National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	**Design and Analysis of Algorithms**	**Course Code:**	**CS2009**
	Degree Program:	**BSCS**	**Semester:**	**Spring 2024**
	Due Date:	**28-03-2024**	**Total Marks:**	**60**
	Section:	**J**	**Page(s):**	**3**
	Exam Type:	**Assignment 2**	**CLO**	**2**

Student : Name:_______________________________ Roll No.________________ Section:_______

Instructions/Notes:

1. You have to submit this assignment in hand written form, no print/online submission will be accepted.
2. No late submission will be entertained.
3. Plagiarism of any sort will not be tolerated. Also, there will be a viva at the end of the semester and if you are unable to explain your work in any assignment, your submission for that particular assignment will be cancelled.
4. Assignment will be collected at the start of class on Thursday March 28, 2024

For all of the questions, you have to perform following tasks:

1. Prove that the optimal substructure property exists
3. What will be the smaller sub-problems?
4. Recursive definition/formula
5. Write a dynamic programming algorithm for the problem that gives optimal value as well as optimal solution.
6. Space and time complexity of your solution.

Question 1: [marks = 10]

Given a set of jobs, each with a specified deadline and profit, the task is to schedule the jobs in a way that maximizes the total profit earned while adhering to the deadlines. Each job can be processed only once, and only one job can be processed at a time. The objective is to find a schedule that maximizes the total profit, considering that a job completed after its deadline yields no profit.
Example:
Job 1 (deadline: 4, profit: 70), Job 2 (deadline: 2, profit: 60), Job 3 (deadline: 4, profit: 50), Job 4 (deadline: 3, profit: 40), Job 5 (deadline: 1, profit: 30).
The optimal schedule to maximize profit is: Job 1 at slot 4, Job 2 at slot 2, Job 3 at slot 3, Job 4 at slot 1, and Job 5 at slot 0.
Total Profit: 250.

Question 2: [marks = 10]

Given *N* balloons, indexed from 0 to *n-1*. Each balloon is painted with a number on it represented by an array *arr*. You are asked to burst all the balloons. If you burst balloon *i* you will get *arr [left] * arr [i] * arr [right]* coins. Here *left* and *right* are adjacent indices of *i*. After the burst, the *left* and *right* then becomes adjacent.

Find the maximum coins you can collect by bursting the balloons wisely.

Note:

(1) You may imagine *arr[-1] = arr[n] = 1*. They are not real therefore you cannot burst them.
(2) $0 \le n \le 500$, $0 \le arr[i] \le 100$

Question 3: [marks = 10]

There is a tower of *K*-floors, and an egg dropper with *N* ideal eggs. The physical properties of the ideal egg are such that it will shatter if it is dropped from floor *k* or above, and will have no damage whatsoever if it is dropped from floor *k-1* or below.
Suppose you have *N* eggs and you want to determine from which floors in a *K*-floor building you can drop an egg such that it doesn't break. You have to determine the minimum number of attempts you need in order to find the critical floor in the worst case while using the best strategy.

Question 4: [marks = 10]

A *contiguous subsequence* of a list S is a subsequence made up of consecutive elements of S. For instance, if S is

$$5, 15, -30, 10, -5, 40, 10,$$

then $15, -30, 10$ is a contiguous subsequence but $5, 15, 40$ is not. Give a linear-time algorithm for the following task:

> *Input:* A list of numbers, $a_1, a_2, \ldots, a_n$.
> *Output:* The contiguous subsequence of maximum sum (a subsequence of length zero has sum zero).

For the preceding example, the answer would be $10, -5, 40, 10$, with a sum of 55.
(*Hint:* For each $j \in \{1, 2, \ldots, n\}$, consider contiguous subsequences ending exactly at position j.)

Question 5: [marks = 10]

Min-Coin Change is the problem of using the minimum number of coins to make change for a particular amount of cents, s, using a given set of denominations d_1, d_2, ... d_n. For example given the denominations 1, 5, 10, 20, 25, and you wish to make change for 40 cents using minimum number of coins. The optimal answer for this example is 2 coins of denomination 20. Device a dynamic programming solution for the coin changing problem for any monetary system. Given a set of n denominations $\{d_1, d_2, \ldots, d_n\}$ and sum s, compute the minimum number of coins that sum to s. You may assume that there is an infinite collection of coins of all denominations.

Question 6: [marks = 10]

Suppose you are given two sequences S1 of length |S1| and S2 of length |S2| on an alphabet set Σ such that space does not belong to Σ. An alignment is defined by inserting any number of spaces in S1 and S2 so that the resulting strings S1' and S2' both have the same length (with the spaces included as part of the sequence). Each character of S1' (including each space as a character) has a corresponding character (matching or non-matching) in the same position in S2'. An optimal alignment would be on that has minimum number of mismatched characters (think of a space as just another character and hence a space matches a space but does not match any of the other characters of Σ).

For example, S1 is CTATG and S2 is TTAAGC. One possible alignment is given by:

C	T	A	T			G	
	T		T	A	A	G	C

Now both S1' and S2' have length 8 and number of mismatched characters are 5. (There are mismatches in position 1, 3, 5, 6 and 8).
Devise a dynamic programming algorithm that given two sequences S1 and S2, compute an alignment of S1 and S2 that has minimum number of mismatched characters.

Example 1:

S1: CTATG
S2: TTAAGC.

C	T	A	T			G	
	T		T	A	A	G	C

Example 2:

S1: GACGGATTAG

S2: GATCGGAATAG

G	A		C	G	G	A	T	T	A	G
G	A	T	C	G	G	A	A	T	A	G

Example 3:

S1: ATGTG

S2: ACGTA

A	T	G	T	G
A	C	G	T	A