2015 University of Florida ACM
High School Programming Contest

Problem A
Simple Polygons

A polygon is a two dimension figure defined by a chain of line segments that form a closed loop (a polygon must have a non-zero area); it is conventional to write a polygon as a sequence of its vertices. For example, we can define a right-triangle with the point sequence $\{(0,0),(0,1),(1,0)\}$. A *simple* polygon is one which does not cross over itself.

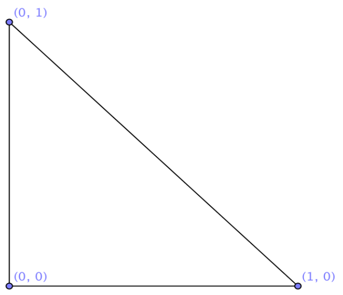


Figure 1: Simple polygon

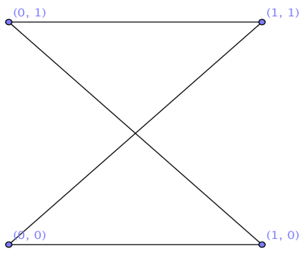


Figure 2: Non-simple polygon

Given a set of points, can you count the number of simple polygons that can be formed using only points from this set?

Input

The input will begin with a line containing a single positive integer t representing the number of test cases you must process. The first line of each test case is N, the number of points $(1 \leq N \leq 9)$. Following will be N lines each specifying a point by its x and y co-ordinates which are guaranteed to be integers. You are guaranteed that no three points will be collinear (no line can be drawn through three points).

Output

For each test case print the number of simple polygons that can be formed on its own line.

Sample Input	**Sample Output**

1
4
0 0
0 1
1 0
1 1
	5

Problem B
Research Problems

Academic research is not all it's cracked up to be. There are scandals everywhere, ranging from plagiarism to fudging numerical data. Some academic papers get more scrutiny than others, such as when some unfortunate soul claims he/she has proven that P=NP; in these cases the authors and their 'results' are quickly put to shame.

In the interest of maintaining the integrity of academia, researchers everywhere would really benefit from having a computer program that could determine if a certain research paper is accurate or not. Unfortunately it's not possible to verify a paper accurately. Instead, it is much easier (but still helpful) to identify circular reasoning. Circular reasoning is characterized by the ability to continually follow the references of research papers and end up back at the paper you started with. Given a set of papers, can you determine if there are any cases of circular reasoning?

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with a single integer N ($1 \leq N \leq 100$) which is the number of papers in this test case. The i-th paper starts with "i: " and is followed by text. The text for a paper may span multiple lines, there will be no ":" character used other than at the beginning of a paper, and a "*" character will signify the end of a test case. Inside the text will be a set of references; you may assume that any number in a paper's contents is a reference to a valid paper in the current set.

Output

For each test case output "No problems here, sir" if there is no circular reasoning; otherwise output "We have got some problems". The output for each test case should be on its own line.

Sample Input

```
2
4
1: Abstract. In this paper we describe a universal algorithm
that proves P is a proper subset of NP. We utilize results from
[2], [3], and [4].
2: Hello this is a totally truthful result.
3: Another totally truthful result using [4].
4: Wow this can’t be entirely truthful. [2]
*
2
1: Shocking result where Euler’s theorem turned out to
not actually be true! [2]
2: Pigs can fly. [1]
*
```

Sample Output

```
No problems here, sir
We have got some problems
```

Problem C
Breaking RSA

Your friends Alice and Bob are very secretive people. Whenever they send a message to each other they encrypt it using the *RSA* algorithm. For the algorithm to work, Alice and Bob must each pick two prime numbers p and q and from these two numbers they can follow the *RSA* procedure to generate their own public and private key.

To send an encrypted message to Alice, Bob would have to take her public-key and encrypt his message with it; the only way to decrypt this message is with the private-key that Alice has. A part of the public-key that Alice releases is the cryptographic modulus: $n = pq$. Since Alice's super secretive private-key is determined solely from the values of p and q, if we were to recover these values we could break her encryption!

Your job is simple: given Alice's modulus help us break her encryption.

Input

The input will begin with a line containing a single positive integer t representing the number of modulus values you must process. Following will be t lines each containing a value x ($x \leq 1,000,000$) which is guaranteed to have only two prime factors.

Output

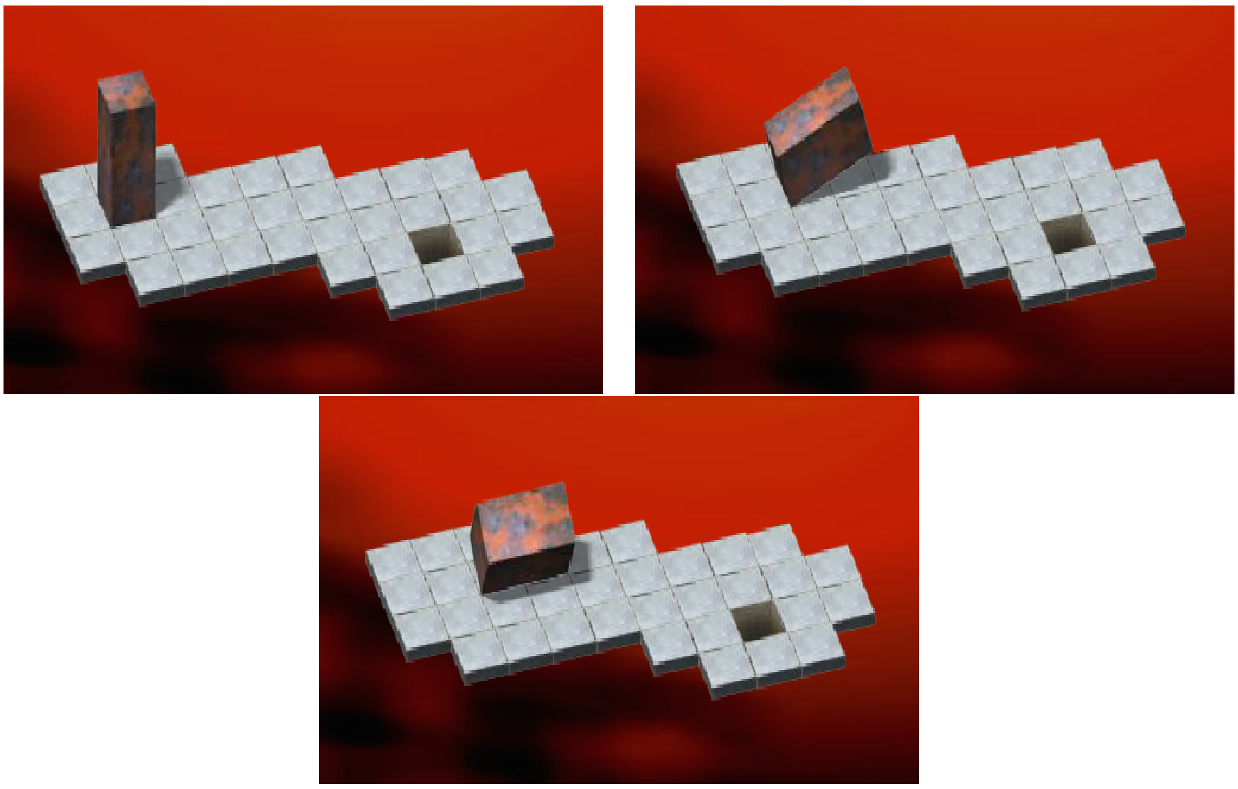
For each modulus print "p q" (where $p \leq q$) on its own line.

Sample Input	Sample Output

4
10
14
77
187
	2 5
	2 7
	7 11
	11 17

Problem D
Bloxorz

Bloxorz, a flash game popularize by the Cool Math website, has a special place in the hearts of all middle schoolers. The game environment consists of a tiled grid on the plane. You as the player are a rectangular prism that is 1 tile in length and width but 2 tiles high. The goal of this game is to make a sequence of moves that will move you onto the end tile in the upright position. There are two 'states' your game character can be in: upright or laying down. Whenever you move in the upright state, your piece lays down in that direction. While you're laying down, if you move to a spot parallel to your piece you simply roll over and stay in the laying down state; otherwise, you move up just like you would expect.



In this problem you will be given a grid corresponding to an instance of the Bloxorz game. The grid has R rows and C columns and consists of characters. 'S' denotes the position you start on (in the upright position, of course) and 'G' is your goal. If any piece of your block is off the grid or touches a '#' tile it will fall off and the game must be restarted. '*' tiles can be moved on.

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with two integers R $(1 \leq R \leq 500)$ and C $(1 \leq C \leq 500)$. Following that will be R lines, each line containing C characters which correspond to a row in the game grid.

Output

For each test case print “YES” if the game can be won and print “NO” otherwise. The output for each test case should be on its own line.

Sample Input	Sample Output

```
2
3 8
S******##
******G*
########
2 4
S*G*
####
```
	```
	YES
	NO
	```

Problem E
Project Management

You have just been put in charge of managing a big project for the Association of Counterintelligence Measures (ACM). Your boss is interested in knowing how hard your team is really working so that she can adjust your team budget accordingly. She believes a good indicator of this is the *greatest* amount of people working on the project at the same time.

Whenever one of your team members works on the project they log their start and end times. Given logs for your whole team, can you determine the greatest number of people working on the project at the same time?

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with an integer N $(1 \leq N \leq 1,000,000)$, which is the number of logs in this test case. Following that will be N lines each specifying the start and end time, i and j $(1 \leq i \leq j \leq 1,000,000,000)$. For the sake of simplicity you may assume that work was being done at the end time of a log.

Output

For each test case print the greatest number of people working on the project at the same time. The output for each test case should be on its own line.

Sample Input

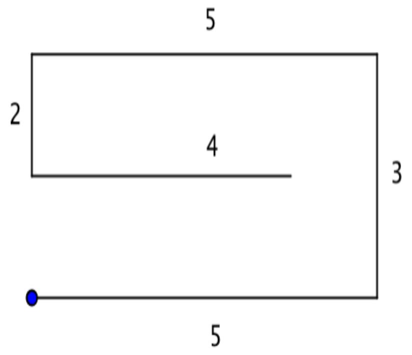
```
2
3
500 10000
10000 1500000
1 499
2
1000 10000
1 999
```

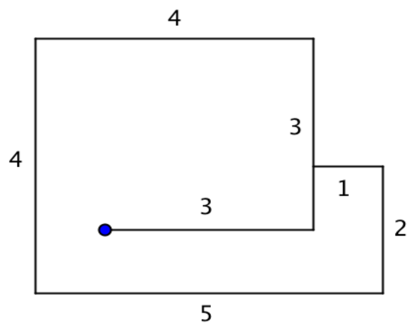
Sample Output

```
2
1
```

Problem F
Rectangular Spirals

Vatsal (a member of the UF Programming Team) is notorious for wandering around campus without any sense of direction. After every walk Vatsal begins to wonder if he ever crossed over his path. Can you help him out?





Vatsal begins each of his walks at the center of campus facing east. During the ith stage in his walk he moves a distance of d_i and then turns 90 degrees counterclockwise. If at any stage Vatsal ever walks over his path it is said to be *self intersecting*.

Input

The input will begin with a line containing a single positive integer t representing the number of paths you must process. Following will be t lines each containing a sequence of positive integers that are space delimited which represent a single instance of wandering Vatsal has done. No sequence will have more than 1,000 stages and no d_i will be greater than 100,000.

Output

For each walk, print "Yes" if the walk is self intersecting and "No" otherwise. The output for each walk should be on its own line.

Sample Input	**Sample Output**

`2`
`5 3 5 2 4`
`3 3 4 4 5 2 1`
	`No`
	`Yes`

Problem G
Combination Lock

After hanging out in the fish bowl (also called the ACM room if you're a scrub) for what seemed like an eternity Joon wanted to go grab an ice cold beverage from the refrigerator. Recently, however, a two-digit combination lock was added to the refrigerator door to prevent unauthorized access (Joon); fortunately Joon already knows the two-digit combination that will unlock the refrigerator door!

Joon is good with numbers but is incredibly lazy, therefore he wants to unlock the door with as little effort as possible. To unlock the door he will make a sequence of moves where each move can alter one of the digits on the lock by either incrementing it or decrementing it (increase/decrease by 1). If Joon increments a digit set to 9, then the digit wheel will wrap around to 0. Similarly, if Joon decrements a digit set to 0 it will wrap around to 9.

Given the initial configuration of the combination lock and the unlocked configuration, can you determine the minimum number of moves Joon must make to open the refrigerator door?

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case consists of a single line with four integers x_1, y_1, x_2, y_2 ($0 \leq x_1, y_1, x_2, y_2 \leq 9$) where (x_1, y_1) is the initial configuration of the lock and (x_2, y_2) is the unlocked configuration.

Output

For each test case print the minimum number of moves Joon must make to open the refrigerator door, each on its own line.

Sample Input	Sample Output

2
0 9 1 3
4 5 7 2
	5
	6

Problem H
Consecutive Numbers

The Association of Counterintelligence Measures (ACM) is at it again. This time they're trying to generate some random numbers. After generating many sequences of 'random' numbers they began to wonder if the sequences are truly random. Again, your boss has a wonderful idea (not really): given some value K, a *truly* random sequence will have K consecutive numbers that sum to a very large value. If the largest K consecutive sum is very big, then the sequence is indeed truly random.

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with two integers N and K ($1 \leq K \leq N \leq 1,000,000$); N is the amount of random numbers that were generated and K is as described in the problem statement. The second line of each test case will contain N space separated positive integers which will be no greater than $1,000$.

Output

For each test case print the maximum sum we can achieve from summing K consecutive numbers.

Sample Input

```
2
5 3
1 5 3 2 5
10 4
12 43 49 20 20 3 0 19 20 30
```

Sample Output

```
10
132
```

Problem I
Food Selection

Josh is hungry. Starving in fact. He goes to his favorite restaurant, and to his surprise, is given the option to select all of his favorite foods from the menu. Each item on the menu has a corresponding cost and amount. Some items have the option of increasing their portion size for an additional cost. Unfortunately Josh is only allowed to have at most one of each dish, and if he opts to increase the portion size for a specific dish (if it's allowed) it may only be done once. Given Josh's budget, what is the maximum amount of food he can order?

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with a line containing two space separated integers N $(1 \leq N \leq 50)$ and K $(1 \leq K \leq 10,000)$ which are the number of favorite dishes Josh has in this specific test case and his budget, respectively. Following will be N lines each specifying a dish. If the dish doesn't have the option to increase its portion size it will be of the form 'x y' where x is the cost for that item and y is the amount of food in that dish $(1 \leq x, y \leq 1,000)$. If the dish has an option to increase its portion size it will be of the form 'x y w h' where x and y are as before, but w is the cost to increase the size of the dish by h amount $(1 \leq w, h \leq 1,000)$.

Output

For each test case print the most amount of food Josh can eat that fits within his budget. The output for each test case should be on its own line.

Sample Input

```
2
3 20
5 5
10 8 5 3
15 6
2 15
10 9 3 4
5 7
```

Sample Output

```
16
16
```

Problem J
Automated Dispatch

In order to automate shipping, your manager at Gainesville Computer Company (GCC) has assigned you the task of sending a dispatch message to the driver once a truck is completely loaded with the required type goods.

The truck leaves the GCC warehouse whenever K laptops have been loaded into it. A scanner is placed in the assembly line to automatically identify items being loaded into the truck. GCC ships out many different products, each of which is identified by an ID. After each item is loaded, its ID (not necessarily unique) is recorded. Given N IDs can you determine whether we should tell the driver to depart or not?

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with two integers N and K ($1 \leq K \leq N \leq 1,000,000$), which are the same as described above. The next line will contain N space delimited integers, each corresponding to the ID of a product loaded on the truck. If an ID is not positive it corresponds to a laptop; otherwise it is not a laptop.

Output

For each test case print "DISPATCH" if the required number of items is loaded, otherwise print "WAIT". The output for each test case should be on its own line.

Sample Input	**Sample Output**

`2`
`4 3`
`-1 -3 4 2`
`4 2`
`0 -1 2 1`
	`WAIT`
	`DISPATCH`