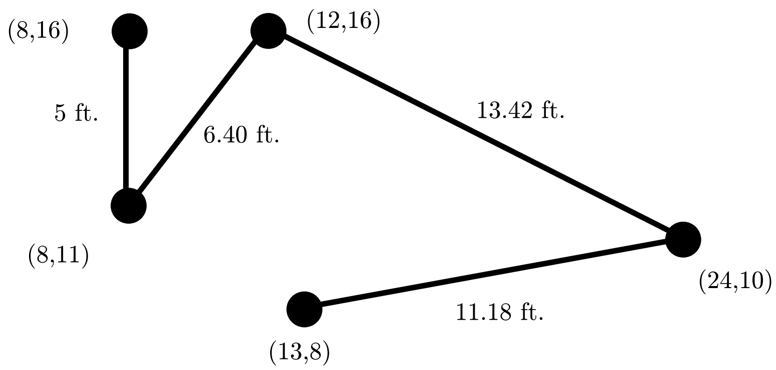
216 Getting in Line

Computer networking requires that the computers in the network be linked.

This problem considers a "linear" network in which the computers are chained together so that each is connected to exactly two others except for the two computers on the ends of the chain which are connected to only one other computer. A picture is shown below. Here the computers are the black dots and their locations in the network are identified by planar coordinates (relative to a coordinate system not shown in the picture).

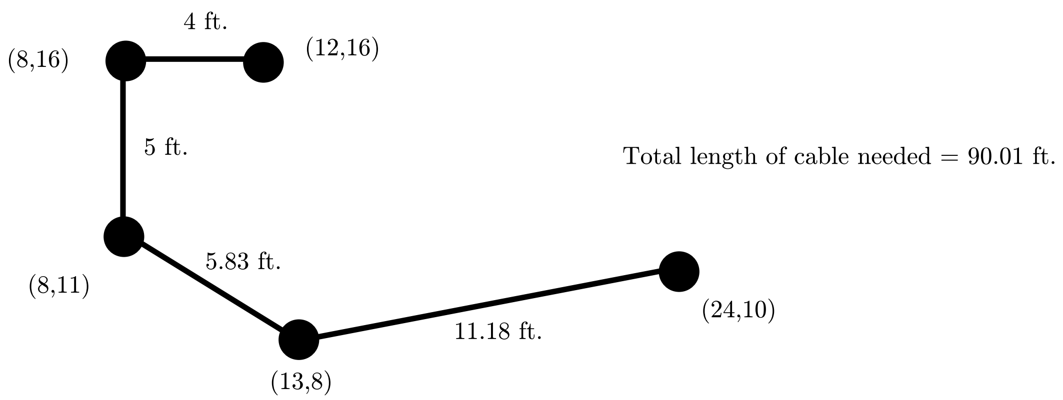
Distances between linked computers in the network are shown in feet.



For various reasons it is desirable to minimize the length of cable used.

Your problem is to determine how the computers should be connected into such a chain to minimize the total amount of cable needed. In the installation being constructed, the cabling will run beneath the floor, so the amount of cable used to join 2 adjacent computers on the network will be equal to the distance between the computers plus 16 additional feet of cable to connect from the floor to the computers and provide some slack for ease of installation.

The picture below shows the optimal way of connecting the computers shown above, and the total length of cable required for this configuration is (4+16)+ (5+16) + (5.83+16) + (11.18+16) = 90.01 feet.



Input

The input file will consist of a series of data sets. Each data set will begin with a line consisting of a single number indicating the number of computers in a network. Each network has at least 2 and at most 8 computers. A value of 0 for the number of computers indicates the end of input.

After the initial line in a data set specifying the number of computers in a network, each additional line in the data set will give the coordinates of a computer in the network. These coordinates will be integers in the range 0 to 150. No two computers are at identical locations and each computer will be listed once.

Output

The output for each network should include a line which tells the number of the network (as determined by its position in the input data), and one line for each length of cable to be cut to connect each adjacent pair of computers in the network. The final line should be a sentence indicating the total amount of cable used.

In listing the lengths of cable to be cut, traverse the network from one end to the other. (It makes no difference at which end you start.) Use a format similar to the one shown in the sample output, with a line of asterisks separating output for different networks and with distances in feet printed to 2 decimal places.

Sample Input

```
6
5 19
55 28
38 101
28 62
111 84
43 116
5
11 27
84 99
142 81
88 30
95 38
3
132 73
49 86
72 111
0
```

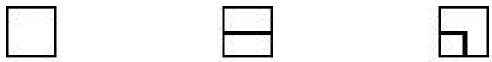
Sample Output

```
**********************************************************
Network #1
Cable requirement to connect (5,19) to (55,28) is 66.80 feet.
Cable requirement to connect (55,28) to (28,62) is 59.42 feet.
Cable requirement to connect (28,62) to (38,101) is 56.26 feet.
Cable requirement to connect (38,101) to (43,116) is 31.81 feet.
Cable requirement to connect (43,116) to (111,84) is 91.15 feet.
Number of feet of cable required is 305.45.
**********************************************************
Network #2
Cable requirement to connect (11,27) to (88,30) is 93.06 feet.
Cable requirement to connect (88,30) to (95,38) is 26.63 feet.
Cable requirement to connect (95,38) to (84,99) is 77.98 feet.
```

```
Cable requirement to connect (84,99) to (142,81) is 76.73 feet.
Number of feet of cable required is 274.40.
**********************************************************
Network #3
Cable requirement to connect (132,73) to (72,111) is 87.02 feet.
Cable requirement to connect (72,111) to (49,86) is 49.97 feet.
Number of feet of cable required is 136.99.
```

10582 ASCII Labyrinth

We are trying to construct a labyrinth on a board of size $m \times n$. Initially, on each square of the board we find a piece of thin plywood of size 1×1 with one of the following three patterns painted on it.



While constructing the labyrinth we may turn the pieces arbitrarily but each piece must exactly cover a square of the board. We are not allowed to move a piece to another square of the grid.

Given an initial board covered with the pieces, we would like to turn the pieces in such a way that the patterns on the pieces form at least one polygonal curve connecting the top left corner square of the board with the bottom right square of the board. The picture below presents an initial state of a board of size 4×6 and a labyrinth constructed from the board in which the above stated goal has been achieved.

Your task is to read a description of the initial board with the pieces placed on it and to decide whether one can turn the pieces in such a way that the patterns form a line connecting some edge of the top left square and some edge of the bottom right square of the board.

Input

The first line of input contains a number c giving the number of cases that follow. The test data for each case start with two numbers m and n giving the number of rows and columns on the board. The remaining lines form an ASCII rendition of the initial board with the pieces placed on squares. The characters used in the rendition are `+`, `-`, `|`, `*` and space. See the sample input for the format. The size of the input board will be such that $m \times n \leq 64$.

Output

For each case print in a single line how many different paths exist in the solutions to the labyrinth problem in the format shown below.

Sample Input

```
1
4 6
+---+---+---+---+---+---+
|   |   |   |   |   |   |
|***|***|** |** |** |***|
|   |   | * | * | * |   |
+---+---+---+---+---+---+
|   |   |   |   |   |   |
```

```
|   |** |** |***|** |** |
|   | * | * |   | * | * |
+---+---+---+---+---+---+
|   |   |   |   |   |   |
|***|** |***|***|***|** |
|   | * |   |   |   | * |
+---+---+---+---+---+---+
|   |   |   |   |   |   |
|** |   |***|   |** |** |
| * |   |   |   | * | * |
+---+---+---+---+---+---+
```

Sample Output

```
Number of solutions: 2
```

UVa Online Judge

291 The House Of Santa Claus

In your childhood you most likely had to solve the riddle of the house of Santa Claus. Do you remember that the importance was on drawing the house in a stretch without lifting the pencil and not drawing a line twice? As a reminder it has to look like shown in Figure 1.

Well, a couple of years later, like now, you have to "draw" the house again but on the computer. As one possibility is not enough, we require *all* the possibilities when starting in the lower left corner. Follow the example in Figure 2 while defining your stretch.

All the possibilities have to be listed in the outputfile by increasing order, meaning that `1234...` is listed before `1235...` .

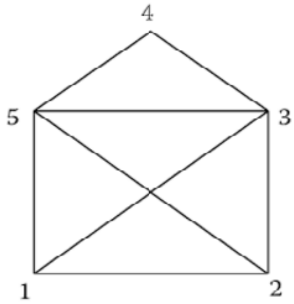


Figure 1: The House of Santa Claus

Output

So, an outputfile could look like this:

```
12435123
13245123
...
15123421
```

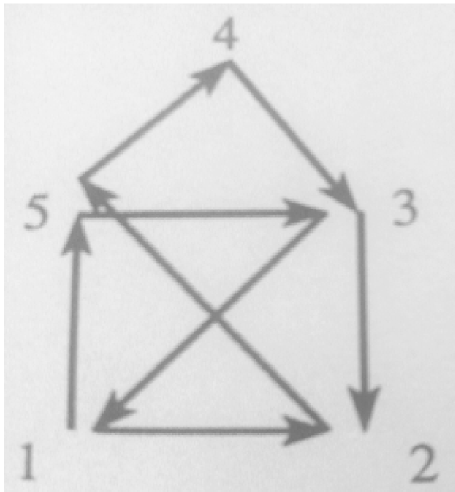


Figure 2: This Sequence would give the Outputline `153125432`

11961 DNA

Alan was comleting his internship at chemistry faculty. At the beginning internship seemed to be easy. However one day dean of the faculty of chemistry have given John a certain task to solve. Alan was asked to perform DNA and protein modelling using free OpenEye software kit.

As Alan like all programmers is very lazy, he would like to use computer as much as possible. So now he needs to generate all possible DNA mutations and share them to OpenEye software. Alan decided to use computer not only for OpenEye software but also for mutation generation purpose.

At the beginning of the internship Alan studying a little bit of chemistry science, so he knows that DNA consisting of 4 elements (**A**denine, **G**uanine, **C**ytosine, and **T**hymine) and can be described as a sequence of four letters (for example: `GATCC`). The K-th mutation of inital DNA sequence of length N is called a sequence that can be produced by replacing (possibly to the same nucleotide) exactly K elements of the sequence (for example `GAT` is the 1-st mutation of `GGT` and the 2-nd mutation of `TTT`).

Alan is given initial DNA sequence and maximal power of its possible mutataion. Can you produce all possible mutated DNA sequences for Alan?

Input

The number of tests T ($T \leq 50$) is given on the first line. Each test case if described by two lines: one the first is number N ($N \leq 10$) and K ($K \leq 5$), on the second line is written DNA sequence of length N.

Output

For each test case print M — the number of different DNA mutations. After this print all mutated DNA sequences in alphabetical order. Refer to the sample output for details.

Sample Input

```
1
3 1
AAA
```

Sample Output

```
10
AAA
AAC
AAG
AAT
ACA
AGA
ATA
CAA
GAA
TAA
```

539 The Settlers of Catan

Within *Settlers of Catan*, the 1995 German game of the year, players attempt to dominate an island by building roads, settlements and cities across its uncharted wilderness.

You are employed by a software company that just has decided to develop a computer version of this game, and you are chosen to implement one of the game's special rules:

> When the game ends, the player who built the longest road gains two extra victory points.

The problem here is that the players usually build complex road networks and not just one linear path. Therefore, determining the longest road is not trivial (although human players usually see it immediately).

Compared to the original game, we will solve a simplified problem here: You are given a set of nodes (cities) and a set of edges (road segments) of length 1 connecting the nodes.

The longest road is defined as the longest path within the network that doesn't use an edge twice. Nodes may be visited more than once, though.

Example: The following network contains a road of length 12.

```
o        o -- o        o
 \      /      \      /
  o -- o        o -- o
 /      \      /      \
o        o -- o        o -- o
               \      /
                o -- o
```

Input

The input file will contain one or more test cases.

The first line of each test case contains two integers: the number of nodes n ($2 \leq n \leq 25$) and the number of edges m ($1 \leq m \leq 25$). The next m lines describe the m edges. Each edge is given by the numbers of the two nodes connected by it. Nodes are numbered from 0 to $n-1$. Edges are undirected. Nodes have degrees of three or less. The network is not neccessarily connected.

Input will be terminated by two values of 0 for n and m.

Output

For each test case, print the length of the longest road on a single line.

Sample Input

```
3 2
0 1
1 2
15 16
0 2
1 2
2 3
```

```
3 4
3 5
4 6
5 7
6 8
7 8
7 9
8 10
9 11
10 12
11 12
10 13
12 14
0 0
```

Sample Output

```
2
12
```

11065 A Gentlemen's Agreement

The mayor of Madman City has decided that the city's crossings need newer and better traffic lights. He has commissioned the company Light's Inc. to remove the old and install the new traffic lights, because it is a very renowned company and one of the leading producers of traffic lights. The people in charge of this task (a mathematician and an engineer) have decided to minimize the ensuing chaos when the traffic lights are replaced, which entails that not all traffic lights can be replaced simultaneously. On the other hand they do not want to replace the traffic lights one at a time as that would take too long. So they decided that if they replaced the traffic lights at one intersection they would not replace the traffic lights at those intersections which were directly connected to the intersection where the traffic light was being replaced.

Now the practical engineer was, of course, interested in the maximum number of intersections where the traffic lights could be replaced simultaneously, so he was looking for the largest set of intersections, where no two intersections in the set are connected by a road. The theoretical mathematician however was interested in the number of different sets of intersections in which no subset containing two intersections is connected by a road and it is not possible to add another road without violating the previous condition.

Input

The first line contains the number of cities which are to be processed. The following lines contain the descrition of the cities. A city consists of a number of intersections and of a number of roads which connect the intersections. The number of intersections i, $1 \leq i \leq 60$ and the number of roads r, $1 \leq r \leq 3600$ are on a line. The intersections are numbered from $0 \ldots i-1$. The following r lines contain the description of the roads. The description of a road consists of the two intersections it connects. No intersections will be connected directly by two different roads. Furthermore the graph representing the city's network of roads will be connected.

Output

For each each city, output the number of sets of intersections, where no two intersections in the set are connected by a road and it is not possible to add another intersection to the set without violating the first condition, on the first line. Print the number of intersections in the largest set on the second line.

Sample Input

```
3
4 4
0 1
1 2
2 3
3 0
6 12
0 1
0 2
0 3
0 4
1 2
```

```
2 4
4 3
3 1
1 5
2 5
3 5
4 5
8 12
0 1
0 2
0 6
1 3
1 7
2 3
2 4
3 5
4 5
4 6
5 7
6 7
```

Sample Output

```
2
2
3
2
6
4
```