of your DP solution in terms of Big-θ. **[Marks: 1]**

d) Do we need to apply Dynamic programming approach on the following problem. Explain your answer in 2 to 3 lines. [2 Marks]
"We are interested in finding the maximum subarray sum (contiguous) of a given 1-D array. Array will contain only positive numbers"

e) You're using a dynamic programming approach to solve the Longest Common Subsequence (LCS) problem on two strings of lengths 10 and 15. How many subproblems will you need to compute? [1 Marks]

i. 150
ii. 2^{10}
iii. 2^{15}
iv. 25

CLO 4: **Implement** the algorithms, compare the implementations empirically, and apply fundamental algorithms knowledge to solve practical problems related to the program.

Question 2: [1*5 = 5 Marks]

a) How many comparisons will be made to sort the array arr = {1, 5, 3, 8, 2} using LSD (starting with the least significant digit) radix sort?

i. 5
ii. 7
iii. 9
iv. 0

b) Sort the following list using the Radix sort algorithm (starting with the least significant digit).
329, 457, 839, 436, 720, 355, 657
What is the output of the algorithm after the second pass?

i. 329, 355, 436, 457, 657, 720, 839
ii. 355, 329, 457, 436, 720, 657, 839
iii. 720, 329, 436, 839, 355, 457, 657
iv. 720, 355, 436, 457, 657, 329, 839

c) Suppose you have a dataset of 1,000,000 integers ranging from -500,000 to 500,000. Which of the following statements best explains why Counting Sort might not be the most suitable choice for this data?

i. Counting Sort is not suitable because it cannot handle negative numbers.
ii. Counting Sort would require a significant amount of additional memory to store the count array, making it not a suitable choice.
iii. Counting Sort has a worst-case time complexity of $O(n^2)$ when applied to large datasets.
iv. Counting Sort requires the input data to be uniformly distributed across the range for optimal performance.

d) You are sorting an array with Counting Sort where the range of values is significantly larger than the number of elements in the array. For example, an array of 10,000 elements contains values from 1 to 10,000,000. Which of the following statements best describes the impact of using Counting Sort in this scenario?

i. Counting Sort will still perform efficiently if the array size is smaller than the range of values.
ii. Counting Sort's performance will degrade due to the high space requirement for the count array, making it inefficient for this scenario.
iii. Counting Sort will be optimal as it sorts in O(n) time regardless of the range of values.
iv. Counting Sort will rearrange elements in a non-stable way because of the large range of values relative to the number of elements.

e) You need to sort a dataset containing 2 million 12-digit positive integers (each in the range from 000000000001 to 999999999999) that represent transaction IDs. The dataset has a fixed number of digits, and there are no negative numbers. Which sorting algorithm would be the most efficient choice for this task, given the characteristics of the data?

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