1. Consider an operating system where each process can only run up to a certain amount of time (say threshold = 4 secs). Then we switch to some other process. In this way, we iterate over all the active processes. In case there is only one active process, it can continuously execute (without considering the threshold). Now, given arrival times, execution times and the threshold, calculate the average waiting time.

Example-
P1 arrives at 0 and executes for 3 secs, P2 arrives at 0 and executes for 4 secs, and P3 arrives at 0 and executes for 5 secs. Threshold = 1sec.
Then the order of process execution is
P1 from 0 to 1.
P2 from 1 to 2.
P3 from 2 to 3.
P1 from 3 to 4.
P2 from 4 to 5.
P3 from 5 to 6.
P1 from 6 to 7.
P2 from 7 to 8.
P3 from 8 to 9.
P2 from 9 to 10.
P3 from 10 to 11.
P3 from 11 to 12.
Waiting times-
P1 – 4 secs
P2 – 6 secs
P3 – 7 secs
Average waiting time = 5.66 secs.

2. Alexa has two stacks of non-negative integers, stack **A** and stack **B** where index denotes the top of the stack. Alexa challenges Nick to play the following game:

i. In each move, Nick can remove one integer from the top of either stack **A** or stack **B**.
ii. Nick keeps a running sum of the integers he removes from the two stacks.
iii. Nick is disqualified from the game if, at any point, his running sum becomes greater than some integer **x** given at the beginning of the game.
iv. Nick's final score is the total number of integers he has removed from the two stacks.

Given **A**, **B** and **x**, find the maximum possible score Nick can achieve (i.e., the maximum number of integers he can remove without being disqualified).

Input -
The first line contains three space-separated integers describing the respective values of **n** (the number of integers in stack **A**), **m** (the number of integers in stack **B**), and **x** (the number that the sum of the integers removed from the two stacks cannot exceed).
The second line contains **n** space-separated integers describing the respective values of **A**.
The third line contains **m** space-separated integers describing the respective values of **B**.

Output
Print an integer on a new line denoting the maximum possible score Nick can achieve without being disqualified.

Example-
Input-
5 4 10
4 2 4 6 1 (**Note the left most element represents the top of the stack.**)
2 1 8 5 (**Note the left most element represents the top of the stack.**)

Output-
4

3. You are given **q** queries. Each query consists of a single number **N**. You can perform any one of the given operations in each move:

i. If we take 2 integers **a** and **b** where **N** = **a*b** then we can change **N** = *max*(**a**, **b**).
ii. Decrease the value of **N** by 1.

Determine the minimum number of moves required to reduce the value of **N** to 0.

Input
The first line contains the integer **q**.
The next **q** lines each contain an integer **N**.

Output-
Output **q** lines, each line containing the minimum number of moves required to reduce the value of **N** to 0.

Example –
Input -
2
3
4

Output –
3
3

4. Suppose there is a circle. There are **N** petrol pumps on that circle. Petrol pumps are numbered **0** to **N-1**. You have two pieces of information corresponding to each of the petrol pumps: **(i)** the amount of petrol that a particular petrol pump will give, and **(ii)** the distance from that petrol pump to the next petrol pump. Initially, you have a tank of infinite capacity carrying no petrol. You can start the tour at any of the petrol pumps. Calculate the first point from where the truck will be able to complete the circle. Consider that the truck will stop at each of the petrol pumps. The truck will move one kilometre for each litre of the petrol.

Input –
The first line will contain the value of **N**.
The next **N** lines will contain a pair of integers each, i.e. the amount of petrol that petrol pump will give and the distance between that petrol pump and the next petrol pump.

Output-
An integer which will be the smallest index of the petrol pump from which we can start the tour.

Example -
Input-
3
1 5
10 3

3 4
Output-
1

5. Rahul can't stand when two same things are placed adjacent to each other. He now has a string and wants that every adjacent pair of characters are different. Before going to begin the task he needs to know if it is at all possible with a given string by exchanging the characters of the string. Help Rahul with this task

Sample Input 1 -
bbbbb

Sample Output 1 -
Not Possible

Sample Input 2 -
aabababbb

Sample Output 2 -
Possible

6. Given a matrix city, each square **grid[i][j]** represents the height of the building at that point (**i**, **j**). Assume that you can swim from one building terrace if they are at same level.

Now rain starts to fall. At time **t**, the depth of the water everywhere is **t**. You can swim from a square to another 4-directionally adjacent square (no diagonal moves) if and only if the elevations of both squares individually are at most **t**. You can swim infinite distance in zero time. Of course, you must stay within the boundaries of the grid during your swim.

You start at the top left building (0, 0). What is the least time in which you can reach the bottom right building (**N** - 1, **N** - 1)?

Sample Input
2
0 2
1 3

Sample Output
3

Explanation:
At time 0, you are in grid location (0, 0).
You cannot go anywhere else because 4-directionally adjacent neighbours have a higher elevation at **t** = 0.
You wait for one second when water level is **t** = 1. Go to (1, 0). Wait for 2 more seconds and go to (1, 1).

7. Sam is going on a world tour. He hires you as his travel agent. He wants you to go to **t** cities, he will provide you the source and destination. He also states that he will willing to go via at maximum **k** cities between his source and destination. Find the cheapest possible flight combination.

(Hint - solve this problem using heaps)

Sample Input Format
N cities **M** flight routes
Next **M** lines would be source, destination, flight price
t – No. of cities he wants to visit

Next **t** lines would be source, destination, **k**

Sample Input
3 3
0 1 100
0 2 500
1 2 100
2
0 2 0
0 2 1

Sample Output
500
200

8. Suppose ShareChat will start its IPO soon. In order to sell a good price of its shares to Venture Capital, ShareChat would like to work on some projects to increase its capital before the IPO. Since it has limited resources, it can only finish at most **k** distinct projects before the IPO. Help ShareChat design the best way to maximize its total capital after finishing at most **k** distinct projects.

You are given several projects. For each project **i**, it has a pure profit P_i and a minimum capital of C_i is needed to start the corresponding project. Initially, you have **W** capital. When you finish a project, you will obtain its pure profit and the profit will be added to your total capital.

To sum up, pick a list of at most **k** distinct projects from given projects to maximize your final capital, and output your final maximized capital.

Sample Input format
K
W - current capital
N - number of projects
Profit array
Capital array

Sample Input
2
0
3
1 2 3
0 1 1

Sample Output
4

****