National University of Computer and Emerging Sciences, Lahore Campus

Course:	Design & Analysis of Algorithms	Course Code:	CS-2009
Program:	BS (Computer Science)	Semester:	Spring 2022
Duration:	60 Minutes	Total Marks:	17
Paper Date:	6-May-22	Section:	ALL
Exam:	Midterm 2	Page(s):	4
Name	*Tahir Ejaz*	~~Roll Number~~	*Solution*

Instruction/Notes: Weightage of the exam is Section Specific (i.e., for each section the weightage would be as per the announcement done in that regard).
Do NOT un-staple your exam, otherwise it might be cancelled.
Ample space is provided for rough work; NO EXTRA sheets will be provided.

Question	1	2	3	Total
Marks	NA/5	NA /5	NA /7	NA /17

Disclaimer: This mid was way too easy. Do not expect the same complexity in your mid.

Question 1: **[5] Marks**

Consider the problem of rod cutting discussed in class. Dry run dynamic programming algorithm on following input for rod length 7. Show all computations and write values in the given array C[i] $r[n]$ for memoization.

Length	1	2	3	4	5	6	7
Price	2	5	6	11	12	14	17

$$r_n = \max_{1 \le i \le n} (p_i + r_{n-i}).$$

Solution:

i n	1	2	3	4	5	6	7
C[i] $\boldsymbol{r[n]}$	**2**	**5**	**7**	**11**	**13**	**16**	**18**

$r[1] = \max_{1 \le i \le 1}(p_i + r[1-i]) = p_1 + 0 = \mathbf{2}$

$r[2] = \max_{1 \le i \le 2}(p_i + r[2-i]) = \max(p_1 + 2, p_2 + 0) = \max(2+2, 5+0) = \max(4,5) = \mathbf{5}$

$r[3] = \max_{1 \le i \le 3}(p_i + r[3-i]) = \max(p_1 + 5, p_2 + 2, p_3 + 0) = \max(7,7,6) = \mathbf{7}$

$r[4] = \max_{1 \le i \le 4}(p_i + r[4-i]) = \max(p_1 + 7, p_2 + 5, p_3 + 2, p_4 + 0) = \max(9,10,8,11) = \mathbf{11}$

$r[5] = \max_{1 \le i \le 5}(p_i + r[5-i]) = \max(p_1 + 11, p_2 + 7, p_3 + 5, p_4 + 2, p_5 + 0) = \max(13,12,11,13,12) = \mathbf{13}$

$r[6] = \max_{1 \le i \le 6}(p_i + r[6-i]) = \max(p_1 + 13, p_2 + 11, p_3 + 7, p_4 + 5, p_5 + 2, p_6 + 0) =$
$\max(15,16,13,16,14,14) = \mathbf{16}$

$r[7] = \max_{1 \le i \le 7}(p_i + r[7-i]) = \max(p_1 + 16, p_2 + 13, p_3 + 11, p_4 + 7, p_5 + 5, p_6 + 2, p_7 + 0) =$
$\max(18,18,17,18,17,16,17) = \mathbf{18}$

Question 2: **[5] Marks**

Consider the weighted interval scheduling problem. Here the input is a set of n jobs, each with a start time finish time, and reward. Our task is to schedule some of these jobs on a single machine. The output is a non-overlapping subset of the jobs, and the goal is to maximize the total reward from the jobs in the set. Consider following greedy strategy:

Sort the jobs by reward and schedule them one by one starting with the highest reward, rejecting any that overlap with jobs already scheduled.

Give counter example to prove that it does not guarantee optimal solution.

Solution:

Consider following three activities:

- Activity a, with start time 2, finish time 4 and reward 3.
- Activity b, with start time 5, finish time 8 and reward 3.
- Activity c, with start time 3, finish time 6 and reward 4.

Now if we select activity c on the basis of the given strategy (reward 4 being the highest), then we cannot select activities a and b, and the total reward would be $\boldsymbol{4}$.

However, in an optimal solution, both activity a and activity b would be selected making a total reward of $3 + 3 = \mathbf{6}$.

Question 3: **[7] Marks**

Consider the following recursive algorithm

```
P(n){
        IF (n = 1)
                RETURN 1
        ELSE
                SUM = 0
                FOR( i = 1 to n − 1 )
                        SUM = SUM + P(i) x P(n − i)
        RETURN SUM
}
```

Convert the recursive code given above into bottom up iterative dynamic programming algorithm.

Following is an example output of program for input i

i	1	2	3	4	5	6
P(i)	1	1	2	5	14	42

Solution:

```
let P[1...n] be a new array
P[1]=1
for j=2 to n
      sum=0
      for i=1 to j-1
            sum = sum + P[i]*P[j-i]
      A[j]=sum
return A[n]
```