# Codeforces Round #378 (Div. 2)

## A. Grasshopper And the String

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

One day, the Grasshopper was jumping on the lawn and found a piece of paper with a string. Grasshopper became interested what is the minimum *jump ability* he should have in order to be able to reach the far end of the string, jumping only on vowels of the English alphabet. *Jump ability* is the maximum possible length of his jump.

Formally, consider that at the begginning the Grasshopper is located directly in front of the leftmost character of the string. His goal is to reach the position right after the rightmost character of the string. In one jump the Grasshopper could jump to the right any distance from $1$ to the value of his *jump ability*.

The picture corresponds to the first example.

The following letters are vowels: 'A', 'E', 'I', 'O', 'U' and 'Y'.

### Input
The first line contains non-empty string consisting of capital English letters. It is guaranteed that the length of the string does not exceed 100.

### Output
Print single integer $a$ — the minimum *jump ability* of the Grasshopper (in the number of symbols) that is needed to overcome the given string, jumping only on vowels.

**Examples**

| input |
| --- |
| ABABBBACFEYUKOTT |
| **output** |
| 4 |

| input |
| --- |
| AAA |
| **output** |
| 1 |

# B. Parade

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Very soon there will be a parade of victory over alien invaders in Berland. Unfortunately, all soldiers died in the war and now the army consists of entirely new recruits, many of whom do not even know from which leg they should begin to march. The civilian population also poorly understands from which leg recruits begin to march, so it is only important how many soldiers march in step.

There will be $n$ columns participating in the parade, the $i$-th column consists of $l_i$ soldiers, who start to march from left leg, and $r_i$ soldiers, who start to march from right leg.

The beauty of the parade is calculated by the following formula: if $L$ is the total number of soldiers on the parade who start to march from the left leg, and $R$ is the total number of soldiers on the parade who start to march from the right leg, so the beauty will equal $|L - R|$.

**No more than once** you can choose one column and tell **all the soldiers** in this column to switch starting leg, i.e. everyone in this columns who starts the march from left leg will now start it from right leg, and vice versa. Formally, you can pick no more than one index $i$ and swap values $l_i$ and $r_i$.

Find the index of the column, such that switching the starting leg for soldiers in it will maximize the the beauty of the parade, or determine, that no such operation can increase the current beauty.

### Input

The first line contains single integer $n$ ($1 \le n \le 10^5$) — the number of columns.

The next $n$ lines contain the pairs of integers $l_i$ and $r_i$ ($1 \le l_i, r_i \le 500$) — the number of soldiers in the $i$-th column which start to march from the left or the right leg respectively.

### Output

Print single integer $k$ — the number of the column in which soldiers need to change the leg from which they start to march, or $0$ if the maximum beauty is already reached.

Consider that columns are numbered from 1 to $n$ in the order they are given in the input data.

If there are several answers, print any of them.

**Examples**

| input |
| --- |
| 3<br>5 6<br>8 9<br>10 3 |
| **output** |
| 3 |

| input |
| --- |
| 2<br>6 5<br>5 6 |
| **output** |
| 1 |

| input |
| --- |
| 6<br>5 9<br>1 3<br>4 8<br>4 5<br>23 54<br>12 32 |
| **output** |
| 0 |

### Note

In the first example if you don't give the order to change the leg, the number of soldiers, who start to march from the left leg, would equal $5 + 8 + 10 = 23$, and from the right leg — $6 + 9 + 3 = 18$. In this case the beauty of the parade will equal $|23 - 18| = 5$.

If you give the order to change the leg to the third column, so the number of soldiers, who march from the left leg, will equal $5 + 8 + 3 = 16$, and who march from the right leg — $6 + 9 + 10 = 25$. In this case the beauty equals $|16 - 25| = 9$.

It is impossible to reach greater beauty by giving another orders. Thus, the maximum beauty that can be achieved is 9.

# C. Epidemic in Monstropolis

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There was an epidemic in Monstropolis and all monsters became sick. To recover, all monsters lined up in queue for an appointment to the only doctor in the city.

Soon, monsters became hungry and began to eat each other.

One monster can eat other monster if its weight is **strictly greater** than the weight of the monster being eaten, and they stand in the queue next to each other. Monsters eat each other instantly. There are no monsters which are being eaten at the same moment. After the monster $A$ eats the monster $B$, the weight of the monster $A$ increases by the weight of the eaten monster $B$. In result of such eating the length of the queue decreases by one, all monsters after the eaten one step forward so that there is no empty places in the queue again. A monster can eat several monsters one after another. Initially there were $n$ monsters in the queue, the $i$-th of which had weight $a_i$.

For example, if weights are $[1, 2, 2, 2, 1, 2]$ (in order of queue, monsters are numbered from $1$ to $6$ from left to right) then some of the options are:

1. the first monster can't eat the second monster because $a_1 = 1$ is not greater than $a_2 = 2$;
2. the second monster can't eat the third monster because $a_2 = 2$ is not greater than $a_3 = 2$;
3. the second monster can't eat the fifth monster because they are not neighbors;
4. the second monster can eat the first monster, the queue will be transformed to $[3, 2, 2, 1, 2]$.

After some time, someone said a good joke and all monsters recovered. At that moment there were $k$ ($k \le n$) monsters in the queue, the $j$-th of which had weight $b_j$. Both sequences ($a$ and $b$) contain the weights of the monsters in the order from the first to the last.

You are required to provide one of the possible orders of eating monsters which led to the current queue, or to determine that this could not happen. Assume that the doctor didn't make any appointments while monsters were eating each other.

## Input

The first line contains single integer $n$ ($1 \le n \le 500$) — the number of monsters in the initial queue.

The second line contains $n$ integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^6$) — the initial weights of the monsters.

The third line contains single integer $k$ ($1 \le k \le n$) — the number of monsters in the queue after the joke.

The fourth line contains $k$ integers $b_1, b_2, ..., b_k$ ($1 \le b_j \le 5 \cdot 10^8$) — the weights of the monsters after the joke.

Monsters are listed in the order from the beginning of the queue to the end.

## Output

In case if no actions could lead to the final queue, print "NO" (without quotes) in the only line.

Otherwise print "YES" (without quotes) in the first line. In the next $n - k$ lines print actions in the chronological order. In each line print $x$ — the index number of the monster in the current queue which eats and, separated by space, the symbol '`L`' if the monster which stays the $x$-th in the queue eats the monster in front of him, or '`R`' if the monster which stays the $x$-th in the queue eats the monster behind him. After each eating the queue is enumerated again.

When one monster eats another the queue decreases. If there are several answers, print any of them.

**Examples**

| input |
| --- |
| 6<br>1 2 2 2 1 2<br>2<br>5 5 |

| output |
| --- |
| YES<br>2 L<br>1 R<br>4 L<br>3 L |

| input |
| --- |
| 5<br>1 2 3 4 5<br>1<br>15 |

| output |
| --- |
| YES<br>5 L<br>4 L<br>3 L |

2 L

| input |
| --- |
| 5<br>1 1 1 3 3<br>3<br>2 1 6 |
| **output** |
| NO |

### Note

In the first example, initially there were $n = 6$ monsters, their weights are $[1, 2, 2, 2, 1, 2]$ (in order of queue from the first monster to the last monster). The final queue should be $[5, 5]$. The following sequence of eatings leads to the final queue:

- the second monster eats the monster to the left (i.e. the first monster), queue becomes $[3, 2, 2, 1, 2]$;
- the first monster (note, it was the second on the previous step) eats the monster to the right (i.e. the second monster), queue becomes $[5, 2, 1, 2]$;
- the fourth monster eats the mosnter to the left (i.e. the third monster), queue becomes $[5, 2, 3]$;
- the finally, the third monster eats the monster to the left (i.e. the second monster), queue becomes $[5, 5]$.

Note that for each step the output contains numbers of the monsters in their current order in the queue.

# D. Kostya the Sculptor

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Kostya is a genial sculptor, he has an idea: to carve a marble sculpture in the shape of a sphere. Kostya has a friend Zahar who works at a career. Zahar knows about Kostya's idea and wants to present him a rectangular parallelepiped of marble from which he can carve the sphere.

Zahar has $n$ stones which are rectangular parallelepipeds. The edges sizes of the $i$-th of them are $a_i$, $b_i$ and $c_i$. He can take **no more than two stones** and present them to Kostya.

If Zahar takes two stones, he should glue them together on one of the faces in order to get a new piece of rectangular parallelepiped of marble. Thus, it is possible to glue a pair of stones together if and only if two faces on which they are glued together match as rectangles. In such gluing it is allowed to rotate and flip the stones in any way.

Help Zahar choose such a present so that Kostya can carve a sphere of the maximum possible volume and present it to Zahar.

## Input

The first line contains the integer $n$ ($1 \le n \le 10^5$).

$n$ lines follow, in the $i$-th of which there are three integers $a_i$, $b_i$ and $c_i$ ($1 \le a_i, b_i, c_i \le 10^9$) — the lengths of edges of the $i$-th stone. Note, that two stones may have exactly the same sizes, but they still will be considered two different stones.

## Output

In the first line print $k$ ($1 \le k \le 2$) the number of stones which Zahar has chosen. In the second line print $k$ distinct integers from $1$ to $n$ — the numbers of stones which Zahar needs to choose. Consider that stones are numbered from $1$ to $n$ in the order as they are given in the input data.

You can print the stones in arbitrary order. If there are several answers print any of them.

**Examples**

| input |
| --- |
| 6<br>5 5 5<br>3 2 4<br>1 4 1<br>2 1 3<br>3 2 4<br>3 3 4 |
| **output** |
| 1<br>1 |

| input |
| --- |
| 7<br>10 7 8<br>5 10 3<br>4 2 6<br>5 5 5<br>10 2 8<br>4 2 1<br>7 7 7 |
| **output** |
| 2<br>1 5 |

## Note

In the first example we can connect the pairs of stones:

- $2$ and $4$, the size of the parallelepiped: $3 \times 2 \times 5$, the radius of the inscribed sphere $1$
- $2$ and $5$, the size of the parallelepiped: $3 \times 2 \times 8$ or $6 \times 2 \times 4$ or $3 \times 4 \times 4$, the radius of the inscribed sphere $1$, or $1$, or $1.5$ respectively.
- $2$ and $6$, the size of the parallelepiped: $3 \times 5 \times 4$, the radius of the inscribed sphere $1.5$
- $4$ and $5$, the size of the parallelepiped: $3 \times 2 \times 5$, the radius of the inscribed sphere $1$
- $5$ and $6$, the size of the parallelepiped: $3 \times 4 \times 5$, the radius of the inscribed sphere $1.5$

Or take only one stone:

- **$1$ the size of the parallelepiped: $5 \times 5 \times 5$, the radius of the inscribed sphere $2.5$**
- $2$ the size of the parallelepiped: $3 \times 2 \times 4$, the radius of the inscribed sphere $1$
- $3$ the size of the parallelepiped: $1 \times 4 \times 1$, the radius of the inscribed sphere $0.5$
- $4$ the size of the parallelepiped: $2 \times 1 \times 3$, the radius of the inscribed sphere $0.5$
- $5$ the size of the parallelepiped: $3 \times 2 \times 4$, the radius of the inscribed sphere $1$
- $6$ the size of the parallelepiped: $3 \times 3 \times 4$, the radius of the inscribed sphere $1.5$

It is most profitable to take only the first stone.

# E. Sleep in Class

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The academic year has just begun, but lessons and olympiads have already occupied all the free time. It is not a surprise that today Olga fell asleep on the Literature. She had a dream in which she was on a stairs.

The stairs consists of $n$ steps. The steps are numbered from bottom to top, it means that the lowest step has number $1$, and the highest step has number $n$. Above each of them there is a pointer with the direction (up or down) Olga should move from this step. As soon as Olga goes to the next step, the direction of the pointer (above the step she leaves) changes. It means that the direction "up" changes to "down", the direction "down" — to the direction "up".

Olga always moves to the next step in the direction which is shown on the pointer above the step.

If Olga moves beyond the stairs, she will fall and wake up. Moving beyond the stairs is a moving down from the first step or moving up from the last one (it means the $n$-th) step.

In one second Olga moves one step up or down according to the direction of the pointer which is located above the step on which Olga had been at the beginning of the second.

For each step find the duration of the dream if Olga was at this step at the beginning of the dream.

Olga's fall also takes one second, so if she was on the first step and went down, she would wake up in the next second.

## Input

The first line contains single integer $n$ ($1 \le n \le 10^6$) — the number of steps on the stairs.

The second line contains a string $s$ with the length $n$ — it denotes the initial direction of pointers on the stairs. The $i$-th character of string $s$ denotes the direction of the pointer above $i$-th step, and is either '`U`' (it means that this pointer is directed up), or '`D`' (it means this pointed is directed down).

The pointers are given in order from bottom to top.

## Output

Print $n$ numbers, the $i$-th of which is equal either to the duration of Olga's dream or to $-1$ if Olga never goes beyond the stairs, if in the beginning of sleep she was on the $i$-th step.

**Examples**

| input |
| --- |
| 3<br>UUD |
| **output** |
| 5 6 3 |

| input |
| --- |
| 10<br>UUDUDUUDDU |
| **output** |
| 5 12 23 34 36 27 18 11 6 1 |

# F. Drivers Dissatisfaction

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In one kingdom there are $n$ cities and $m$ two-way roads. Each road connects a pair of cities, and for each road we know the level of drivers dissatisfaction — the value $w_i$.

For each road we know the value $c_i$ — how many lamziks we should spend to reduce the level of dissatisfaction with this road by one. Thus, to reduce the dissatisfaction with the $i$-th road by $k$, we should spend $k \cdot c_i$ lamziks. And **it is allowed for the dissatisfaction to become zero or even negative**.

In accordance with the king's order, we need to choose $n$ - $1$ roads and make them the *main roads*. An important condition must hold: it should be possible to travel from any city to any other by the *main roads*.

The road ministry has a budget of $S$ lamziks for the reform. The ministry is going to spend this budget for repair of some roads (to reduce the dissatisfaction with them), and then to choose the $n$ - $1$ *main roads*.

Help to spend the budget in such a way and then to choose the main roads so that the total dissatisfaction with the *main roads* will be as small as possible. The dissatisfaction with some roads can become negative. It is not necessary to spend whole budget $S$.

It is guaranteed that it is possible to travel from any city to any other using existing roads. Each road in the kingdom is a two-way road.

## Input

The first line contains two integers $n$ and $m$ ($2 \le n \le 2 \cdot 10^5$, $n$ - $1 \le m \le 2 \cdot 10^5$) — the number of cities and the number of roads in the kingdom, respectively.

The second line contains $m$ integers $w_1, w_2, ..., w_m$ ($1 \le w_i \le 10^9$), where $w_i$ is the drivers dissatisfaction with the $i$-th road.

The third line contains $m$ integers $c_1, c_2, ..., c_m$ ($1 \le c_i \le 10^9$), where $c_i$ is the cost (in lamziks) of reducing the dissatisfaction with the $i$-th road by one.

The next $m$ lines contain the description of the roads. The $i$-th of this lines contain a pair of integers $a_i$ and $b_i$ ($1 \le a_i, b_i \le n$, $a_i \ne b_i$) which mean that the $i$-th road connects cities $a_i$ and $b_i$. All roads are two-way oriented so it is possible to move by the $i$-th road from $a_i$ to $b_i$, and vice versa. It is allowed that a pair of cities is connected by more than one road.

The last line contains one integer $S$ ($0 \le S \le 10^9$) — the number of lamziks which we can spend for reforms.

## Output

In the first line print $K$ — the minimum possible total dissatisfaction with *main roads*.

In each of the next $n$ - $1$ lines print two integers $x$, $v_x$, which mean that the road $x$ is among main roads and the road $x$, after the reform, has the level of dissatisfaction $v_x$.

Consider that roads are numbered from $1$ to $m$ in the order as they are given in the input data. The edges can be printed in arbitrary order. If there are several answers, print any of them.

**Examples**

| input |
| --- |
| 6 9<br>1 3 1 1 3 1 2 2 2<br>4 1 4 2 2 5 3 1 6<br>1 2<br>1 3<br>2 3<br>2 4<br>2 5<br>3 5<br>3 6<br>4 5<br>5 6<br>7 |

| output |
| --- |
| 0<br>1 1<br>3 1<br>6 1<br>7 2<br>8 -5 |

| input |
| --- |
| 3 3<br>9 5 1<br>7 7 2<br>2 1 |

```
3 1
3 2
2
```

**output**

```
5
3 0
2 5
```