Problem A

Crashing Robots

In a modernized warehouse, robots are used to fetch the goods. Careful planning is needed to ensure that the robots reach their destinations without crashing into each other. Of course, all warehouses are rectangular, and all robots occupy a circular floor space with a diameter of 1 meter. Assume there are N robots, numbered from 1 through N. You will get to know the position and orientation of each robot, and all the instructions, which are carefully (and mindlessly) followed by the robots. Instructions are processed in the order they come. No two robots move simultaneously; a robot always completes its move before the next one starts moving.

A robot crashes with a wall if it attempts to move outside the area of the warehouse, and two robots crash with each other if they ever try to occupy the same spot.

Input specifications

The first line of input is K, the number of test cases. Each test case starts with one line consisting of two integers, $1 \leq A, B \leq 100$, giving the size of the warehouse in meters. A is the length in the EW-direction, and B in the NS-direction.

The second line contains two integers, $1 \leq N, M \leq 100$, denoting the numbers of robots and instructions respectively.

Then follow N lines with two integers, $1 \leq X_i \leq A, 1 \leq Y_i \leq B$ and one letter (`N`, `S`, `E` or `W`), giving the starting position and direction of each robot, in order from 1 through N. No two robots start at the same position.

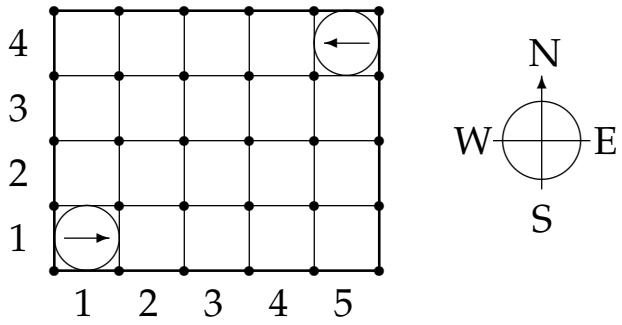


Figure 1: The starting positions of the robots in the sample warehouse

Finally there are M lines, giving the instructions in sequential order. An instruction has the following format:

`<robot #> <action> <repeat>`

Where `<action>` is one of

- `L`: turn left 90 degrees,
- `R`: turn right 90 degrees, or
- `F`: move forward one meter,

and $1 \leq$ `<repeat>` ≤ 100 is the number of times the robot should perform this single move.

Output specifications

Output one line for each test case:

- `Robot i crashes into the wall`, if robot i crashes into a wall. (A robot crashes into a wall if $X_i = 0$, $X_i = A + 1$, $Y_i = 0$ or $Y_i = B + 1$.)
- `Robot i crashes into robot j`, if robots i and j crash, and i is the moving robot.
- `OK`, if no crashing occurs.

Only the first crash is to be reported.

Sample input

```
4
5 4
2 2
1 1 E
5 4 W
1 F 7
2 F 7
5 4
2 4
1 1 E
5 4 W
1 F 3
2 F 1
1 L 1
1 F 3
5 4
```

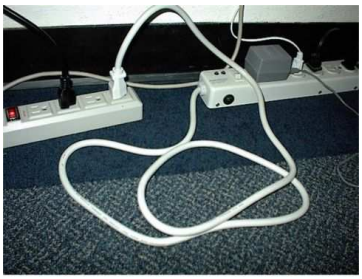
```
2 2
1 1 E
5 4 W
1 L 96
1 F 2
5 4
2 3
1 1 E
5 4 W
1 F 4
1 L 1
1 F 20
```

Output for sample input

```
Robot 1 crashes into the wall
Robot 1 crashes into robot 2
OK
Robot 1 crashes into robot 2
```

Problem E

Electrical Outlets



Roy has just moved into a new apartment. Well, actually the apartment itself is not very new, even dating back to the days before people had electricity in their houses. Because of this, Roy's apartment has only one single wall outlet, so Roy can only power one of his electrical appliances at a time.

Roy likes to watch TV as he works on his computer, and to listen to his HiFi system (on high volume) while he vacuums, so using just the single outlet is not an option. Actually, he wants to have all his appliances connected to a powered outlet, all the time. The answer, of course, is power strips, and Roy has some old ones that he used in his old apartment. However, that apartment had many more wall outlets, so he is not sure whether his power strips will provide him with enough outlets now.

Your task is to help Roy compute how many appliances he can provide with electricity, given a set of power strips. Note that without any power strips, Roy can power one single appliance through the wall outlet. Also, remember that a power strip has to be powered itself to be of any use.

Input specifications

Input vill start with a single integer $1 \leq N \leq 20$, indicating the number of test cases to follow. Then follow N lines, each describing a test case. Each test case starts with an integer $1 \leq K \leq 10$, indicating the number of power strips in the test case. Then follow, on the same line, K integers separated by single spaces, $O_1\ O_2 \ldots O_K$, where $2 \leq O_i \leq 10$, indicating the number of outlets in each power strip.

Output specifications

Output one line per test case, with the maximum number of appliances that can be powered.

Sample input

```
3
3 2 3 4
```

```
10 4 4 4 4 4 4 4 4 4 4
4 10 10 10 10
```

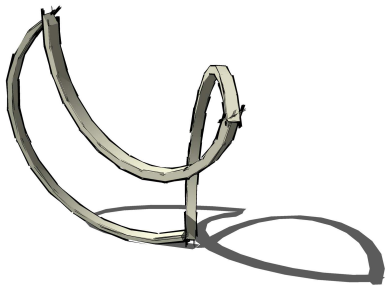
Output for sample input

```
7
31
37
```

Problem I

Playground

George has $K \leq 20$ steel wires shaped in the form of half-circles, with radii $a_1, a_2, \ldots, a_K$. They can be soldered (connected) at the ends, in any angle. Is it possible for George to make a closed shape out of these wires? He does not have to use all the wires.



The wires can be combined at any angle, but may not intersect. Beware of floating point errors.

Input specifications

Each data set consists of a number $0 < K \leq 20$ on a line by itself, followed by a line of K space-separated numbers a_i. Each number is in the range $0 < a_i < 10^7$, and has at most 3 digits after the decimal point.

The input will be terminated by a zero on a line by itself.

Output specifications

For each test case, there should be one word on a line by itself; "YES" if it is possible to make a simple connected figure out of the given arcs, and "NO" if it isn't.

Sample input

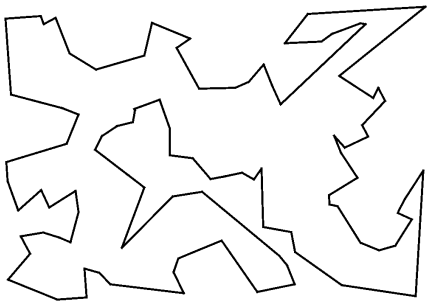
```
1
4.000
2
1.000 1.000
3
1.455 2.958 4.424
7
1.230 2.577 3.411 2.968 5.301 4.398 6.777
0
```

Output for sample input

```
NO
YES
NO
YES
```

Problem F

Traveling Salesman



Long before the days of international trade treaties, a salesman would need to pay taxes at every border crossed. So your task is to find the minimum number of borders that need to be crossed when traveling between two countries. We model the surface of Earth as a set of polygons in three dimensions forming a closed convex 3D shape, where each polygon corresponds to one country. You are not allowed to cross at points where more than two countries meet.

Input specifications

Each test case consists of a line containing c, the number of countries ($4 \leq c \leq 6000$), followed by c lines containing the integers $n\ x_1\ y_1\ z_1\ \ldots\ x_n\ y_n\ z_n$, describing (in order) the n corners of a closed polygon ($3 \leq n \leq 20$). Then follows a line with one integer m ($0 < m \leq 50$), and then m lines with queries $c_a\ c_b$, where c_a and c_b are country numbers (starting with 1). No point will be on the line between two connected points, and $-10^6 \leq x, y, z \leq 10^6$ for all points. No two non-adjacent edges of a country share a common point. The input is terminated by a case where $c = 0$, which should not be processed.

Output specifications

For each query, output the number of borders you must cross to go from c_a to c_b.

Sample input

```
6
4 0 0 0 0 0 1 0 1 1 0 1 0
4 1 0 0 1 0 1 1 1 1 1 1 0
4 0 0 0 1 0 0 1 0 1 0 0 1
4 0 1 0 1 1 0 1 1 1 0 1 1
4 0 0 0 0 1 0 1 1 0 1 0 0
4 0 0 1 0 1 1 1 1 1 1 0 1
2
1 2
1 3
0
```

Output for sample input

```
2
1
```

Problem I

Honeycomb Walk

A bee larva living in a hexagonal cell of a large honeycomb decides to creep for a walk. In each "step" the larva may move into any of the six adjacent cells and after n steps, it is to end up in its original cell.

Your program has to compute, for a given n, the number of different such larva walks.

Input specifications

The first line contains an integer giving the number of test cases to follow. Each case consists of one line containing an integer n, where $1 \leq n \leq 14$.

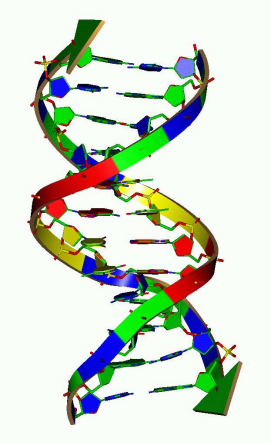
Output specifications

For each test case, output one line containing the number of walks. Under the assumption $1 \leq n \leq 14$, the answer will be less than 2^{31}.

Sample input	Output for sample input
2	
2	6
4	90

Problem D

Copying DNA



Evolution is a seemingly random process which works in a way which resembles certain approaches we use to get approximate solutions to hard combinatorial problems. You are now to do something completely different.

Given a DNA string S from the alphabet {`A,C,G,T`}, find the minimal number of copy operations needed to create another string T. You may reverse the strings you copy, and copy both from S and the pieces of your partial T. You may put these pieces together at any time. You may only copy contiguous parts of your partial T, and all copied strings must be used in your final T. Example: From S = "`ACTG`" create T = "`GTACTATTATA`"

1. Get `GT.........` by copying and reversing "`TG`" from S.
2. Get `GTAC.......` by copying "`AC`" from S.
3. Get `GTAC...TA..` by copying "`TA`" from the partial T.
4. Get `GTAC...TAAT` by copying and reversing "`TA`" from the partial T.
5. Get `GTACAATTAAT` by copying "`AAT`" from the partial T.

Input specifications

The first line of input gives a single integer, $1 \leq t \leq 100$, the number of test cases. Then follow, for each test case, a line with the string S of length $1 \leq m \leq 18$, and a line with the string T of length $1 \leq n \leq 18$.

Output specifications

Output for each test case the number of copy operations needed to create T from S, or "`impossible`" if it cannot be done.

Sample input

```
5
ACGT
GTAC
A
C
ACGT
TGCA
ACGT
TCGATCGA
A
AAAAAAAAAAAAAAAAAA
```

Output for sample input

```
2
impossible
1
4
6
```

Problem H

Exploding CPU

source: cpu.*

The well known hardware manufacturing company *Processors for Professors* is about to release a highly specialized CPU with exceptional functionality in, amongst other areas, number theory. It has, for example, an instruction PFACT that takes one parameter and returns all prime factors of that parameter, with an outstanding execution speed. It has, however, one considerable problem. The scientists at the testing lab has just found out that the PFACT instruction for some special input values freaks out and makes the entire processor explode. Even though this could be an amusing effect, it is not the way it was intended to work. The skilled mathematicians have, by trial and error, found that the explosive numbers all share the same interesting number theoretic properties, which might be of help when troubleshooting.

An explosive number is a number $x = p_0 p_1 p_2 \dots p_n$ where all p_is are distinct prime numbers such that $p_i = A p_{i-1} + B$ for $i = 1, 2, \dots, n$. $n \geq 3$, $p_0 \equiv 1$. A and B are always integers, and might be different for different explosive numbers.

For example, the processor will explode when factorizing the number 4505, because $4505 = 1 \cdot 5 \cdot 17 \cdot 53$ and $5 = 3 \cdot 1 + 2$, $17 = 3 \cdot 5 + 2$ and $53 = 3 \cdot 17 + 2$ and the numbers 5, 17 and 53 are all prime numbers. In this case $A = 3$ and $B = 2$.

You are kindly asked to write a computer program that will aid this company in estimating the impact of the errors, by calculating the amount of explosive numbers that exists within a given range of integers.

Input specifications

The input starts with a row containing the number $0 \leq N \leq 100$ of test cases that will follow. For each test case, there will be one row containing two integers, x_L and x_H separated by a single space. These numbers are such that $0 \leq x_L \leq x_H \leq 2,000,000,000$.

Output specifications

For each test case, output the number of explosive numbers that exist in the range $x_L \leq x \leq x_H$.

Sample input

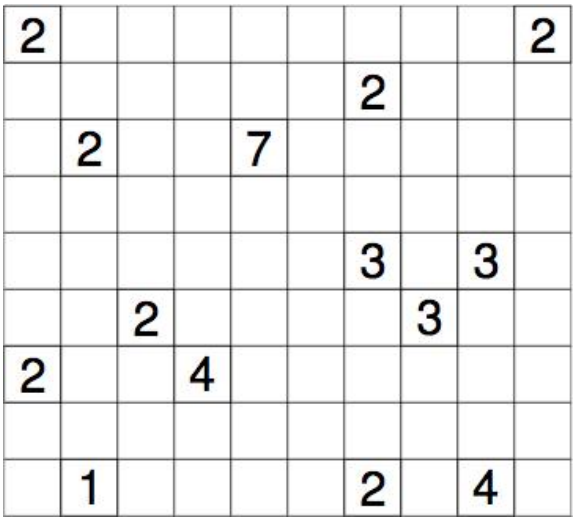
```
2
4505 4505
0 5000
```

Output for sample input

```
1
5
```

Problem J

Solving Nurikabe



Nurikabe is a binary determination puzzle originating from Japan. Given a grid where some cells contain numbers, your objective is to mark each cell as either island (white) or water (black), while obeying the following constraints:

- Each island has exactly one numbered tile n. An island consists of n connected cells. Two cells are connected if they share a side. Two cells belong to the same island if there exists a path going through connected island cells.
- All water cells (black) are connected. Water cells are connected in the same manner as island cells.
- Within a 2×2 block there must be at least one cell belonging to an island.

In this problem, you are asked to solve Nurikabe puzzles.

Input specifications

The first line of input contains a single number T, the number of test cases that follow. The first line of each test case contains integers n and m, the size of a puzzle in rows and columns. The next n lines contains the rows of the puzzle. Each line contains characters from the set `123456789.` where `.` represent a cell to be filled in. Each puzzle is guaranteed to have at exactly one solution.

Output specifications

For each test case, output the puzzle in its solved state. Use the `#` character to represent water, and `.` to represent island. The numbers in the puzzle should be included.

Notes and Constraints

- $0 < T \leq 100$
- $1 \leq n, m \leq 10$

Sample input

```
1
9 10
2........2
......2...
.2..7.....
..........
......3.3.
..2....3..
2..4......
..........
.1....2.4.
```

Output for sample input

```
2.#...##.2
###..#2###
#2#.7#.#.#
#.######.#
##.#..3#3#
.#2####3##
2##4.#..#.
##..#####.
#1###.2#4.
```