Sereja and Sorting 2

- SUBMIT
- ALL SUBMISSIONS

Problem code: SEASORT2

Read problems statements in Mandarin Chinese and Russian.

Sereja has an array **A[1 .. N]**, which contains **N** integers. Now Sereja wants to sort it.

The only thing that Sereja can do, is to reverse all elements in some sub-array, which is a consecutive parts of **A**. In other words, in one operation, Sereja can choose two integers **L** and **R (1 <= L < R <= N)**, and swap elements **A[L]** and **A[R]**, **A[L+1]** and **A[R-1]**, **A[L+2]** and **A[R-2]** and so on.

In such operation of **L, R**, the total energy that Sereja spends **R - L + 1**. Clearly, Sereja wants to minimize the spent energy. Also, Sereja would like to minimize the total number of all operations. Therefore, we give a mixed objective as shown in Scoring.

Input

First line contains an integer **N**. Next line contains a sequence of integers **A[1], A[2], A[3], ..., A[N]**.

Output

First line contains an integer **Q** - the total number of operations you need to sort the array. In each of next**Q** lines, there should be two integers **L** and **R (1 <= L < R <= N)** indicate the operation that Sereja should do.

Also you should remember, that **Q** shouldn't be greater than **N**.

Scoring

Suppose **S** is the total spent energy of your output, i.e. the sum of **R - L + 1** of all operations. Your score is **S / N + Q**. Lower scores will earn more points.

We have 40 official test files. You must correctly solve all test files to receive OK. During the contest, your overall score is the sum of the scores on the first 10 test files. After the contest, all solutions will be rescored by the sum of the scores on the rest 10 test files. Note, that public part of the tests may not contain some border cases.

Constraints

- **1** <= **N** <= **10000**
- **1** <= **A[i]** <= **5000**

Example

Input:
6
2 1 5 4 3 2
Output:
2
3 6
1 2

Chef and Graph Queries

- [SUBMIT]
- [ALL SUBMISSIONS]

Problem code: GERALD07

Read problems statements in Mandarin Chinese and Russian.

Problem Statement

Chef has a undirected graph **G**. This graph consists of **N** vertices and **M** edges. Each vertex of the graph has an unique index from **1** to **N**, also each edge of the graph has an unique index from **1** to **M**.

Also Chef has **Q** pairs of integers: $\mathbf{L_i}$, $\mathbf{R_i}$ ($\mathbf{1 \le L_i \le R_i \le M}$). For each pair $\mathbf{L_i}$, $\mathbf{R_i}$, Chef wants to know: how many connected components will contain graph **G** if Chef erase all the edges from the graph, except the edges with indies **X**, where $\mathbf{L_i \le X \le R_i}$. Please, help Chef with these queries.

Input

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T**test cases follows.
The first line of each test case contains three integers **N**, **M**, **Q**. Each of the next **M** lines contains a pair of integers $\mathbf{V_i}$, $\mathbf{U_i}$ - the current edge of graph **G**. Each of the next **Q** lines contains a pair of integers $\mathbf{L_i}$, $\mathbf{R_i}$ - the current query.

Output

For each query of each test case print the required number of connected components.

Constraints

- $\mathbf{1 \le T \le 1000}$.
- $\mathbf{1 \le N, M, Q \le 200000}$.
- $\mathbf{1 \le U_i, V_i \le N}$.
- $\mathbf{1 \le L_i \le R_i \le M}$.
- Sum of all values of **N** for test cases is not greater than **200000**. Sum of all values of **M** for test cases is not greater than **200000**. Sum of all values of **Q** for test cases is not greater than **200000**.
- Graph **G** can contain self-loops and multiple edges.

Example

Input:
2
3 5 4
1 3

1 2
2 1
3 2
2 2
2 3
1 5
5 5
1 2
1 1 1
1 1
1 1

Output:

2
1
3
1
1

The Street

- [SUBMIT]
- [ALL SUBMISSIONS]

Problem code: STREETTA

Read problems statements in Mandarin Chinese and Russian.

The String street is known as the busiest street in Codeland.

Tourists from all over the world want to visit the street once they are in Codeland.

The Chef owns **N** souvenir stores across the street (numbered from **1** to **N**).

At the beginning there is no souvenir in any store, the Chef has some plans to add some new items.

Each the Chef's plan is represented by 4 numbers: **u v a b** which mean an items with price **b**

is added to the store **u**, an items with price **a + b** is added to the store **u + 1** and so on.

More formally, an item with price **a * i + b** is added to the store **u + i** for all (**0 ≤ i ≤ v - u**).

In additional to the cost of the item itself, the tourist must pay some conservation fees as well.

The Codeland regularly defines the new conservation fee. Each fee is represented by 4 numbers: **u v a b**which means

the tourist buying any item in the store **u + i** will be charged a fee of **i * a + b** for all (**0 ≤ i ≤ v - u**).

In the case that several conservation fees have effect on the same store, the customer needs to pay all of those fees.

At some point of time, a tourist at store **i** asks you what is the **largest** amount of money they have to spend for

a souvenir at that store (the amount of money includes the price of one of the souvenirs and all the conservation fees for that store).

Input

- The first line of the input contains two integers **N** and **M** represent the number of stores and the number of events
- Each of the next **M** lines represents an event of three types below in the chronological order.
- The new plan of the Chef: "1 **u v a b**".
- The new conservation fee: "2 **u v a b**".
- The query from tourist: "3 **i**".

Output

For each query from tourist, print in one line the corresponding answer.

If there is no item at the **i**th store, print out "NA" (without quotes) as the answer.

Constraints

- **$1 \le N \le 10^9$**
- **$1 \le M \le 3*10^5$**
- For events of type 1: **$1 \le u \le v \le N$**. **$|a|, |b| \le 10^9$**
- For events of type 2: **$1 \le u \le v \le N$**. **$|a|, |b| \le 10^4$**
- For events of type 3: **$1 \le i \le N$**

Example

Input:

```
10 10
3 5
1 3 8 3 1
3 5
1 5 10 -8 2
3 5
3 10
2 1 10 0 1
3 6
2 5 7 2 1
3 6
```

Output:

```
NA
7
7
-38
11
14
```

Explanation

- At the beginning all stores are empty so the answer for the first query which asks about the store 5 is "NA".
- The first plan of the Chef is "3 8 3 1" which means the items with price 1, 4, 7, 10, 13, 16 are added to the stores 3, 4, 5, ..., 8 correspondingly. So in the next query (asking about store 5) the answer is 7 (we have only one item to buy with no conservation fee).
- The second plan of the Chef is "5 10 -8 2" so the items with price 2, -6, -14, -22, -30, -38 are added to the stores 5, 6, 7, ..., 10 correspondingly. Now the store 5 now contains two items with the corresponding prices are 7 and 2, the answer for the query about store 5 is still 7 (we still don't have any conservation fee). The store 10 contain only one item with price is -38 so the answer for the query about this store is -38.

- The first conservation fee policy is "1 10 0 1" which cause a conservation fee of 1 for each of 10 stores. The stores 6 contains 2 items, one costs 10 unit of money and the other costs -6. We need to pay 1 unit of money for the conservation fee so the largest amount of money we can spend for one item in store 6 is 10 + 1 = 11.
- The second conservation fee policy is "5 7 2 1" so a conservation fees of 1, 3, 5 are added to the stores 5, 6 and 7 correspondingly. Hence the largest amount of money we can spend for one item in store 6 is increased by 3. The answer for the last query is 14.

Note: Both of the price and the conservation fee can be negative.