GLOBUSSOFT
TECHNOLOGY AHEAD OF TIME

Greetings From Globussoft

- Given below are 5 Programming questions, you have to solve any 3 out of 5 questions.

- These 5 questions you can attempt in any technology like C/C++, java, .Net, PHP

- To solve these 3 questions you've max. 3 hours.

- While Solving these questions you are not allowed to use any **Search Engine** like Google, Yahoo, Bing …

All the best for your test

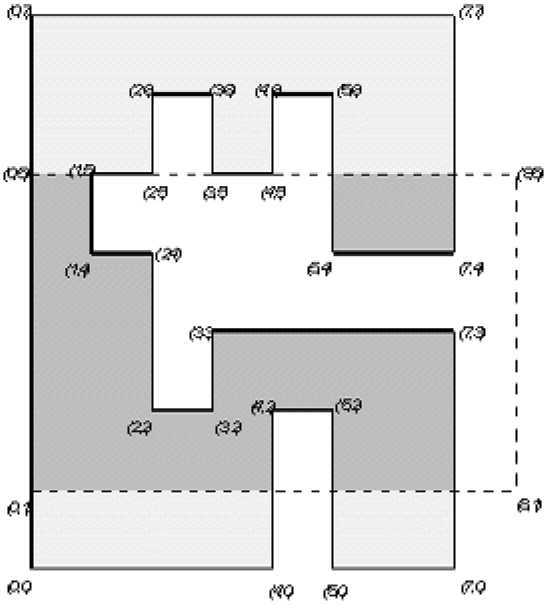
Globussoft

QUESTION - 1

We have a polygon chosen in the cartesian coordinate system. Sides of the polygon are parallel to the axes of coordinates. Every two consecutive sides are perpendicular and coordinates of every vertex are integers. We have also given a window that is a rectangle whose sides are parallel to the axes of coordinates. The interior of the polygon (but not its periphery) is coloured red. What is the number of separate red fragments of the polygon that can be seen through the window?

Example

Look at the figure below:



There are two separate fragments of the polygon that can be seen through the window.

Task

Write a program that for each test case:

- reads descriptions of a window and a polygon;
- computes the number of separate red fragments of the polygon that can be seen through the window;
- outputs the result.

Input

The number of test cases *t* is in the first line of input, then *t* test cases follow separated by an empty line.

In the first line of a test case there are four integers x_1, y_1, x_2, y_2 from the range [0..10000], separated by single spaces. The numbers x_1, y_1 are the coordinates of the top-left corner of the window. The numbers x_2, y_2 are the coordinates of the bottom-right corner of the window. The next line of the input file contains one integer *n*, 4 <= *n* <= 5000, which equals the number of vertices of the polygon. In the following *n* lines there are coordinates of polygon's vertices given in anticlockwise direction, i.e. the interior of the polygon is on the left side of its periphery when we move along the sides of the polygon according to the given order. Each line contains two integers *x*, *y* separated by a single space, 0 <= *x* <= 10000, 0 <= *y* <= 10000. The numbers in the *i*-th line, are coordinates of the *i*-th vertex of the polygon.

Output

For each test case you should output one line with the number of separate red fragments of the polygon that can be seen through the window.

Example

Sample input:

```
1
0 5 8 1
24
0 0
4 0
4 2
5 2
5 0
7 0
7 3
3 3
3 2
2 2
2 4
1 4
1 5
2 5
2 6
3 6
3 5
4 5
4 6
5 6
5 4
7 4
7 7
0 7
```

Sample output:

```
2
```

QUESTION – 2

There is a building with flat square roof of size 3^k*3^k and sides parallel to north-south and east-west directions. The roof is covered with square tiles of size *1* (with a side of length 1), but one of the tiles has been removed and there is a hole in the roof (big enough to fall in). The tiles form a rectangular mesh on the roof, so their positions may be specified with coordinates. The tile at the southwestern corner has coordinates (*1,1*). The first coordinate increases while going eastwards, and the second while going northwards.

Sleepwalker wanders across the roof, in each step moving from the tile he is standing on to the adjacent one on the east(E), west(W), south(S), or north(N). The sleepwalker roof ramble starts from the southwestern corner tile. The description of the path is a word d_k built of the letters `N, S, E, W` denoting respectively a step to the north, south, east and west. For $k = 1$ the word describing the path of sleepwalker is

d_1 = `EENNWSWN`

For $k = 2$ the word describing the path of sleepwalker is

```
d2 = NNEESWSEENNEESWSEEEENNWSWNNEENNWSW -
     NNEENNWSWNWWWSSENESSSSWWNENWWSSW -
     WNENWNEENNWSWN.
```

(See the picture that shows how the sleepwalker would go across a roof of dimension *3*3* or *9*9*.) Generally, if $k>=1$, the description of a sleepwalker's path on the roof of dimension $3^{k+1}*3^{k+1}$ is a word:

d_{k+1} = a(d_k) E a(d_k) E d_k N d_k N d_k W c(d_k) S b(d_k) W b(d_k) N d_k

where functions **a**, **b** and **c** denote the following permutations of letters specifying directions:

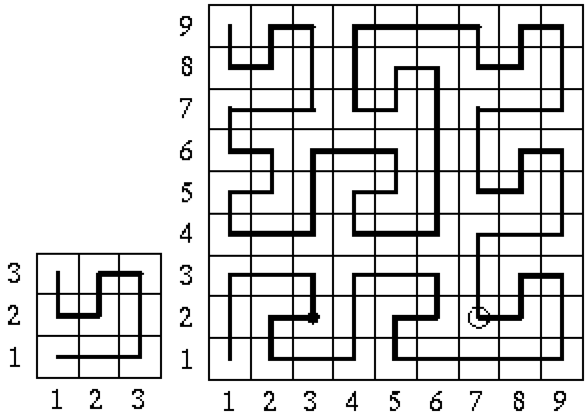
```
a: E->N W->S N->E S->W
b: E->S W->N N->W S->E
c: E->W W->E N->S S->N
```

E.g. a(SEN)=WNE, b(SEN)=ESW, c(SEN)=NWS.

We start observing sleepwalker at the time he stands on the tile of coordinates (u_1, u_2). After how many steps will sleepwalker fall into the hole made after removing the tile of coordinates (v_1, v_2)?

Example

There are sleepwalker's paths on roofs of dimension *3*3* and *9*9* on the picture below. In the second case, the point at which the observation starts and the hole have been marked. The sleepwalker has exactly *20* steps to the hole (from the moment the observation starts).



Task

Write a program which:

- reads from the standard input integer *k* denoting the size of the roof (3^k*3^k), the position of the sleepwalker at the moment the observation starts and the position of the hole,
- computes the number of steps that the sleepwalker will make before he falls into the hole,
- writes the result to the standard output.

Input

The number of test cases t is in the first line of input, then t test cases follow separated by an empty line. In the first line of each test case one integer *k*, *1<=k<=60*, denoting the size of the roof (3^k*3^k) is written. In each of the following two lines of the test case two natural numbers *x, y* separated with a space are written, *1<=x<=3^k, 1<=y<=3^k*. The numbers in the second line are the coordinates of the tile the sleepwalker is standing on. The numbers in the third line are the coordinates of the hole. You may assume, that with these data the sleepwalker will eventually fall into the hole after some number of steps.

Output

The only line of output for each test case should contain the number of steps on the sleepwalker's path to the hole.

Example

Sample input:

```
1
2
3 2
7 2
```

Sample output

```
20
```

QUESTION – 3

Protocol is really weird in Byteland. For instance, it is required that, when presenting arms before an officer, soldiers should stand in a single row (at positions numbered from 1 to *n*). Soldiers may have one of 4 possible ranks, distinguished by the number of squiggles on the epaulets (between 1 and 4). Soldiers standing beside each other must have a difference in rank of at least two squiggles. Moreover, there are additional sets of rules (different for every province). Each rule states that soldiers standing at some given positions of the row must differ in rank by at least a squiggle.

Starting from the new year onwards, some provinces are changing their set of protocol rules. As the Senior Military Secretary of Protocol, it is your task to approve the new rules. To your surprise, some of the provinces have put forward protocol rules which are quite impossible to fulfill, even if the soldiers were to be specially selected for the purpose of presenting arms. Detect all such offending provinces and on no account approve their laws.

Input

The first line of input contains a single positive integer t<=10 - the number of provinces which are proposing new laws. t sets of rules follow, separated by empty lines.

Each set of rule begins with a line containing two non-negative integers n p (n<=100000, p<=100000) - the number of soldiers arranged and the number of rules proposed in the province, respectively. Each of the next p lines contains a single rule: an integer b_i (2<=b_i<=n), followed by b_i integers $a_1,a_2,...,a_{b_i}$ (1<=a_k<=n). Such a rule means that soldiers standing at positions $a_1,a_2,...,a_{b_i}$ must all be of different rank.

Output

For every set of rules presented at input, output a single line containing the word `rejected` if no unit of soldiers can be arranged in accordance with protocol, or the word `approved` in the opposite case.

Example

```
Input:
2

2 1
2 1 2

5 2
3 1 3 2
4 2 3 4 5

Output:
approved
rejected
```

QUESTION – 4

Let us consider a triangle of numbers in which a number appears in the first line, two numbers appear in the second line etc. Develop a program which will compute the largest of the sums of numbers that appear on the paths starting from the top towards the base, so that:

- on each path the next number is located on the row below, more precisely either directly below or below and one place to the right;
- the number of rows is strictly positive, but less than 100;
- all numbers are positive integers between O and 99.

Take care about your fingers, do not use more than **256** bytes of code.

Input

In the first line integer n - the number of test cases (equal to about 1000). Then n test cases follow. Each test case starts with the number of lines which is followed by their content.

Output

For each test case write the determined value in a separate line.

Example

Input:

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

Output:

```
5
9
```

QUESTION – 5

A very big corporation is developing its corporate network. At the beginning, each of the **N** enterprises of the corporation, numbered from 1 to **N**, organized its own computing and telecommunication center. Soon, for amelioration of the services, the corporation started to collect some enterprises in clusters, each of them served by a single computing and telecommunication center as follows. The corporation chose one of the existing centers **I** (serving the cluster **A**) and one of the enterprises **J** in some other cluster **B** (not necessarily the center) and linked them with a telecommunication line. The length of the line between the enterprises **I** and **J** is |**I** – **J**|(mod 1000). In such a way two old clusters are joined to form a new cluster, served by the center of the old cluster **B**. Unfortunately after each join the sum of the lengths of the lines linking an enterprise to its serving center could be changed and the end users would like to know what is the new length.

Write a program to keep trace of the changes in the organization of the network that is able at each moment to answer the questions of the users.

Input

The first line of the input file will contains only the number **T** of the test cases (1 <= **T** <= 5). Each test will start with the number **N** of enterprises (5<=**N**<=20000). Then some number of lines (no more than 200000) will follow with one of the commands:

E I– asking the length of the path from the enterprise **I** to its serving center at the moment; **I I J** – informing that the serving center **I** is linked to the enterprise **J**. The test case finishes with a line containing the word **O**. There are fewer **I** commands than **N** commands.

Output

The output should contain as many lines as the number of **E** commands in all test cases. Each line must contain a single number – the requested sum of lengths of lines connecting the corresponding enterprise with its serving center.

Example

Input:

```
1
4
E 3
I 3 1
E 3
I 1 2
E 3
I 2 4
E 3
O
```

Output:

```
0
2
3
5
```