Design and Analysis of Algorithms

CS 302

Each problem has 10 marks. Submit handwritten solutions on Slate.

Q1) A person is booking a flight to bring back Pakistanis stranded in some other country due to COVID-19 pandemic. The flight has fixed passenger capacity M (M passengers can travel other than the airline crew). He wants to book as many passengers as possible for this flight without exceeding the passenger capacity of the aircraft but there is a constraint. Some people have families with them and they only want to book the flight if all family members get a seat. Every person has different number of family members with him. The number of family members with 1 person can range from 0 to k (0 means he will travel alone and k means he needs k+1 seats in order to travel). There are n person interested in booking flights and each person has different number of family members (0 to k). Give an efficient algorithm for this problem. Be sure to prove that your algorithm yields an optimal solution and analyze the time complexity.

Input to algorithm: Number of family members (0 to k) for each of the n persons interested in booking the flight.

Q2) Suppose a manager of a company wants to find out the best order of execution of different processes for production. Some processes have constraints such that a particular process should be completed before another can be started. Given all possible such constraints among particular processes, give an efficient algorithm (pseudocode) to find the best order of execution of processes which will not violate any constraint. There are n processes and each constraint is expressed as follows: (p_1, p_2) means p_1 should be completed before p_2.

Input to algorithm: n processes and constraints. For each process p_i, you are given a list of processes such that p_i should be completed before those processes.

Q3) You are given a set of substrings of DNA sequence. A DNA sequence is defined as a string of 4 alphabets (A, G, C, T) as follows:
GCAACGTTAGA….
A substring of DNA sequence is a consecutive string. For example, ACG is a substring of above DNA sequence but GGA is not substring of this sequence.
Given a new DNA sequence S_n with length n and a set of substrings K of another DNA sequence, find out if the new DNA sequence S_n can be divided into substrings of the given set K. For example:
Let S_n = GCAACGTTAGA
K = {AGA, GT, GC, AACG, TT}
S_n can be divided into following substrings from K
GC, AACG, TT, AGA
Second Example:

Let S_n = GCAGCCTGTACT
K = {AG, GT, AACG, CC}
S_n cannot be divided into substrings from K
Given a DNA sequence S_n of length n and a set K of substrings, describe an efficient algorithm that will detect whether or not the input DNA sequence S_n can be split into substrings of set K. Your algorithm should also print the substrings. Analyze the time complexity of your algorithm. You can assume that there is a function that given a substring, checks in O(1) time if the substring belongs to input set K of substrings.
Input to algorithm: A DNA sequence S_n of length n and a set K of substrings

Q4) The semester is over! You've rented a car and are ready to set out on a long drive on the Strange Highway. There are n tourist stores on the Strange Highway, numbered 1, 2, …, n and you would want to stop at these and buy some souvenirs (only one souvenir may be bought from a store and all souvenirs everywhere have the same price). You are a greedy shopper and are most interested in maximizing the total discount you get on your shoppings. You know that each store i offers a discount d_i on a souvenir. But it's a strange highway after all. It turns out that you can only buy a souvenir from a store i if you have not bought anything from the previous f_i stores. For example, if $f_6 = 3$ then you can only buy a souvenir from store 6, and get the discount d_6, if you haven't bought anything from stores 3, 4, and 5. All the d_i and f_i are known to you in advance (passed as input). You have recently learnt the DP technique in your algorithms course and wish to apply it here in order to maximize your total discount under the given constraints.

(i) I will help you by defining the optimal sub-structure. In fact, I will give you two possible definitions:

A. $D[i]$ = max total discount when store i is the last store where you buy a souvenir.
B. $D[i]$ = max total discount for the trip *till store* i whether buying at store i or not.

Option A Example

Option B Example

i	f(i)	d(i)	D(i)
1	0	2	2
2	0	4	6
3	2	2	6
4	2	5	7

Provide recurrences for D[i] in both cases A and B. Do both these recurrences produce equally efficient algorithms?

(ii) Provide complete pseudo-code of the bottom-up DP algorithm based on one of the recurrences above. The algorithm should return the optimal total discount as well as the list of stores where to buy the souvenirs to get that discount.

Q5) Suppose there are n children and n shirts. Each child has different height ($h_1, h_2, \ldots, h_n$) and each shirt has different size ($s_1, s_2, s_3, \ldots, s_n$) (suppose both are given in inches). The problem is to give each child one shirt such that the average difference between height of child and size of shirt is minimized. If child i is given shirt a_i then the goal is to minimize the following:

$$\sum_{i=1}^{n} \frac{|h_i - s_{a_i}|}{n}$$

(a) Consider the following greedy algorithm. Find the child and shirt with smallest difference in height of child and size of shirt. Assign this shirt to the child. Repeat the process until every child has a shirt. Prove of disprove that this algorithm is correct.
(b) Consider another greedy algorithm. Give the shortest child the smallest shirt, give the second shortest child the second smallest shirt, give the third shortest child the third smallest shirt, etc. Prove of disprove that this algorithm is correct.
HINT: One of the above greedy algorithms is correct and one is incorrect.

Q6) A person is traveling by air from country 1 to country k. He can take different connecting flights on his way. For each connecting flight the air fare $a_{i,j}$ from country i to j is given. The air fares are arbitrary. For example, it is possible that $a_{1,3} = 11$ and $a_{1,4} = 6$. He wants to minimize the total air fare even if he has to take many connecting flights. Describe an efficient algorithm for solving this problem and analyze its time complexity.

Input to algorithm: for each $1 \leq i < j \leq$ k the airfare of flight from country i to country j.