ACM ICPC Guangzhou Summer Series July Contest
Time Limit: 5 hours

Judge Setup:
Core 2, 1.6Ghz, 2GB ram.
7/22/2017

Problem A: Alice's Travels
Time Limit: 5 seconds

Description

Alice is a merchant in the world. Layout of this world is a tree. There is only one path between any two cities. Each city has a unique gem with a starting price. Alice will choose to buy from city A and then sell in City B. As Alice passes through each of the cities, the city's gem prices will rise. Calculate the maximum profit that Alice can earn after traveling. If a profit is not possible, Alice will stay at home. Note that the price of all gems (regardless of which one or type) in a given city is always the same.

Input

A number of of inputs (≤**20**) described as follows. Input start with **N**, the number of cities (0<**N**≤50000). The next **N** lines has the initial price **P** in each city (0<**P**≤1000). This is followed by **N**-1 line consecutively, with two numbers **x** and **y** between 1 and **N** on each line, specifying there is a road between cities **x** and **y**. The next line is an integer **Q**, the number of inquiries (0<**Q**≤50000).
Then **Q** lines, each line input three positive integer **a, b**, **v**, which means Alice travels from **a** to **b**, and the price of gem rises by **v** (0 < **v** ≤ 1000000000, 1 ≤ **a**,**b** ≤ **N**) for each city on the shortest path from **a** to **b**. Note that price increases from a query will persist for future queries on the same input set.

Output

Output for each query, Alice's maximum profit possible. Note, return 0, if Alice stays at home (no profit is possible.

Sample Input

```
3
1 2 3
1 2
2 3
2
1 2 100
1 3 100
```

Sample Output

```
1
1
```

Problem B: Brackets Everywhere
Time Limit: 5 seconds

Description

Compute the minimum number of brackets you need to insert into an arbitrary sequence of brackets, such that all brackets are matched, and the number of ways of doing so. Two ways are considered different if and only if their corresponding final matched sequences of brackets are different. Note that the empty string is a matching bracket sequence; If **S** is a matching bracket, then [**S**] is also a matching bracket; If **S**, **T** are matching bracket sequence, then **ST** is also a matching bracket sequence.

Input

A number of of inputs (≤**100**) described as follows. Each input is just a single string sequence of brackets consisting of only [and]. The maximum length of the string is **1000** characters.

Output

For each input, output a line with the minimum number of brackets to insert followed by the number of different ways modulo **1000000007**, as described above.

Sample Input

```
[]]
[[]]
```

Sample Output

```
1 2
0 1
```

Problem C: Catch the Rats
Time Limit: 5 seconds

Description

Rats are loose upon the world, each at a 2D coordinate. Bob is going to release a number of devices to catch the rates. If the device falls on the rat, the rat is caught. All rats on the segment between any 2 given devices is also considered caught. Finally, all rats that fall within the triangle formed by any 3 devices is considered caught. Calculate the minimum number of devices needed to catch all rats.

Input

A number of of inputs (≤**100**) described as follows. The first two integers **n** and **m** (0 < **n**, **m** ≤ 300).
The next **n** lines are two integers **x**, **y**, representing the coordinates of a rat.
The next **m** line is two integers **x**, **y**, that can be a coordinate of the device.
All coordinates fit into 32 bit unsigned integers.

Output

For each input, output the minimum number of devices needed on a single line.
If it is not possible to cat all rats, output -1 on a single line.

Sample Input

4 4
0 0
1 0
0 1
-1 0
0 1
1 0
0 -1
-1 0

Sample Output

3

Problem D: Do Pillars Again
Time Limit: 5 seconds

Description

Assuming that there are **N** pillars, and we need to put onto the pillars, a bunch of balls, i.e., numbered 1, 2, 3, 4, 5, …, in increasing order such that on the same pillar, the sum of the numbers of any 2 adjacent balls is a *cube* ($\mathbf{k}^3$ for positive integer **k**). Calculate the maximum number of balls that can be placed on the **N** pillars. You may put the ball on any pillar, but no balls can be skipped. The process stops once you cannot not place a ball.

Input

A number of of inputs (≤**1000**), each with **N (**0 < **N** ≤ 2000000**).**

Output

For each input, output the total number of balls on one line.

Sample Input

1
2
8

Sample Output

1
2
15

Problem E: Easy Permutation Problem
Time Limit: 5 seconds

Description

Define an alternating permutation of the set {1, 2, 3, ..., **n**} to be an arrangement of those numbers such that the permutation $\mathbf{a_1 \ldots a_n}$ satisfies ($\mathbf{a_{i-1}} < \mathbf{a_i}$ AND $\mathbf{a_i} > \mathbf{a_{i+1}}$) or ($\mathbf{a_{i-1}} > \mathbf{a_i}$ AND $\mathbf{a_i} < \mathbf{a_{i+1}}$) for all $1 < \mathbf{i} < \mathbf{n}$. In this problem, compute the number of alternating permutations for a given triple of ($\mathbf{n}$, $\mathbf{a_1}$, $\mathbf{a_n}$).

Input

A number of of inputs (≤**1500**), each line with $\mathbf{n}$, $\mathbf{a_1}$, $\mathbf{a_n}$ ($2 \le \mathbf{n} \le 2000$, $1 \le \mathbf{a_1}, \mathbf{a_n} \le \mathbf{n}$).

Output

For each input, output the total number of permutations modulo **1000000007** on one line.

Sample Input

2 1 2
4 2 3

Sample Output

1
2

Problem F: Fantastic Sorting Algorithm
Time Limit: 5 seconds

Description

Sort a sequence of integers in non-decreasing order by repeating a single operation:
Delete any of one of the numbers in the sequence and add it to an existing number. For example, to sort (3,2,2), simply take one of the 2's and add to other 2 to get (3,4) in a single operation.

Input

A number of of inputs (≤**150**), each starting with **n** on a line,
followed by a line with **n** numbers $\mathbf{a_i}$ **(**$1 \le \mathbf{n} \le 5000$, $1 \le \mathbf{a_i} \le 100000$**).**

Output

For each input, output the minimum number of operations required to sort the sequence.

Sample Input

```
3
3 2 2
5
8 2 7 3 1
```

Sample Output

```
1
3
```

Problem G: Graph Cut of Maximum XOR Weight
Time Limit: 5 seconds

Description

A cut is a partition of the vertices of a graph into two disjoint subsets. Any cut creates a cut-set, the set of edges that have one endpoint in each subset of the partition. Let **V**(cut-set) denote the XOR of all the weights on all the edges in the cut-set. In this problem you will start with an empty graph with **n** nodes. A number of weighted edges will be successively added to the graph. After the addition of each weighted edge, output the value of the maximum XOR cut, such that **V**(cut-set) is maximized!

Input

A number of of inputs (≤**100**) with the following format.
The first two integers **n, m** represent the number of points in the graph and the total number of edges to be added successively. Next, we have **m** lines, with **x**, **y**, **w** where (**x**, **y**) is the undirected the edge of weight **w**. **w** will be given in binary form listed from the highest binary bit to lowest binary bit.
Note that $1 \le \mathbf{n} \le 500$, $1 \le \mathbf{m} \le 1000$, $0 \le \text{length}(\mathbf{w}) \le 1000$, $1 \le \mathbf{x}, \mathbf{y} \le \mathbf{n}$.

Output

For each edge, output the value of the maximum XOR cut in binary form (from high bit to low bit).

Sample Input

```
3 6
1 2 11
1 2 11
3 3 1110
1 3 1011011
1 2 10111
2 3 1110110
```

Sample Output

```
11
0
0
1011011
1011011
1100001
```

Problem H: Hedgehogs Communicate
Time Limit: 5 seconds

Description

Hedgehogs communicate via complex calls. Hedgehogs with better calls can communicate a longer distance. Consider **n** Hedgehogs (working together) on the X-axis, with coordinates $\mathbf{X}_i$ for $1 \le i \le \mathbf{n}$, and communication ability $\mathbf{A}_i$, then 2 hedgehogs can communicate if and only if $|\mathbf{X}_i + \mathbf{X}_j| \le \mathbf{A}_i + \mathbf{A}_j$. Exactly **k** hedgehogs are not underground looking for food, and can currently communicate and lookout for attacking Eagles. The remaining **n-k** hedgehogs are foraging for food. The units of food each hedgehog can forage underground each day is given by $\mathbf{S}_i$. Each Hedgehog that is communicating can increase their communication ability $\mathbf{A}_i$ by **D** from consuming **D** unit of food. Compute the minimal food cost on any given day for all pairs of hedgehogs to be able to communicate directly. If there is food surplus, just print a negative integer indicating negative food cost.

Input

A number of of inputs (≤**50**), each starting with two integers **n** and **k** are given ($1 \le \mathbf{k} \le \mathbf{n} \le 100000$). On each of the following **n** lines are $\mathbf{X}_i$, $\mathbf{A}_i$, $\mathbf{S}_i$ ($1 \le \mathbf{X}_i , \mathbf{A}_i , \mathbf{S}_i \le 1000000000$).

Output

For each input, output the minimal food cost (or maximal gain).
In case of a gain, the printed number should be negative.

Sample Input

```
5 3
41 632 33
131 22 43
871 32 93
1211 62 153
1593 52 21
```

Sample Output

```
412
```

Problem I: Intersecting Semi-Circles
Time Limit: 5 seconds

Description

There are **n** points on X-axis, and their coordinates are (1,0), (2,0), ⋯, (**n**, 0).
The color of the point is (**i**, 0) is $\mathbf{a_i}$. If two points have the same color, then a semi-circle centered at their midpoint on the X-axis, connecting them, is drawn with color $\mathbf{a_i}$ in the first quadrant (this is thus the top half of the circle, with these two points on the diameter). Compute the number of intersections where 2 arcs of different colors intersect modulo **1000000007**.

Input

A number of of inputs (≤**150**), each starting with **n** on a line,
followed by a line with **n** numbers $\mathbf{a_i}$ **(**$1 \le \mathbf{n} \le 100000$, $1 \le \mathbf{a_i} \le 100000$**).**

Output

For each input, output the answer on one line.

Sample Input

1
1
8
1 2 3 1 2 3 2 1

Sample Output

0
8

Problem J: Just More Permutations
Time Limit: 5 seconds

Description

Let ⌊ **x** ⌋ be the floor of **x.** Count the number of permutations ($a_1, a_2, ..., a_n$) of (**1, 2, …, n**) such that

$$| a_1 - 1 | + | a_2 - 2 | \ldots + | a_n - n | = \lfloor n^2 / 3 \rfloor$$

Input

A number of of inputs (≤**1000**), each start with the number of value of integer **n** (1 ≤ **n** ≤ 1000000).

Output

Output the number of permutations modulo **1000000007**.

Sample Input

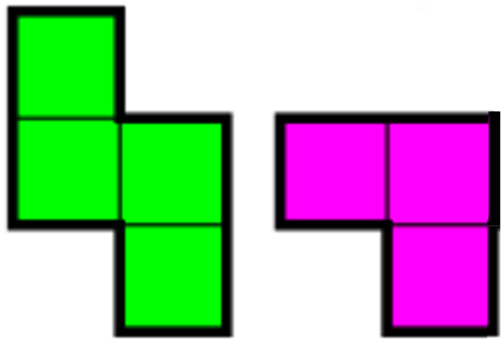
1
5

Sample Output

1
35

Problem F: Kid's Simple Puzzle Problem
Time Limit: 5 seconds

Description



Kid's are playing a tiling game. First they draw an **N**x**M** rectangle with **N** rows and **M** columns (N*M squares), then they try to cover it completely with the 2 wooden pieces shown above (left piece covers 4 squares, while right piece covers 3). Note that the pieces can be rotated or flipped. Compute the minimum number of puzzle pieces required, or output **-1** if it's not possible.

Input

A number of of inputs (≤**1000**), each starting with **n, m (**1 ≤ **n, m** ≤ 1000000000**)** on a line**.**

Output

For each input, output the minimum number of puzzle pieces, or **-1** if it's not possible.

Sample Input

```
1 1
2 3
```

Sample Output

```
-1
2
```

Problem L: Looking at Divisors
Time Limit: 5 seconds

Description

Let **d**(**n**) be the sum of all divisors of n. For example **d**(6)=1+2+3+6=12. Given integers **n** and **k**, compute the sum of all integers **m** for $1 \le$ **m** < **n**, such that **d**(**m**) is a multiple of **k**, i.e. **d**(**m**) = **l*****k**, where **l** is a positive integer.

Input

A number of of inputs (≤**100**), each start with the number of value of integers **n, k** (1≤ **n,k**≤10000000).

Output

Output the answer modulo **1000000007**.

Sample Input

10 5
20 5

Sample Output

8
27