United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Mid Term Total Marks: **20** Fall-2020

Course Code: CSI 227 Course Title: Data Structure and Algorithms II

Time: *1 hour* for answering. *Another 15 minutes* for submitting.

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

There are **FOUR questions**. **Answer all of them**. Figures in the right-hand margin indicate full marks.

1	Derive the exact-cost equation for the running-time of the algorithm at *Figure 1*. Provide both the best-case and the worst-case time-complexities in Big-Oh(O) notation. *Figure 1*	**[3+2]**
2	Using dynamic programming, solve the following instance of the 0/1 knapsack problem with knapsack capacity 7 for five items (weight, value) – (2, \$100), (4, \$60), (3, \$80), (7, \$250), (1, \$200). You must show each step in your calculation of finding maximum value and selection of items.	**[5]**
3	Suppose, A problem X can be divided into four subproblems each of size (n/4), each of the problem can be solved recursively in time T(n/4) respectively. The cost of dividing the problem and combining the results of the subproblems is $\boldsymbol{\theta}$(n). Formulate the recurrence relation and solve it using the iteration method. [Assuming that, T(1)=1]	**[5]**
4	Given the arrival and the departure times of $\boldsymbol{n}$ trains for a railway platform where each one is at the format: [arrival time, departure time) (a) Provide a greedy algorithm to find out the maximum number of trains that can use that platform without any collision. Note that, there must exist at least $\boldsymbol{M}$ minutes of safety break between the departure of one train and the arrival of a next one. [do not write pseudocode, explain your algorithm in 3-4 lines] (b) If $\boldsymbol{n = 8}$ and $\boldsymbol{M = 1}$ and the train schedules are {[8, 12), [6, 9), [11, 14), [2, 7), [1, 7), [12, 20), [7, 12), [13, 19)}, then find the maximum number of trains that can use the platform without any collision using your algorithm. Clearly write which schedules you chose.	**[2.5 +2.5]**

```
1  Algorithm1(m,n):
2      for(i = 1; i <= n; i = i+1)
3          print(i)
4      for(i = n; i >= n/2; i = i-1)
5          for(j = 1; j<m; j = j+2)
6              print(i * j)
```