The road network in a country consists of cities, represented by points in a coordinate system, and roads, represented by line segments that connect individual pairs of cities.

It is allowed for the roads to intersect, but in that case overpasses and underpasses are built and, additionally, it is possible to cross over from one road to another only in the two cities which that road connects. Therefore, if there is a road that connects cities A and B, then it can be only used to travel from A to B and vice versa, even if the road intersects with other roads or passses through other cities. The length of the road is the length of the line segment used for describing it. In other words, the length is the Euclidean distance of two points which that road connects.

In the beginning, the road network consists of only **two cities connected by road** and grows as time passes so that in each step **exactly one new city** is added and is connected by **exactly two new roads** with **two existing cities** which have to be already **directly connected by road**.

Write a programme that will simulate the growth of the road network and find answers to queries about the shortest path between two arbitrary cities. More specifically, your programme must support the following commands:

- d X Y A B - A new city is added, its coordinates being (X, Y), and is connected by two new roads with cities A and B for which it holds that they are directly connected by road in that moment.
- u A B - The road distance is required (length of the shortest path) between cities A and B.

The cities are marked respectively with integers starting from 1. The location of cities 1 and 2 are given in the input data and each new city being added is marked with the following integer. Cities 1 and 2 are also connected by road.

INPUT DATA

The first line of input contains integers X_1, Y_1 ($0 \leqslant X_1, Y_1 \leqslant 10^6$) – coordinates of city 1.
The second line of input contains integers X_2, Y_2 ($0 \leqslant X_2, Y_2 \leqslant 10^6$) – coordinates of city 2.
The third line of input contains the integer N ($1 \leqslant N \leqslant 10^5$) – number of commands.

Each of the following N lines consists of exactly one command.

The command can be also in the form 'd X Y A B' where X and Y are integers, coordinates of the city being added ($0 \leqslant X, Y \leqslant 10^6$), and A and B different integers less than or equal to the current number of cities, the labels of cities with which the current city is being connected to directly by road. The cities A and B will always already be directly connected by road.

The command can also be in the form 'u A B' where A and B are different integers less than or equal to the current number of cities so far, the labels of cities that we want to know the distance between in that moment.

No two cities will have the same coordinates.

OUTPUT DATA

For each command of the type 'u' from the input data, you must output one integer – the distance between required cities. The answers to individual queries must be printed in the order of the given queries in the input data.

The solution is considered correct if the absolute error of each printed value compared to the official solution is less than or equal to 0.1. More specifically, if your solution outputs an integer R for each query and the official solution outputs the integer S, then the value is considered correct if it holds that $|R - S| \leqslant 0.1$.

Please note: We advise you to use a floating-point data type sized at least 64 bits, for example `double` (C/C++) or `Double` (Pascal).

SCORING

In test cases worth 21 points total, it will hold $N \leqslant 10^3$.

In test cases worth an additional 24 points, the city A in all commands of the type 'u' is going to be the same.

SAMPLE TESTS

input

```
6 4
10 4
9
d 6 7 2 1
u 1 2
u 3 2
d 10 2 1 2
u 3 4
d 12 7 2 4
u 5 3
u 4 5
u 1 5
```

output

```
4.000000
5.000000
7.000000
8.605551
5.385165
7.605551
```

input

```
1 1
3 8
10
d 8 2 1 2
u 2 1
d 2 9 1 3
u 1 4
d 4 7 3 4
u 4 3
d 6 1 5 4
u 6 5
d 0 0 4 6
u 7 3
```

output

```
7.280110
8.062258
9.219544
6.324555
18.439089
```