
tasks

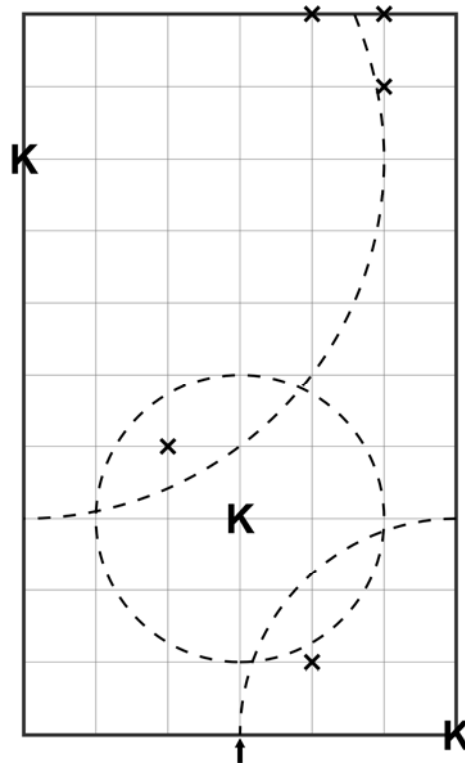
task	grad	kutije	kaos
source file	grad.pas grad.c grad.cpp	kutije.pas kutije.c kutije.cpp	kaos.pas kaos.c kaos.cpp
input data	stdin		
output data	stdout		
time limit (Intel Celeron 2.66Ghz)	1 sec		2 sec
memory limit (heap)	32 MB		
memory limit (stack)	8 MB		
points	50	70	80
	200		

grad

A construction site is defined as a rectangle in the coordinate system with sides parallel to the coordinate axes, with one corner in $(0,0)$ and the opposite corner in (X,Y) .

The entrance to the site is **in the middle of the bottom side**. There are N cranes on the construction site. Each crane is situated in a point on the site, it can rotate 360 degrees and the maximum reach is known for each crane.

The truck unloads heavy equipment on the entrance of the site and after that the equipment is carried across the site by a sequence of crane movements. In each step, one crane **picks up the equipment** and **leaves it at any place within the maximum reach of that crane**.



Write a program that, given a list of K destinations within the site, determines for each destination if it is possible to carry the equipment to this destination.

input data

The first line of input contains two integers X and Y , $2 \leq X, Y \leq 200$, X is even.

Next line contains an integer N , $1 \leq N \leq 50$, the number of cranes.

Each of the following N lines contains three integers A , B and C – (A,B) is the position of the crane, and C is its maximum reach, $0 \leq A \leq X$, $0 \leq B \leq Y$, $0 \leq C \leq 200$.

The next line contains an integer K , $3 \leq K \leq 30$, the number of destinations.

Each of the following N lines contains two integers D and E – (D,E) is the position of one destination, $0 \leq D \leq X$, $0 \leq E \leq Y$.

output data

Each of the K lines should contain the word '**DA**' or '**NE**' – '**DA**' means that it is possible to deliver the equipment to that destination, and '**NE**' means that it is not possible.

examples**input**

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

output

```
DA
NE
NE
DA
```

input

```
6 10
3
3 3 2
6 0 3
0 8 5
5
4 1
2 4
4 10
5 10
5 9
```

output

```
DA
DA
DA
NE
NE
```

input

```
8 5
4
2 1 3
4 5 1
6 4 1
5 2 2
5
0 2
0 3
4 4
7 4
7 5
```

output

```
DA
DA
NE
DA
NE
```

kutije

There are N open boxes, arranged from left to right. There are **exactly two balls** in each box, and each ball is either **white** or **black**.

There is a crane above the first (leftmost) box and we can operate the crane with the following commands:

- **LIJEVO** move the crane one box to the left
- **DESNO** move the crane one box to the right
- **UZMI BIJELU** take the white ball from the box below
- **UZMI CRNU** take the black ball from the box below
- **SPUSTI BIJELU** drop the white ball to the box below
- **SPUSTI CRNU** drop the black ball to the box below

The crane can carry **only two balls** at the same time, but in each box we can put an **unlimited number** of balls.

We say that the balls are "**arranged**" if all of the following statements are satisfied:

- there are exactly two balls in each box and they have the same color
- there is no box with black balls between any two boxes with white balls
- there is no box with white balls between any two boxes with black balls

Write a program that will find some **smallest** sequence of commands that will arrange the balls so that the above rules are satisfied.

Each command in your solution must be **valid** (the crane must not go left from the leftmost box, it must not carry more than two balls, etc...).

Note: if your solution is correct for some test data, but the number of the commands is not minimal, you will score 3 of 7 points.

input data

The first line of input contains an integer N , $2 \leq N \leq 500$.

The second line contains a sequence of N letters – the initial contents of the boxes.

For each box (from left to the right) there are two letters 'B' or 'C'. The letter 'B' represents a white ball and the letter 'C' represents a black ball.

output data

Output all of the commands, each on its own line.

Note: the test data will be such that a solution, although not necessarily unique, will always exist.

examples**input**

3
CC BC BC

output

DESN
UZMI BIJELU
DESN
SPUSTI BIJELU
UZMI CRNU
LIJEVO
SPUSTI CRNU

input

5
BC BB CB BB CC

output

UZMI CRNU
DESN
DESN
UZMI CRNU
DESN
SPUSTI CRNU
UZMI BIJELU
SPUSTI CRNU
UZMI BIJELU
LIJEVO
SPUSTI BIJELU
LIJEVO
LIJEVO
SPUSTI BIJELU

input

7
BB BC CC CC CC BC BB

output

UZMI BIJELU
UZMI BIJELU
DESN
DESN
DESN
DESN
SPUSTI BIJELU
SPUSTI BIJELU
UZMI CRNU
UZMI CRNU
LIJEVO
LIJEVO
LIJEVO
LIJEVO
SPUSTI CRNU
SPUSTI CRNU
DESN
UZMI BIJELU
DESN
DESN
DESN
DESN
SPUSTI BIJELU
UZMI CRNU
LIJEVO
LIJEVO
LIJEVO
LIJEVO
SPUSTI CRNU

Little Lovro likes to play games with words. During the last few weeks he realized that some words don't like each other.

The words A and B **don't like** each other if **the word A is lexicographically before the word B**, but **the word B' is lexicographically before the word A'**, where X' stands for the word X reversed (if X="kamen" then X'="nemak"). For example, the words "lova" and "novac" like each other, but the words "aron" and "sunce" don't like each other.

Given some set of the words, we define **the degree of chaos** of the set as **the number of pairs** of different words that **don't like** each other.

Write a program that, given a set of words, finds the chaos degree for the set.

input data

The first line of input contains an integer N, $2 \leq N \leq 100\,000$.

Each of the following N lines contains one word – a sequence of at most 10 lowercase letters of the English alphabet ('a'-'z'). There will be no two identical words in the set.

output data

The first and only line of output should contain a single integer – the chaos degree of the given set of words.

Note: use 64-bit signed integer type (int64 in Pascal, long long in C/C++).

examples

input

2
lopta
kugla

output

0

input

4
lova
novac
aron
sunce

output

3

input

14
branimir
vladimir
tom
kruz
bred
pit
zemlja
nije
ravna
ploca
ko
je
zapalio
zito

output

48