Programming Day,

19. Juni 2011

A 12004 - Bubble Sort

Check the following code which counts the number of swaps of bubble sort.

```
int findSwaps( int n, int a[] )
{
    int count = 0, i, j, temp, b[100000];

    for( i = 0; i < n; i++ ) {
        b[i] = a[i];
    }
    for( i = 0; i < n; i++ ) {
        for( j = 0; j < n - 1; j++ ) {
            if( b[j] > b[j+1] ) {
                temp = b[j];
                b[j] = b[j+1];
                b[j+1] = temp;

                count++;
            }
        }
    }
    return count;
}
```

You have to find the average value of 'count' in the given code if we run findSwaps() infinitely many times using constant 'n' and each time some random integers (from 1 to n) are given in array a[]. You can assume that the input integers in array a[] are distinct.

Input

Input starts with an integer T ($\leq$ 1000), denoting the number of test cases. Each test case contains an integer n ($1 \leq n \leq 10^5$) in a single line.

Output

For each case, print the case number and the desired result. If the result is an integer, print it. Otherwise print it in 'p/q' form, where p and q are relative prime.

Sample Input

```
2
1
2
```

Sample Output

```
Case 1: 0
Case 2: 1/2
```

B 12011 - Complete the Set

Archaeologists have made a discovery on the Temple of Topology. The temple was once used as a place for ritual ceremony thousands of years ago. Among the relics that were unearthed, a scroll of parchment raised the interest of scientists. The parchment contained many numbers written in ancient symbols.

By decrypting the words carved on a stone, scientists know that these numbers form an interesting set of integers satisfying the following two properties:

1. Bitwise AND any number of integers from the set result in an integer in that set again.
2. Bitwise OR any number of integers from the set result in an integer in that set again.

As the parchment is extremely old, some part of it were broken and the numbers were lost. Now you job is to complete the original set from the remaining integers such that the size of the set is as small as possible.

Input

The input contains several test cases. The total number of test cases is less than 1100. Each test case begins with a line containing an integer $n(n > 1)$. The 16 following line contains n integers a_i ($0 \leq a_i < 2^{18}$), the remaining integers on the parchment. The integers are distinct.

Output

For each test case, output one line containing a single integer, the minimal number of additional integers to make the set complete. If these numbers are already a complete set, print `0'.

Sample Input

```
4
5
0 1 3 5 7
2
2 4
3
3 7 11
3
1 2 4
```

Sample Output

```
Case #1: 0
Case #2: 2
Case #3: 1
Case #4: 5
```

C 11992 - Fast Matrix Operations

There is a matrix containing at most 10^6 elements divided into r rows and c columns. Each element has a location (x,y) where 1<=x<=r,1<=y<=c. Initially, all the elements are zero. You need to handle four kinds of operations:

```
1 x1 y1 x2 y2 v
```

Increment each element (x,y) in submatrix (x1, y1, x2, y2) by v (v>0)

```
2 x1 y1 x2 y2 v
```

Set each element (x,y) in submatrix (x1, y1, x2, y2) to v

```
3 x1 y1 x2 y2
```

Output the summation, min value and max value of submatrix (x1, y1, x2, y2)

In the above descriptions, submatrix (x1,y1,x2,y2) means all the elements (x,y) satisfying x1<=x<=x2 and y1<=x<=y2. It is guaranteed that 1<=x1<=x2<=r, 1<=y1<=y2<=c. After any operation, the sum of all the elements in the matrix does not exceed 10^9.

Input

There are several test cases. The first line of each case contains three positive integers r, c, m, where m (1<=m<=20,000) is the number of operations. Each of the next m lines contains a query. There will be at most twenty rows in the matrix. The input is terminated by end-of-file (EOF). The size of input file does not exceed 500KB.

Output

For each type-3 query, print the summation, min and max.

Sample Input

```
4 4 8
1 1 2 4 4 5
3 2 1 4 4
1 1 1 3 4 2
3 1 2 4 4
3 1 1 3 4
2 2 1 4 4 2
3 1 2 4 4
1 1 1 4 3 3
```

Output for the Sample Input

```
45 0 5
78 5 7
69 2 7
39 2 7
```

D 10278 - Fire Station

A city is served by a number of fire stations. Some residents have complained that the distance from their houses to the nearest station is too far, so a new station is to be built. You are to choose the location of the fire station so as to reduce the distance to the nearest station from the houses of the disgruntled residents.

The city has up to 500 intersections, connected by road segments of various lengths. No more than 20 road segments intersect at a given intersection. The location of houses and firestations alike are considered to be at intersections (the travel distance from the intersection to the actual building can be discounted). Furthermore, we assume that there is at least one house associated with every intersection. There may be more than one firestation per intersection.

The Input

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

The first line of input contains two positive integers: *f*,the number of existing fire stations (*f* <= 100) and *i*, the number of intersections (*i* <= 500). The intersections are numbered from 1 to *i* consecutively. *f* lines follow; each contains the intersection number at which an existing fire station is found. A number of lines follow, each containing three positive integers: the number of an intersection, the number of a different intersection, and the length of the road segment connecting the intersections. All road segments are two-way (at least as far as fire engines are concerned), and there will exist a route between any pair of intersections.

The Output

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.

You are to output a single integer: the lowest intersection number at which a new fire station should be built so as to minimize the maximum distance from any intersection to the nearest fire station.

Sample Input

```
1

1 6
2
1 2 10
2 3 10
3 4 10
4 5 10
5 6 10
6 1 10
```

Output for Sample Input

```
5
```

E 11984 - A Change in Thermal Unit

Measuring temperature and temperature differences are common task in many research and applications. Unfortunately, there exists more than one unit of measuring temperatures. This introduces a lot of confusion at times. Two popular units of measurements are Celsius(**C**) and Fahrenheit (**F**). The conversion of **F** from **C** is given by the formula:

$$F = \frac{9}{5} C + 32$$

In this problem, you will be given an initial temperature in **C** and an increase in temperature in **F**. You would have to calculate the new temperature in **C**.

Input

Input starts with an integer **T (≤ 100)**, denoting the number of test cases.

Each case contains a line with two integers **C** and **d (0 ≤ C, d ≤ 100)**, where **C** represents the initial temperature in Celsius and **d**represents the increase in temperature in Fahrenheit.

Output

For each case, print the case number and the new temperature in Celsius after rounding it to two digits after the decimal point.

Sample Input	Output for Sample Input
2 100 0 0 100	Case 1: 100.00 Case 2: 55.56

F 11966 - Galactic Bonding

Calvin likes to lie in a field and look at the night sky. Since he doesn't know any real star constellations, he makes them up: if two stars are close to each other, they must belong to the same constellation. He wants to name them all, but fears to run out of names. Can you help him and count how many constellations there are in the sky?

Two stars belong to the same constellation if distance between their projections on a two-dimensional sky plane isn't more than D units.

INPUT

There is a number of tests T ($T \leq 50$) on the first line. Each test case contains the number of stars N ($0 \leq N \leq 1000$) a real distance D ($0.00 \leq D \leq 1000.00$).
Next N lines have a pair of real coordinates X Y ($-1000.00 \leq X, Y \leq 1000.00$) for each star. Real numbers in the input will have at most 2 digits after a decimal point.

OUTPUT

For each test case output a single line `"Case T: N"`. Where T is the test case number (starting from 1) and N is the number of constellations.

SAMPLE INPUT

```
2
5 1.5
1.0 0.1
2.0 0.0
5.0 0.2
6.0 0.4
3.0 -0.1
3 4.0
121.12 254.06
645.04 301.85
912.49 568.96
```

SAMPLE OUTPUT

```
Case 1: 2
Case 2: 3
```

G 11936 - The Lazy Lumberjacks

Once upon a time in a far, far away forest, there was a team of lazy lumberjacks. Since they were too lazy to cut trees, they were always figuring out ways to sneak out of work. Their foreman, on the other side, was always trying to put them all to work.

After a lot of discussions the foreman and the lumberjacks came to an agreement: they will work, but only if the area of the forest assigned to each one was a triangle. If it was any other shape they will be free not to work that week. The idea was to give each lumberjack three numbers representing the length of each of the triangles side. If the numbers were correct and form a triangle, the lumberjacks had to work, else, they were free to leave and not work.

Since our lumberjacks are as cunning as they are lazy, they convince the foreman to let them determine the surface and the site in the forest were they will work. As a result, the lumberjacks keep passing the foreman sets of numbers that could not form the sides of a triangle. After a while, the foreman began to suspect and decide to write a program that validates the input of each lumberjack. Now when the lumberjacks decide to pass wrong numbers they get a fine of $1000.00 (more than a day's salary).

Your job is to write the program that the foreman has to use to determine if the numbers (all integers) passed by the lumberjacks can be the sides of a triangle. If they can, you have to print ``OK'' else you have to print ``Wrong!!''

Input

The input consist in a data set describing the numbers of that each lumberjack has passed to the foreman for the day: The data is formatted as follows: The first line is an integer N ($2 \leq N \leq 20$). Then follows N lines, each one containing three integers separated by a space.

Output

For each line in the input you have to find if the integers can represent the sides of a triangle. If they can you have to print ``OK'' for each line in the input, else you have to print ``Wrong!!''

Sample Input

```
6
1 2 3
3 2 5
3 4 5
6 6 1
3 3 3
7 3 10
```

Sample Output

```
Wrong!!
Wrong!!
OK
OK
OK
Wrong!!
```

H 11932 - Net Profit

Thomas has just designed a new game called "Net Profit". The game is played by two players on a "net" of business ventures, each of which offers a certain amount of profit (or loss) in dollars. The term "net" refers to the fact that ventures are connected to each other by randomly generated links (chosen in such a way that all the ventures are connected).

The first player may pick any venture to start, and he scores the associated profit or loss. This venture is now referred to as exhausted. Afterward, the players take turns exhausting ventures and collecting profits, following two simple rules:

1. An exhausted venture may not be selected again (by either player)
2. Only ventures connected to an already exhausted venture are eligible for exhaustion

The game ends once all the ventures are exhausted, and the winner is the player with the greatest profit (or smallest loss). With a given "net" of ventures and associated profits, Thomas would like to know the final outcome of the game assuming optimal play from both players.

Input

Input consists of several test cases. Each test case begins with an integer N ($1 \leq N \leq 16$), representing the number of ventures in the net. This is followed by a line containing N integers $p_1, p_2,..., p_N$; where p_k is the profit associated with venture k ($|p_k| \leq 1000$).

Next is a line containing a non-negative integer *M*, followed by *M* lines, each describing a link in the net. Each link description consists of two integers *a* and *b* ($1 \leq a, b \leq N$, $a \neq b$), which means that ventures *a* and *b* are linked.

You may assume that the described net connects all the ventures in one component, and that a given link is described at most once (so if link*a b* is given, link *b a* will not be).

The input is terminated by a line containing `0' which should not be processed.

Output

For each test case, output a line with the final result and score of the game assuming optimal play by both players (see the sample output for details).

Sample Input

```
2
25 -20
1
1 2
3
15 15 -5
2
1 2
2 3
2
30 30
1
1 2
0
```

Sample Output

```
First player wins! 25 to -20.
Second player wins! 15 to 10.
Tie game! 30 all.
```

I 11960 - Divisor Game

Steve is playing a game with numbers. He picks up a random positive number N and finds the largest positive number not bigger than N that has the most divisors. As N becomes larger it's more and more difficult for Steve to avoid mistakes when counting the divisors and he asks you to write a program. You argue that it is a very easy task to just find the divisors and suggest that you could solve the original task of Steve as well.

INPUT

You are given a number of tests T ($T \leq 50000$). Each test on a single line specifies a number N ($1 \leq N \leq 10^6$).

OUTPUT

You need to find the largest number not bigger than N that has the most divisors. For each test output one line containing the answer to the game.

SAMPLE INPUT

```
3
1
10
37
```

SAMPLE OUTPUT

```
1
10
36
```

J 11554 - Hapless Hedonism

Bob is a world-renowned stick collector. His most prized stick possessions include:

- an Arctic Redwood branch from a hike near Dawson City,
- a Desert Pine stick from a visit to the Grand Canyon, and
- a Chinese Arbour twig from an adventure into Tibet.

Bob collects sticks in a peculiar way. He will only accept a new stick into his collection if its length is exactly length n+ 1 cm where n is the number of sticks currently in his collection. This implies his collection of n sticks contains exactly one stick of length 1 cm through n cm.

One day Alice visited Bob to inspect his stick collection (upon Bob's insistence of course). Alice wasn't particularly interested in Bob's excessive descriptions and needed a quick conversation changer. Cleverly, she posed the following question to Bob: "If you are allowed to take any 3 sticks from your collection, how many different triangles can you make?"

Can you help Bob answer the question so he can get back to telling Alice about his sticks?

Program Input

The input will begin with t ($1 \leq t \leq 1000$), the number of test cases. Each test case will contain an integer n ($3 \leq n \leq 1000000$), the number of sticks in Bob's collection. (Recall if Bob has n sticks, then he has exactly one stick of each of the lengths from 1 cm through n cm.)

Program Output

For each test case, output on a line the number of different triangles you can make with Bob's sticks. Triangles X and$Y$ are different if there is at least one stick in X that is not in Y. A triangle has area strictly greater than 0.

Sample Input & Output

INPUT

```
3
3
4
10
```

OUTPUT

```
0
1
50
```

K 10432 - Polygon Inside A Circle

Consider a polygon of equal sides inside a circle as shown in the figure below.

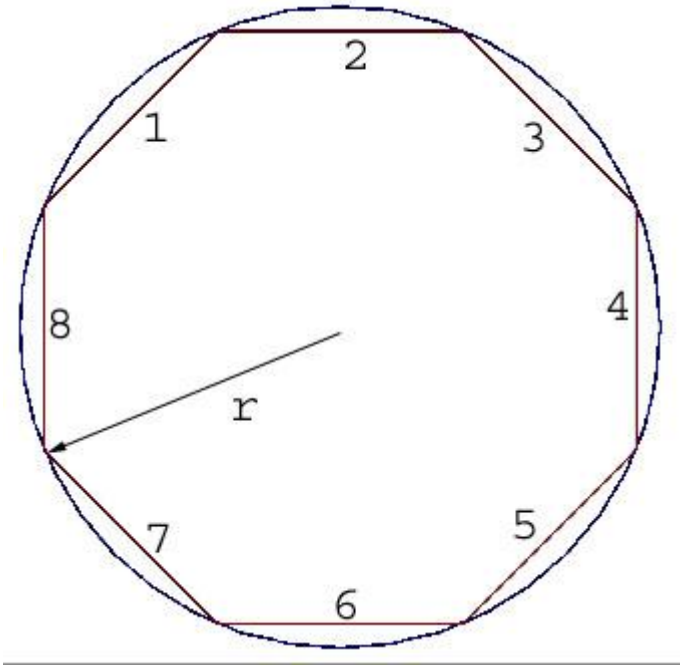

Figure: The regular polygon inside a circle

Given the radius of the circle and number of sides. You have to find the area of the polygon.

The Input

In each line there will be two numbers indicating the radius **`r'** (0<**r**<20000) and the number of sides of the polygon **`n'** (2<**n**<20000) respectively. Input is terminated by `EOF'.

The Output

Output the area in each line. The number must be rounded to the third digit after the decimal point.

Sample Input

```
2 2000
10 3000
```

Sample Output

```
12.566
314.159
```

L 11793 - Electoral Districts

Background

In a certain democracy, the party in the government is reorganizing the electoral districts in order to benefit them in the next election.

One delegate is elected for each district. And, as the regime is *de facto* a two-party system, they just need one more vote than the opposition for each district.

The Problem

The country is divided into a square mesh of NxN zones; these zones are grouped into electoral districts. There are N districts, and each district is composed by N zones. The zones of a district are adjacent, that is, we can reach all the zones of a same district by moving to the left, right, up or down.

Suppose the party in the government is A, and the party in the opposition is B. From previous elections, we know the number of votes for each party in each zone. Two NxN matrices are available, one with the number of votes for party A, and another one for the votes for party B.

For each district, we take the sum of the votes of its zones. Then, the party with more total votes in that district gains a delegate. If the number of votes is equal for A and B, then no delegate is assigned.

Party A has hired you to compute the largest possible difference they can obtain -- in number of delegates-- with an adequate arrangement of the districts.

The Input

The input begins with a line where the number of test cases (T) is indicated. The data for each test case appear in successive lines. For each test case, the first line contains the dimension of the country (N, with maximum value 5, and the country has N by N zones). Following, there are 2N lines, each line with N integers between 1 and 1000, separated by a space. The first N lines contain the votes for party A, and the other N lines the votes for party B.

The Output

The output consists of T lines, one for each test case. For each case, the maximum achievable difference between the number of delegates obtained by party A with respect to party B is represented.

Sample Input

```
4
2
2 3
2 4
3 1
2 3
3
2 3 4
1 3 2
2 3 5
3 4 1
3 2 1
2 1 3
3
1 2 1
2 1 2
1 2 1
2 1 2
1 2 1
2 1 2
3
1 1 1
1 1 1
1 1 1
2 2 2
2 2 2
2 2 2
```

Sample Output

```
2
2
1
-3
```