Code Jam 2010 - Round 2

Elegant Diamond

Problem

The king has hired you to make him an elegant diamond. An elegant diamond is a two-dimensional object made out of digits that's symmetric about a horizontal and a vertical axis. For example, the following four shapes are elegant diamonds:

```
   2       8      3     7
  3 3     8 8    2 2
 4 1 4     8      3
  3 3
   2
```

These three shapes are diamonds, but are not elegant:

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

These three shapes are not diamonds:

```
  1     2     8   8
 1 1   222      0
        2     00000
```

The king will start by giving you a diamond, which may not be elegant. Your job is to make it elegant by *enhancing* it, adding digits on to make a bigger diamond. Because you don't want to spend too much money, you want to do it with as little *cost* as possible.

Definitions

A **diamond of size *k*** is 2k-1 lines of digits, 0-9, separated by single spaces, organized in the following way:

- Line i (1 ≤ i ≤ k) contains k-i spaces, then i digits separated by single spaces.
- Line i (k < i < 2k) contains i-k spaces, then 2k-i digits separated by single spaces.

An **elegant diamond of size *k*** is a diamond of size k with the following two symmetry properties:

- Horizontal symmetry: Let c_i be the number of digits on line i. The j^{th} digit on line i (where j=1 for the first digit) must be the same as the c_i+1-j^{th} digit.
- Vertical symmetry: The j^{th} digit on line i (where i=1 for the first line) must be the same as the j^{th} digit on line 2k-i.

A diamond of size k can be **enhanced** by adding digits to it. The result of enhancing a diamond of size k has the following properties:

- The result is a diamond of size ≥ k.

- The original diamond is part of the result. In other words, there exist some X and some Y such that, for all values of i and j such that the jth character of the ith line of the original is a digit (as opposed to a space), the j+Xth character on the i+Yth line of the result is also a digit and it's the same as the jth character on the ith line of the original.

The **cost** of enhancing a diamond is equal to the number of digits in the result of the enhancement, minus the number of digits in the original diamond.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case consists of a single integer **k** in a line on its own, followed by a diamond of size **k**.

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 minimum cost required to enhance the given diamond into an elegant diamond. If the diamond is already elegant, y=0.

Limits

Memory limit: 1GB.
1 ≤ **T** ≤ 100.

Small dataset (Test set 1 - Visible)

Time limit: 30 seconds.
1 ≤ **k** ≤ 10.

Large dataset (Test set 2 - Hidden)

Time limit: 60 seconds.
1 ≤ **k** ≤ 51.

Sample

Sample Input

```
4
1
0
2
 1
2 2
 1
2
 1
1 2
 1
3
  1
 6 3
9 5 5
```

Sample Output

```
Case #1: 0
Case #2: 0
Case #3: 5
Case #4: 7
```

```
  6 3
   1
```

Explanation

There are four cases. The first two cases start as elegant diamonds of size 1 and 2, respectively, and don't need to be enhanced; so the cost is 0. The third case can be enhanced to look like:

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

There are several possible enhancements, but this is one with the lowest possible cost, which is 5. In the fourth case we can enhance the diamond into the following elegant diamond:

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

...for a cost of 7.