Decision Tree Clustering
Problem ID: decisiontreeclusters
Time limit: 5 seconds

Given a collection of points P in the plane, your goal is to cluster them using a simple-to-follow algorithm that any human can easily execute. That is, the clustering will be performed by a *decision tree* $\mathcal{T}$, which is a rooted tree with the following properties:

- $\mathcal{T}$ is a binary tree, each non-leaf node has precisely two children which we denote as the *left* child and the right child.
- Each non-leaf node comes with a statement of the form $x \leq C$ or $y \leq C$ where C is some given value.

Using this decision tree, we cluster the points in P as follows: each $p_i \in P$ starts at the root of $\mathcal{T}$. The corresponding statement $x \leq C$ or $y \leq C$ is then evaluated using the x- or y-value of the point p_i. If the statement is true, we move to the left child of this node, otherwise we move to the right child. In this way, all points in P are partitioned between the leaves of $\mathcal{T}$.

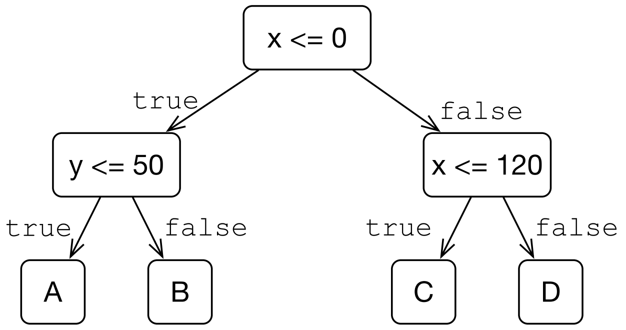
On one hand, we want the clustering to be *good*, i.e. to put similar points in a group together. On the other hand, it can be a bit expensive to evaluate the statements at non-leaf nodes. Each point $p_i \in P$ comes with an *evaluation cost* c_i which is the cost of evaluating a statement at a non-leaf node for point p_i.

If point p_i is evaluated k_i times in the decision tree before it reaches a leaf, its total evaluation cost is then $c_i \cdot k_i$. For each leaf node ℓ, the cost of the corresponding "cluster" is the boundary length of the smallest axis-parallel bounding box of the points in that cluster: call this value $b(\ell)$.

Your job is to design a decision tree $\mathcal{T}$ for the given point set P to minimize:

$$\sum_{p_i \in P} c_i \cdot k_i + \sum_{\ell\text{: leaf of } \mathcal{T}} b(\ell).$$

For example, the last sample input below is solved optimally by the following decision tree:



Each point will pass through two decision tree nodes (i.e. is evaluated twice) as it proceeds down to a leaf so in total they pay $2 \cdot 30 + 2 \cdot 30 + 2 \cdot 30 + 2 \cdot 30 + 2 \cdot 60 + 2 \cdot 60 = 480$ toward the cost of the clustering. The four leaves correspond to the partitioning:

$$A = \{(-115, 0)\}, B = \{(-109, 119), (-119, 109)\}, C = \{(100, 111), (110, 100)\}, D = \{(200, 105)\}$$

The perimiters of the smallest axis-parallel bounding boxes of these four clusters are, respectively, $0, 40, 42, 0$. So the total cost of this decision tree is $480 + 40 + 42 = 562$.

Input

The first line of input contains a single integer N ($1 \leq N \leq 40$) indicating the number of points in the set P. The next N lines describe the points in P where the ith such line is for point $p_i \in P$ and is given by three values x_i, y_i, c_i (all between -10^9 and 10^9, inclusive). No two points have the same x-value and no two points have the same y-value.

Output

Output a single integer denoting the minimum possible cost of a decision tree to cluster the points.

Sample Input 1

```
4
0 0 1
1 1 1
2 2 1
10 10 1
```

Sample Output 1

```
8
```

Sample Input 2

```
4
0 0 100
1 1 1
2 2 1
10 10 1
```

Sample Output 2

```
40
```

Sample Input 3

```
7
0 0 1
1 1 1
2 2 1
3 3 1
4 4 1
5 5 1
6 6 1
```

Sample Output 3

```
20
```

Sample Input 4

```
2
-2 1 6
2 -1 5
```

Sample Output 4

```
11
```

Sample Input 5

```
6
100 111 30
110 100 30
-109 119 30
-119 109 30
-115 0 60
200 105 60
```

Sample Output 5

```
562
```

Max Cut Min Flow

Problem ID: maxcutminflow

Time limit: 3 seconds

On opposite day, you try to prove some theorems that are the opposite of known theorems. Today, you try to prove the "max cut / min flow" theorem.

Well, min flow is simple: it's always 0.

What about max cut? An s, t cut is just a subset of nodes C containing s but not t. The value of the cut C is the number of edges having precisely one endpont in C.

Oh, your algorithms textbook says that is NP-hard to compute a maximum s, t cut? Nevermind, opposite day theorems don't seem to work.

Anyway, you liked the max cut problem so much you want to solve it on simpler graphs. At first you thought of trees, but then you decided they are far too simple. You will solve the max cut problem on trees with one extra edge.

Given a tree with an extra edge G and given distinct nodes s, t in the graph, what is the maximum possible value of an s, t cut?

Input

The first line of input contains a single integer $3 \leq N \leq 100,000$ indicating the number of vertices in G. The following N lines each contain two integers u and v ($1 \leq u, v \leq N, u \neq v$) indicating there is an edge connecting u and v in G.

It is guaranteed that all edges are distinct and the resulting graph is connected.

The last line of input contains two integers s and t ($1 \leq s, t \leq N, s \neq t$), giving the vertices you want to find a max cut for.

Output

For each test case, output a single integer giving the maximum possible value of an s, t cut.

Sample Input 1

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

Sample Output 1

```
2
```

Sample Input 2

```
6
1 2
2 3
3 4
4 1
5 1
6 2
5 6
```

Sample Output 2

```
6
```

Sample Input 3

```
6
1 2
2 3
3 4
4 1
5 1
6 2
1 6
```

Sample Output 3

```
5
```

Sample Input 4

```
5
1 2
2 3
3 1
4 1
5 2
4 5
```

Sample Output 4

```
4
```

Help! My Pizza Is Too Big!

Problem ID: pizzatoobig

Time limit: 5 seconds

Due to a clerical error, my local pizza place sent me a 14’ diameter pizza instead of a 14” pizza. They were kind enough to not charge me for the difference, but I’ve still got to figure out if I can fit it in my dining room! Can you help me out?

My local pizza place specializes in atomically-thin pizzas, so you can assume the pizza has no depth. Given three dimensions of my cuboid dining room, can you tell me the largest pizza that would fit?

Input

The first line of input contains a single integer $1 \leq N \leq 1000$ indicating the number of test cases. The next N lines each contain three integers l, w, and h on a single line, where $1 \leq l \leq 100000000$, $1 \leq w \leq 100000000$, and, $1 \leq h \leq 100000000$ are the length, width, and height of the dining room.

Output

For each test case, output a single line with the diameter of the largest pizza that would fit in the dining room. Your answer will be considered correct if it is within an absolute or relative error of 10^{-6}.

Sample Input 1

```
2
3 4 5
154 148 96
```

Sample Output 1

```
5.000000000000
155.8301164052
```

Power Curve
Problem ID: powercurve
Time limit: 5 seconds

A new rouge-like was just released–the goal is to survive for S seconds. The game consists of many unique items that give the player P additional power while taking T time to equip, as well as various enemies that gain strength at a constant rate of E power each second.

Although the player can defend themselves while equipping an item, they will not gain any power until the item is fully equipped. The player can equip any combination of items they desire, but at any point in time only one item may be in the process of being equipped.

Despite the playerbase expressing much joy, the developers are worried their game may be too hard! In an attempt to make the game easier, the player now begins with an equipped item giving exactly 50 power. To ensure this buff was sufficient, you must check if there exists some path through each game that is 'easy'.

An 'easy' path is a sequence of items such that if the player equips items in sequence order, their power level never falls below the enemy power curve!

If an 'easy' path exists, output 'EASY'; otherwise, output 'BUFF'.

Input

The first line of input contains three integers $N (1 \leq N \leq 100,000)$, the amount of items, $E (1 \leq E \leq 50)$, the enemies rate of power growth, and $S (1 \leq S \leq 10,000,000)$, the amount of seconds that must pass to win.

Each of the remaining N lines describe a single item. An item is specified by two integers $P, T (1 \leq P, T \leq 1000)$, indicating the power an item will give once equiped, and the time it takes to equip respectively.

Output

Output a single string, either 'EASY' or 'BUFF', depending on if there exists an 'easy' path through the game.

Sample Input 1

```
4 3 34
35 12
30 9
25 13
50 20
```

Sample Output 1

```
EASY
```

Sample Input 2

```
5 5 25
21 10
31 6
25 12
17 7
26 8
```

Sample Output 2

```
BUFF
```

The Cost is Correct

Problem ID: thecostiscorrect

Time limit: 3 seconds

You have won a spot on the famous game show “The Cost is Correct”. On this show, you are presented with a series of briefcases in a known order. For the i’th briefcase, there is some promised amount of money v_i in it. But the briefcase only has that amount of money with probability $p_i \in [0, 1]$. With probability $1 - p_i$, the briefcase is completely empty.

When you are presented with a briefcase, you are allowed to see how much money is inside (i.e. if it is empty or not). After this, you must either choose to take the money, or discard the briefcase and continue playing. If you take the money, the game is over and you win that amount of money. If you discard the briefcase, the game continues and you may not later return to the discarded briefcase.

If you do not accept any briefcase after seeing all of them, you win \$0.

You know the order you will be presented the briefcases in along with their v_i and p_i values. You wish to find the expected amount of money you will earn if you play perfectly.

Input

The first line of input contains a single integer N ($1 \leq N \leq 100,000$) indicating the number of briefcases. The next N lines each contain one integer and one real number. The i’th of which contains v_i and p_i describing the i’th suitcase you will see ($1 \leq v_i \leq 100,000$, $0 \leq p_i \leq 1$).

All real numbers are given with exactly 6 digits of precision after the decimal.

Output

Output a single floating-point value indicating the maximum expected value you can gain if you play perfectly. Your answer will be considered correct if it is within an absolute or relative error of 10^{-6} from the correct answer.

Sample Input 1

```
3
1 0.900000
2 0.500000
3 0.100000
```

Sample Output 1

```
1.1500000000
```

Sample Input 2

```
3
1 0.500000
1 0.500000
1 0.500000
```

Sample Output 2

```
0.8750000000
```

Sample Input 3

```
2
1 1.000000
1000 0.001000
```

Sample Output 3

```
1.0000000000
```

Sample Input 4

```
2
1000 0.001000
1 1.000000
```

Sample Output 4

```
1.9990000000
```

Treehouse
Problem ID: treehouse
Time limit: 5 seconds

In Alice's backyard, there is a large tree with some bridges (at most 10) built between nodes of the tree. Alice can get around the tree either by climbing along branches or by walking across bridges. To help her navigate around the tree quickly, she would like to know what the lengths of the shortest paths are between some pairs of nodes on the tree. Can you help her out?

Input

The first line of input contains three integers M $(1 \leq M \leq 10^5)$ the number of branches, K $(1 \leq K \leq 10)$, the number of bridges, and Q $(1 \leq Q \leq 10^5)$, the number of shortest path queries.

The next M lines each contain two integers s and t $(1 \leq s < t \leq M+1)$ indicating a *branch* between nodes s and t.

The next K lines each contain two integers s and t $(1 \leq s < t \leq M+1)$ indicating a *bridge* between nodes s and t.

The next Q lines each contain two integers s and t $(1 \leq s < t \leq M+1)$ indicating that Alice would like to know the shortest path between nodes s and t (utilizing branches, bridges, or both).

It is guaranteed that between any two nodes there is at most one branch or bridge. Furthermore, the branches form a tree, i.e., if we ignore the bridges and use only branches to get around, there is exactly one path between any two nodes.

Output

For each of the Q queries (in the same order they appear in the input), output a single integer indicating the total length of the shortest path between the endpoint nodes (s and t) of the query. The length of a path is the total number of branches and bridges that it crosses.

Sample Input 1

```
3 1 2
1 2
2 3
2 4
3 4
3 4
1 4
```

Sample Output 1

```
1
2
```

Sample Input 2

```
6 2 4
1 2
2 3
1 4
4 5
1 6
6 7
3 5
5 7
1 5
3 7
2 6
2 7
```

Sample Output 2

```
2
2
2
3
```