HKUST 2008 FALL

Programming Contest

Sep. 14, 2008

Hong Kong University of
Science and Technology

Contest Rules & Details:

1. The programming languages to be used in this contest are C/C++ and JAVA. The contestants use PC^2 to submit their source codes to the judge and the source codes are compiled by Visual Studio C++ 6.0, Visual Studio C++ 2005 or JAVA.

2. In this contest, the contestants are given six to seven programming problems. The goal is to solve as many problems as possible. For those who solve the same number of problems, the one with lower score wins. (The scoring system will be explained below.)

3. **The contestant should read the input and write the output via standard I/O**. The contestants can assume that all test cases are of the format as stated in the problem statements. i.e. No exception handling is needed.

4. The correctness of each submission is judged by inputting test cases into the submitted program. The submission is regarded as correct if its outputs match completely with the model outputs. The submission is judged as correct or wrong. No partial credit is given.

5. The contestants can re-submit another source code after previous wrong submissions.

6. Unless specified in the problem statements, **all programs should not run for more than 5 seconds** (in most cases a "correct" implementation will run far less than 5 seconds).

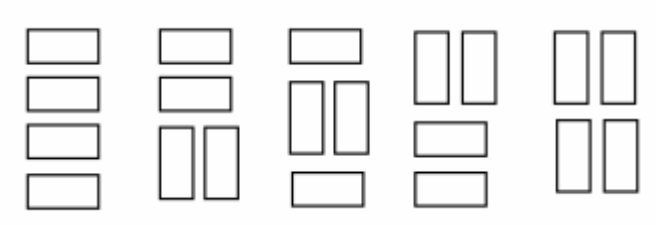
7. The contestants are ranked firstly by the number of problems solved, and secondly the total time spent on solving the problems. Time spent on solving one problem is the time between the start of contest and the submission of the correct implementation of that problem. For each problem you solved, a penalty of 20 minutes will be added to your score for each wrong submission of that problem.

8. The contestants are allowed to bring any hard copies of books, notes, references, dictionaries and sketch papers to the contest site. Electronic devices are forbidden.

Problem A -- Grid Dominoes

Problem

We wish to tile a grid 4 units high and N units long with rectangles (dominoes) 2 units by one unit (in either orientation). For example, the figure shows the five different ways that a grid 4 units high and 2 units wide may be tiled.



Write a program that takes as input the width, W, of the grid and outputs the number of different ways to tile a 4-by-W grid.

Input

The first line of input contains a single integer N, ($1 \leq N \leq 1000$) which is the number of datasets that follow.

Each dataset contains a single decimal integer, the width, W ($1 \leq W \leq 19$), of the grid for this problem instance.

Output

For each problem instance, there is one line of output: The problem instance number as a decimal integer (start counting at one), a single space and the number of tilings of a 4-by-W grid. The values of W will be chosen so the count will fit in a 32-bit integer.

Sample Input

```
3
2
3
7
```

Sample Output

```
1 5
2 11
3 781
```

Problem B -- Zig-Zag Sequence

Problem

A sequence of numbers is called a zig-zag sequence if the differences between successive numbers strictly alternate between positive and negative. The first difference (if one exists) may be either positive or negative. A sequence with fewer than two elements is trivially a zig-zag sequence.

For example, 1,7,4,9,2,5 is a zig-zag sequence because the differences (6,-3,5,-7,3) are alternately positive and negative. In contrast, 1,4,7,2,5 and 1,7,4,5,5 are not zig-zag sequences, the first because its first two differences are positive and the second because its last difference is zero.

Given a sequence of integers, sequence, please find the length of the longest subsequence of sequence that is a zig-zag sequence. A subsequence is obtained by deleting some number of elements (possibly zero) from the original sequence, leaving the remaining elements in their original order.

Input

The first line contains one integer C, which is the number of datasets that follow..
For each dataset, the first line contains one integer N ($1 \leq N \leq 50$), the number of integers in the sequence. The second line contains N integers, represent the input sequence in order.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, the length of the longest subsequence that is a zig-zag sequence.

Sample Input

```
1
9
1 2 3 4 5 6 7 8 9
```

Sample Output

```
1 2
```

Problem C -- Binary Codes

Problem

Consider a binary string ($b_1 ... b_N$) with N binary digits. Given such a string, the matrix of Figure 1 is formed from the rotated versions of the string.

$b_1\ b_2\ ...\ b_{N-1}\ b_N$
$b_2\ b_3\ ...\ b_N\ \ b_1$
...
$b_{N-1}\ b_N\ ...\ b_{N-3}\ b_{N-2}$
$b_N\ b_1\ ...\ b_{N-2}\ b_{N-1}$

Figure 1. The rotated matrix

Then the rows of the matrix are sorted in alphabetical order, where '0' is before '1'. You are to write a program which, given the last column of the sorted matrix, finds the first row of the sorted matrix.
As an example, consider the string (00110). The sorted matrix is

0 0 0 1 1
0 0 1 1 0
0 1 1 0 0
1 0 0 0 1
1 1 0 0 0

and the corresponding last column is (1 0 0 1 0). Given this last column your program should determine the first row, which is (0 0 0 1 1).

Input

The first line of input contains a single integer C, ($1 \leq C \leq 10$) which is the number of datasets that follow.
For each dataset, the first line contains one integer N ($1 \leq N \leq 1000$), the number of binary digits in the binary string. The second line contains N integers, the binary digits in the last column from top to bottom.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, followed by N integers: the binary digits in the first row from left to right.

Sample Input

```
1
5
1 0 0 1 0
```

Sample Output

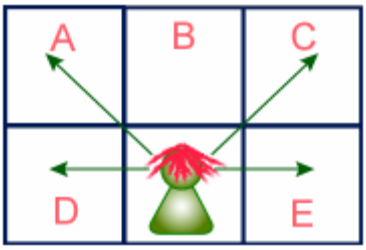
```
1 0 0 0 1 1
```

Problem D -- No Cheating

Problem

A local high school is going to hold a final exam in a big classroom. However, some students in this school are always trying to see each other's answer sheet during exams! The classroom can be regarded as a rectangle of M rows by N columns of unit squares, where each unit square represents a seat.

The school principal decided to set the following rule to prevent cheating: Assume a student is able to see his left, right, upper-left, and upper-right neighbors' answer sheets. The assignment of seats must guarantee that nobody's answer sheet can be seen by any other student.



As in this picture, it will not be a good idea to seat anyone in A, C, D, or E because the boy in the back row would be able to see their answer sheets. However, if there is a girl sitting in B, he will not be able to see her answer sheet.

Some seats in the classroom are broken, and we cannot put a student in a broken seat.
The principal asked you to answer the following question: What is the maximum number of students that can be placed in the classroom so that no one can cheat?

Input

The first line of input gives the number of cases, C. C test cases follow. Each case consists of two parts. The first part is a single line with two integers M and N ($1 \leq M,N \leq 80$): The height and width of the rectangular classroom. The second part will be exactly M lines, with exactly N characters in each of these lines. Each character is either a '.' (the seat is not broken) or 'x' (the seat is broken, lowercase x).

Output

For each test case, output one line containing "Case #X: Y", where X is the case number, starting from 1, and Y is the maximum possible number of students that can take the exam in the classroom.

Sample Input

```
4
2 3
...
...
2 3
x.x
xxx
2 3
x.x
```

```
x.x
10 10
....x.....
..........
..........
..x.......
..........
x...x.x...
.........x
...x......
........x.
.x...x....
```

Sample Output

```
Case #1: 4
Case #2: 1
Case #3: 2
Case #4: 46
```

Problem E -- Square

Problem
Given a set of sticks of various lengths, is it possible to join them end-to-end to form a square?

Input
The first line of input contains N, the number of test cases. Each test case begins with an integer 4 <= M <= 20, the number of sticks. M integers follow; each gives the length of a stick - an integer between 1 and 10,000.

Output
For each case, output a line “Case #X: Y”, where X is the case number, starting from 1. Y is "yes" if it is possible to form a square; otherwise output "no".

Sample Input
3
4 1 1 1 1
5 10 20 30 40 50
8 1 7 2 6 4 4 3 5

Sample Output
Case #1: yes
Case #2: no
Case #3: yes

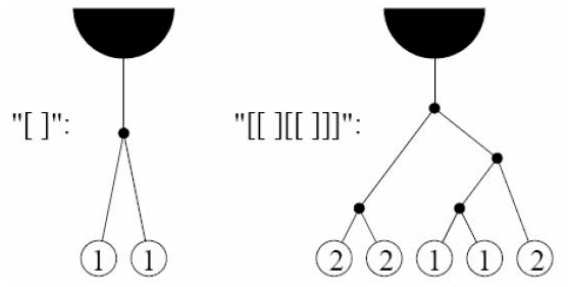
Problem F -- Monkey Vines

Problem

Deep in the Amazon jungle, exceptionally tall trees grow that support a rich biosphere of figs and juniper bugs, which happen to be the culinary delight of brown monkeys.

Reaching the canopy of these trees requires the monkeys to perform careful navigation through the tall tree's fragile vine system. These vines operate like a see-saw: an unbalancing of weight at any vine junction would snap the vine from the tree, and the monkeys would plummet to the ground below. The monkeys have figured out that if they work together to keep the vines properly balanced, they can all feast on the figs and juniper bugs in the canopy of the trees.

A vine junction supports exactly two sub-vines, each of which must contain the same number of monkeys, or else the vine will break, leaving a pile of dead monkeys on the jungle ground. For purposes of this problem, a vine junction is denoted by a pair of matching square brackets [ ], which may contain nested information about junctions further down its sub-vines. The nesting of vines will go no further than 25 levels deep.



You will write a program that calculates the minimum number of monkeys required to balance a particular vine configuration. There is always at least one monkey needed, and, multiple monkeys may hang from the same vine.

Input

The first line of input contains a single integer N, ($1 \leq N \leq 1000$) which is the number of datasets that follow. Each dataset consists of a single line of input containing a vine configuration consisting of a string of [and] characters as described above. The length of the string of [and] will be greater than or equal to zero, and less than or equal to 150.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and the minimum number of monkeys required to reach the canopy successfully. Assume that all the hanging vines are reachable from the jungle floor, and that all monkeys jump on the vines at the same time.

Sample Input

```
3
[]

[[][[]]]
```

Sample Output

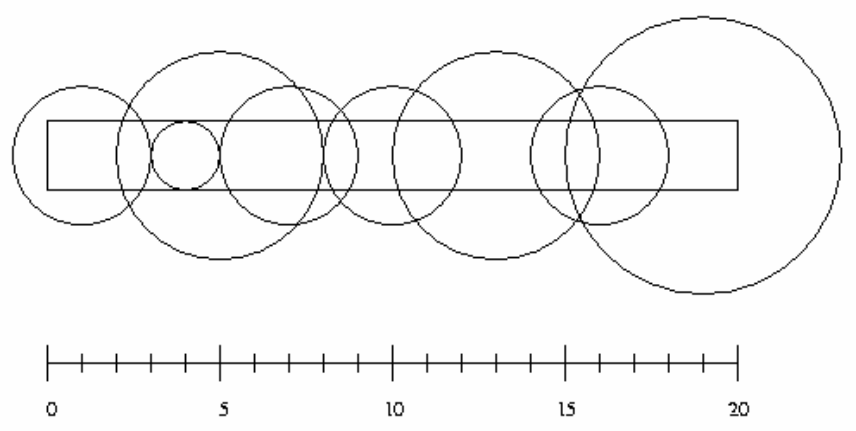
```
1 2
2 1
3 8
```

Problem G -- Watering Grass

Problem

n sprinklers are installed in a horizontal strip of grass *l* meters long and *w* meters wide. Each sprinkler is installed at the horizontal center line of the strip. For each sprinkler we are given its position as the distance from the left end of the center line and its radius of operation.

What is the minimum number of sprinklers to turn on in order to water the entire strip of grass?



Input

The first line of input gives the number of cases, C. C test cases follow. The first line for each case contains integer numbers *n*, *l* and *w* with *n* <= 1000. The next *n* lines contain two integers giving the position of a sprinkler and its radius of operation. (The picture above illustrates the first case from the sample input.)

Output

For each test case, output one line containing "Case #X: Y", where X is the case number, starting from 1, Y is the minimum number of sprinklers needed to water the entire strip of grass. If it is impossible to water the entire strip output -1.

Sample Input

```
3
8 20 2
5 3
4 1
1 2
7 2
10 2
13 3
16 2
19 4
3 10 1
3 5
9 3
```

```
6 1
3 10 1
5 3
1 1
9 1
```

Sample Output

```
Case #1: 6
Case #2: 2
Case #3: -1
```