Assignment 2

Instructions:

- Read the questions very carefully and write all the functions as you are instructed in the question. You should take the input following the sample input format. Your output should also match the sample outputs. Note that your code should work for all reasonable inputs, not just sample inputs.
- Adopting any unfair means will result in -100%.
- Submit the codes in ELMS. Name the files 1.cpp, 2.cpp etc. Only submit the .cpp files.

Question 1

There are **n** boxes of **n** different items in a warehouse. Each box has a label that says the name (**m_i**), total weight (**w_i**) in kg and the total value (**v_i**) in taka of that item (**i**). *All items are divisible.* Suppose, **k** thieves have come to steal from the warehouse, each with a knapsack of capacity **W_i**. Given each thief wants to maximize his/her profit, how many thieves will be needed to empty the warehouse?

Write a code to solve this problem using a **greedy algorithm**.

Sample input **n** **m_1, v_1, w_1** **...** **m_n, v_n, w_n** **k** **W_1 W_2 ... W_k**	Sample output
4 silver-dust 300 4 gold-dust 2000 8 salt 80 10 sugar 89 10 2 15 15	Taking gold-dust: 8.0 kg 2000.0 taka Taking silver-dust: 4.0 kg 300.0 taka Taking sugar: 3.0 kg 26.7 taka Thief 1 profit: 2326.7 taka Taking sugar: 7.0 kg 62.3 taka Taking salt: 8.0 kg 64.0 taka Thief 2 profit: 126.3 taka Total 2 thieves stole from the warehouse. Still following items are left salt 2.0 kg 16.0 taka
4 silver-dust 300 4 gold-dust 2000 8 salt 80 10 sugar 89 10 4 8 10 6 10	Taking gold-dust: 8.0 kg 2000.0 taka Thief 1 profit: 2000.0 taka Taking silver-dust: 4.0 kg 300.0 taka Taking sugar: 6.0 kg 53.4 taka Thief 2 profit: 353.4 taka Taking sugar: 4.0 kg 35.6 taka Taking salt: 2.0 kg 16.0 taka Thief 3 profit: 51.6 taka Taking salt: 8.0 kg 64.0 taka Thief 4 profit: 64.0 taka Total 3 thieves stole from the warehouse.

Question 2

Suppose you were to drive from A to B, which is **D** miles away, along a straight road. Your gas tank, when full, holds enough gas to travel **m** miles, and you have a map that gives distances between gas stations along the route. Let **d1** < **d2** < … < **dn** be the locations of all the gas stations along the route where **di** is the distance from St. Louis to the gas station. You can assume that the distance between neighboring gas stations is at most **m** miles. Your goal is to make as few gas stops as possible along the way. Give the most efficient algorithm you can to determine at which gas stations you should stop.

Write a code to solve this problem using a **greedy algorithm**. Keep the time complexity of your code O(n).

Sample input **D** **m** **n** **d1 d2 … dn**	Sample output
20 10 8 2 4 5 8 12 14 16 19	stop at gas station 4 (8 miles) stop at gas station 7 (16 miles)
25 10 5 2 4 5 8 12	