Problem #1:

Suppose that you run a popular TV channel and a lot of companies want to show their commercials on your channel. But you know, these days the housewives who enjoy watching TV serials on your channel, hate it when they are forced to see a commercial. So when you are going to show a commercial you have a limited period of time for doing so (given in seconds). Suppose that you have, at your disposal a set of TV commercials from different companies each of which has a *name*, *profit* and *duration* (in seconds). Assume that each TV commercial is distinct and *profit* represents the amount of money that you can earn by showing the TV commercial for *duration* seconds. You can choose to not show a TV commercial for the entire *duration* i.e. you can show any TV commercial partially and switch to another. Your job is:

1. Find the maximum earnable profit by broadcasting the TV commercials
2. Find the names and the duration of the TV commercials that you will chose for the maximum revenue

Sample I/O:

Name	Profit	Duration
Google Commercial	100 BDT	20 seconds
Microsoft Internet Explorer Commercial	60 BDT	10 seconds
RedHat Commercial	120 BDT	30 seconds

Maximum time duration for a TV commercial is 50 seconds.
Maximum profit: 240 BDT by broadcasting the Google Commercial for 20 seconds, Microsoft Commercial for 10 seconds and RedHat Commercial for ⅔ of 30 seconds.

Problem #2:

You are given *m* arrays of integers, each of length *n*. Find the maximum sum of a contiguous subarray for each of those *m* arrays.

1. Print the maximum subarray sum, whose subarray length is minimum among all of the *m* arrays.
2. From question no. 1, suppose that you found a subarray whose length is minimum among all *m* arrays. If a cost *c* is associated with each element in the sub array, then the maximum subarray sum from question no. 1 will become smaller proportional to the size of the subarray. Print this new maximum subarray sum. In case of a tie, choose any one of them.

Sample I/O

For sample, there 3 arrays each of length 6:

[1, -2, -3, 6, 5, -1]

[-2, 3, 4, -1, 2, -3]

[1, 2, -4, 2, -1, -3]

If the cost *c=1* and the maximum subarray sums are 11 with length 2, 8 with length 4 and 3 with length 2. There is a tie here so I chose the first one(arbitrarily) with sum being 11 and length being 2. If we now adjust the sum with respect to the cost per element of the chosen subarray, the final sum is 9 i.e. 11-(2× 1)=9

Problem #3:

Suppose that you are working as an event manager in a kindergarten class party. Many children came to the party. Organize them into the minimum possible number of groups such that the age of any two children in the same group differ by at most k years, where $1 \leq k \leq 5$. Print the minimum number of groups required to group all the children. You will be given an array of real numbers $Ages$ an integer n, denoting the length of the array and the integer k. For example, consider the following input:

Sample I/O:

Given the array $Ages = [1,1.5,2,2.5,3,4,6,7,8,10]$ and k=3the minimum number of required such groups is 3 and the grouping is: [1,4]; [6,9]; [10,13]. For the first group, the children of ages 1.5, 2, 2.5 and 3 were covered by the first group. The second and third groups covered the rest of the children.