CS1020 Take-home Lab #5
Exercise #1: Packing Luggage

http://www.comp.nus.edu.sg/~cs1020/3_ca/takehomelabs.html

Objective:

- Solving an optimisation problem using recursion.

Recursion is the objective of this exercise. Hence, you must use recursion. If recursion is not used (or it is used but not in solving the problem but in other side quests), no attempt mark will be awarded.

Task statement:

One of the CS1020 labTAs, YanHao, wants to pack items into a luggage. Items are of different sizes and values. There is not enough space in the luggage to accommodate all of them, so he decides to pick some of them, with the goal of maximizing the total value of all the items in the luggage. If there are more than one way to pack the luggage resulting in the same maximum total value, YanHao prefers the solution with more free space in the luggage (see test case 1 below).

For this exercise we assume that an item can fit into the luggage if there is enough empty space. We do not need to worry about the shape of the item.

Write a program **Packing.java** to help YanHao to find the optimal solution.

Number of submissions:

You are given **10** submissions. Only the final submission will be graded.

Input/output format

The first line of input contains an integer which represents the total size of the luggage. From the second line onwards, each line contains 2 integers representing the size of an item and its value, separated by a space.

The output consists of a single line containing 2 integers, separated by a space, that represent the total space occupied by the items in the luggage and their total value.

Sample run #1:

```
80
50 1000
20 10
20 100
35 300
40 800
```
(User press ctrl-d to end input)
```
70 1100
```

Note: The last two items also have a value sum of 1100, but they will not be chosen because their total size is 75, whereas the total size of items 1 and 3 is 70.

Sample run #2:

```
100
50 1000
50 1000
10 400
30 400
20 200
```
(User press ctrl-d to end input)
```
100 2000
```

Note: If we define the *value density* of an item as the ratio of its value to size, then the third item has the highest value density of 400/10 or 40. However, it is not in the solution. This means that you should not just go for items with the highest value density.