Dodgeball Diplomacy

A region consists of N cities, numbered from 1 to N. These cities may establish treaties, which indicate their willingness to collaborate with each other.

An alliance is a group of cities where each city is connected to every other city in the group through one or more treaties. The size of an alliance is the number of cities that belong to that alliance.

Each treaty has a length, representing the number of pages used to draft it. To reduce bureaucracy, cities will occasionally remove the longest active treaty (i.e., the one with the highest page count).

Beyond administrative matters, the cities also engage in dodgeball rounds between alliances. A dodgeball round followes the following rules

- Each alliance forms a team.
- Alliances are paired up to play against each other.
- If the number of alliances is odd, one team will play against an empty alliance of size 0.
- The unfairness score of a match between two alliances of sizes A and B is given by: $|A - B|$
- The total unfairness score for a round is the sum of unfairness scores across all matches.

You must determine the minimum possible total unfairness score by optimally pairing the alliances.

Queries

You must support Q queries of the following types:

- `a`: Add a treaty between cities u and v with p pages.
- `r`: Remove the longest treaty (i.e., the one with the highest page count). All page counts are unique.
- `d`: Compute the minimum unfairness score for a dodgeball round based on the current alliances.

Queries of type `d` **must be answered immediately** upon request, before processing any further queries.

Input

You should write a program with the following behavior. It will be run once for each test case. The program should read the standard input in the following format:

- line 1: N Q
- The next Q lines are each in one of the following formats:
  - `a` u v p
  - `r`
  - `d`

Corresponding to the different query operations.

Output

The program should for each query of type `d` immidiately respond before processing any additional queries. In addition you should flush the output immidiately after each answer. This can be done in one of the following ways:

- C++: `std::cout << std::endl;`
- Python: `print("",flush=True)`

Both of which also adds a newline to the input.

Constraints

- $1 \leq N \leq 100000$.
- $1 \leq Q \leq 500000$.
- $1 \leq p \leq 10^9$.
- $1 \leq u \leq N$.
- $1 \leq v \leq N$.
- $u \neq v$ for each query of type `a` query.
- Whenever a treaty is added between u and v, no treaty is in effect between u and v right before.
- All page counts p are unique.

Subtasks

1. (9 points) $N \leq 10, Q \leq 20$,
2. (10 points) $N \leq 2000, Q \leq 4000$,
3. (6 points) There are at most 10 `d` queries.
4. (17 points) For each `a` query, $u + 1 = v$,
5. (14 points) The treaties are created in order of increasing page count.
6. (26 points) The treaties are created in order of decreasing page count.
7. (18 points) *No additional constraints.*

Example 1

Input:

```
3 5
a 1 2 1
a 2 3 2
d
r
d
```

Output:

```
3
1
```

Explanation:

When the first dodgeball tournament is played, city 1, 2 and 3 are all in aliance. So they play against an empty alliance for an unfairness score of $|0-3| = 3$. Then the treaty between city 2 and city 3 is removed. So the dodgeball tournament is between the alliance of city 1 and city 2 against the alliance of city 3. This gives the unfairness score of $|2 - 1| = 1$.

Example 2

Input:

```
6 10
a 2 3 10
a 1 2 5
a 3 4 8
d
r
d
a 4 5 1
a 3 6 7
r
d
```

Output:

```
4
0
2
```

Explanation

For the first dodgeball round, there are one alliance of size 4 and two alliances of size 1. This gives a total unfairness score of 4.

For the second dodgeball round, there are two alliances of size 2 and two alliances of size 1, with a total unfairness score of 0.

For the third dodgeball round, there are three alliance of size 2, with a total unfiarness score of 2.