Bennett University Greater Noida
Department of CSE

Subject Lab: Algorithms & Complexity **Lab Duration: 10:40-12:35**
Lab Code: ECSE202L **Max Marks: 10**

Submission Guidelines:

1. The purpose of the course is to learn how to analyse the complexity of the algorithm.
2. You are supposed to do this assignment on your own. While you may discuss the problem with other students, you are not allowed to copy any part of the code from other students or to copy from any other source. Any form of **plagiarism** will not be tolerated. If there is substantial overlap between the codes submitted by two students, both will get reduction in the course grade.
3. The assignment should be **shown to lab instructor** in the lab session and **must be submitted** on LMS by **given date**.

It should also carry the following statement:

"***I have done this assignment on my own. I have not copied any code from another student or any online source. I understand if my code is found similar to somebody else's code, my case can be sent to the Disciplinary committee of the institute for appropriate action.***"

Lab Assignment 6

Q1. Write a program in Java and return the time complexity of the following problem:

You are given N items, each has two parameters: a weight and a cost. Let's define M as the sum of the weights of all the items.Your task is to determine the most expensive cost of a knapsack, which capacity equals to 1, 2, ..., M. A cost of a knapsack equals to the sum of the costs of all the elements of the knapsack. Also, when you have a knapsack with a capacity is equal to C, then you can fill it with items, whose sum of weights is not greater than C.

Input: The first line of the input contains one integer N, denoting the number of the items. The next N lines contain two integers W and C each, denoting the weight and the cost of the corresponding item.

Output: For each capacity C ($1 \le C \le M$) of the knapsack, output a single integer - the most expensive cost for that capacity.

Constraints $3 \le N \le 100$;
$1 \le W \le 2$, for each item;
$1 \le C \le 109$, for each item.

Example

Input:

5

1 1

2 2

2 3

2 4

2 5

Output:

1 5 6 9 10 12 13 14 15

Explanations:

In the test case, M equals to 9.

For C = 1, it's optimal to choose {1} items;

For C = 2, it's optimal to choose {5} items;

For C = 3, it's optimal to choose {1, 5} items;

For C = 4, it's optimal to choose {4, 5} items;

For C = 5, it's optimal to choose {1, 4, 5} items;

For C = 6, it's optimal to choose {3, 4, 5} items;

For C = 7, it's optimal to choose {1, 3, 4, 5} items;

For C = 8, it's optimal to choose {2, 3, 4, 5} items;

For C = 9, it's optimal to choose {1, 2, 3, 4, 5} items.

Q2. Write a program in Java and return the time complexity of the following problem.You are given N items, each has two parameters: a weight and a cost. Let's define M as the sum of the weights of all the items.Your task is to determine the most expensive cost of a knapsack .Items can be broken into smaller pieces, hence thief can select fractions of items.

Input:

```
Items as (value, weight) pair
```

```
arr[] = {{60, 10}, {100, 20}, {120, 30}}

  Knapsack Capacity, W = 50;
```

Output :

```
   Maximum possible value = 240

   By taking full items of 10 kg, 20 kg and

   2/3rd of last item of 30 kg
```