**Q:-1**

**Min. Absolute Difference In Array**

Send Feedback

Given an integer array A of size N, find and return the minimum absolute difference between any two elements in the array.

We define the absolute difference between two elements ai, and aj (where i != j ) is |ai - aj|.

**Input format :**

Line 1 : Integer N, Array Size

Line 2 : Array elements (separated by space)

**Output Format :**

Minimum difference

**Constraints :**

1 <= N <= 10^6

**Sample Input :**

5

2 9 0 4 5

**Sample Input :**

1

**Q:-2**

**Nikunj and Donuts**

Send Feedback

Nikunj loves donuts, but he also likes to stay fit. He eats n donuts in one sitting, and each donut has a calorie count, ci. After eating a donut with k calories, he must walk at least 2^j x k(where j is the number donuts he has already eaten) miles to maintain his weight.

Given the individual calorie counts for each of the n donuts, find and print a long integer denoting the minimum number of miles Nikunj must walk to maintain his weight. Note that he can eat the donuts in any order.

Input

The first line contains an integer, n, denoting the number of donuts.

The second line contains n space-separated integers describing the respective calorie counts of each donut I, i.e ci.

Output

Print a long integer denoting the minimum number of miles Nikunj must walk to maintain his weight.

Constraints

**1 ≤ n ≤ 40**

**1 ≤ ci ≤ 1000**

Sample Input

3

1 3 2

Sample Output

11

**Q:-3**

**Activity Selection**

Send Feedback

You are given n activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time.

Input

The first line of input contains one integer denoting N.

Next N lines contains two space separated integers denoting the start time and finish time for the ith activity.

Output

Output one integer, the maximum number of activities that can be performed

Constraints

**1 ≤ N ≤ 10^6**

**1 ≤ ai, di ≤ 10^9**

Sample Input

6

1 2

3 4

0 6

5 7

8 9

5 9

Sample Output

4

**Q:-4**

**Fractional Knapsack**

Send Feedback

You want to paint your house. The total area of your house is D units. There are a total of N workers. The ith worker is available after time Ti, has hiring cost Xi and speed Yi. This means he becomes available for hiring from time Ti and remains available after that. Once available, you can hire him with cost Xi, after which he will start painting the house immediately, covering exactly Yi units of house with paint per time unit. You may or may not hire a worker and can also hire or fire him at any later point of time. However, no more than 1 worker can be painting the house at a given time.

Since you want the work to be done as fast as possible, figure out a way to hire the workers, such that your house gets painted at the earliest possible time, with minimum cost to spend for hiring workers.

Note: You can hire a previously hired worker without paying him again.

Input

The first line of input contains two integers "N D", the number of workers and the area of your house respectively. The ith of the next N lines denotes the ith worker, and contains three integers "Ti Xi Yi", described in the statement.

Output

Output one integer, the minimum cost that you can spend in order to get your house painted at the earliest.

Constraints

**1 ≤ N, T, X, Y ≤ 10^5**

**1 ≤ D ≤ 10^11**

Sample Input

3 3

1 1 1

2 2 2

3 1 5

Sample Output

3

**Q:-5**

**Weighted Job Scheduling**

Send Feedback

You are given N jobs where every job is represented as:

1.Start Time

2.Finish Time

3.Profit Associated

Find the maximum profit subset of jobs such that no two jobs in the subset overlap.

Input

The first line of input contains one integer denoting N.

Next N lines contains three space separated integers denoting the start time, finish time and the profit associated with the ith job.

Output

Output one integer, the maximum profit that can be achieved.

Constraints

**1 ≤ N ≤ 10^6**

**1 ≤ ai, di, p ≤ 10^6**

Sample Input

4

3 10 20

1 2 50

6 19 100

2 100 200

Sample Output

250

**Q:-6**

**Perimeter with conditions**

Send Feedback

Aahad gives an array of integers and asks Harshit to find which three elements form a triangle (non-degenerate). The task seems easy to Harshit.

So, Aahad adds some conditions to this task -

1. Find the triangle with maximum perimeter

2. If there are two or more combinations with same value of maximum perimeter, then find the one with the longest side.

3.If there are more than one combinations which satisfy all the above conditions the find with maximum longest minimum side.

**Input Format**

The First line contains no of elements of array: N

Each T lines contains N space-separated integers: A [i]

**Output Format**

The output contains three space-separated elements that denote the length of the sides of triangle. If no such triangle is possible, then print -1.

**Constraints**

1 =< N <= 10^5

1 <= A[i] <= 10^9

Time Limit: 1 second

**Sample Input1:**

5

1 1 1 3 3

**Sample Output1:**

1 3 3

**Sample Input2:**

3

2 2 4

**Sample Output3:**

-1

**Explaination**

In the First Sample case, the elements that form a triangle with maximum perimeter is 1,3,3.

In the Second Sample case, the elements that can form a triangle are degenerate, so, we printed -1.

**Q:-7**

**Problem discussion**

Send Feedback

Harshit gave Aahad an array of size N and asked to minimize the difference between the maximum value and minimum value by modifying the array under the condition that each array element either increase or decrease by k(only once).

It seems difficult for Aahad so he asked for your help

**Input Format**

The First line contains two space-separated integers: N,K

Next lines contain N space-separated integers denoting elements of the array

**Output Format**

The output contains a single integer denoting the minimum difference between maximum value and the minimum value in the array

**Constraints**

1 =< N <= 10^5

1 <= Ai,K <= 10^9

**Sample Input1:**

3 6

1 15 10

**Sample Output1:**

5

**Explaination**

We change from 1 to 6, 15 to 9 and 10 to 4. Maximum difference is 5 (between 4 and 9). We can't get a lower difference.

**Q:-8**

**Winning Lottery**

Send Feedback

Harshit knows by his resources that this time the winning lottery number is the smallest number whose sum of the digits is S and the number of digits is D. You have to help Harshit and print the winning lottery number.

**Input Format**

The Input line contains two space-separated integers: S,D

**Output Format**

The output contains a single integer denoting the winning lottery number

**Constraints**

1 <= D <= 1000

1 <= S <= 9\*D

Time Limit: 1 second

**Sample Input1:**

9 2

**Sample Output1:**

18

**Explanation**

There are many other possible numbers like 45, 54, 90, etc with the sum of digits as 9 and number of digits as 2. The smallest of them is 18.