Cornell Programming Competition
Fall 1999

Adam has recently bought a wonderful new house in the country. The house has a large, beautiful garden, much of which is occupied by a tree plantation. Oddly enough, the trees are planted on a square grid, where the grid spacing is **10 yards**. However, not all points on the grid are occupied by trees.

The plantation contains six kinds of trees: birches(`B`), chestnuts(`C`), elms(`E`), firs(`F`), maples(`M`), and oaks(`O`). As mentioned above, some of the grid points are empty(`-`).

All the problems will revolve around the tree plantation, and **they will all use the same input format**. While there is sort of a story line running through the problem set, it is not necessary to have solved the previous problems in order to be able to tackle the next one. You can solve the problems in any order you wish.

The first line of each file contains a single non-negative number. This is the number of **data sets** contained in the file.

Each data set has the following form: The first line contains two positive integers, `h` and `w` ($1 \leq$ `h` $\leq 100, 1 \leq$ `w` ≤ 100), which are the height and width of the garden. Then there are `h` lines. Each line is a string of `w` characters. These lines provide a map of the garden, showing which tree (if any) is at each grid point. The ith position of the jth string contains the tree that stands at position (j, i), or a '`-`' if there is no tree at this position.

The following is an example of such an input file and will be called `sample.in` We will use it repeatedly later on to illustrate the problem statements.

```
2
4 3
EEM
E-E
F--
--B
5 7
E------
-O---E-
--FFE--
-F---M-
M-----B
```

To illustrate the position notation, in data set 1, the maple tree is at position (1,3) and the fir tree is at position (3,1).

You may use either Java or C++. You do not need to use the same language on all problems. All programs must be console applications. You are to read input from standard input and write output to standard output. You are to perform the same processing on each data set. Data sets are independent. It is not necessary to read in all the data sets before writing any output.

Remember that you must use fixed file names. Your solution to Problem A must be called `proba.cpp` or `proba.java`; obvious modifications apply to the other problems. If your solution is not named properly, it will be returned as incorrect. For Java programmers, this means that you can have only one public class, and for Problem A is must be called `proba`.

Cornell Programming Competition

Fall 1999
Problem A – Counting Trees

Description

Adam would like an inventory of how many trees of the different kinds he owns. And not just that – the kind that he owns most of are supposed to be on top of the list.

Input / Output

See the introduction for the input format. No further assumptions are made on the input.

The output for each data set is 7 lines of text, followed by a blank line. The first line should say `Data set` followed by the data set number. The last 6 lines should list the types of trees, and the number of trees of each type, one per line. The list should be sorted by non-increasing number of trees. If two types of trees have the same number, they should be sorted in alphabetical order.

Sample

```
Data set 1:
Elm: 4
Birch: 1
Fir: 1
Maple: 1
Chestnut: 0
Oak: 0

Data set 2:
Elm: 3
Fir: 3
Maple: 2
Birch: 1
Oak: 1
Chestnut: 0
```

Cornell Programming Competition

Fall 1999
Problem B – Favorite Grove

Description

Adam was wondering which grove of his garden is his favorite. A *grove* is a rectangle with its upper-left corner at (a, b) and its lower-right corner at (c, d), where $1 \leq a \leq c \leq h$ and $1 \leq b \leq d \leq w$. To determine his favorite grove, Adam will assign a "quality number" to each type of tree. Then, the quality of a grove is the sum of the quality numbers of all trees contained in that grove. Adam wishes to determine which grove has the highest quality. If no grove has a positive quality, he would like to know that too.

Adam has decided on the following quality numbers:

birch	-5
chestnut	6
elm	3
fir	-2
maple	1
oak	2
empty	0

Input / Output

See the introduction for the input format. No further assumptions are made on the input.

The output for each data set is as follows: The first line contains the sentence "Data set n:". Then, if a favorite grove has a positive quality, print the message "grove $(a,\ b)$ $(c,\ d)$ has quality q" where a, b, c and d are the coordinates of the grove, and q is its quality. However, if no grove has a positive score, print "no grove has a positive score". A blank line should follow each data set.

Favorite groves need not be unique. In this case, print the coordinates of any of the favorite groves.

Sample

```
Data set 1: grove (1, 1) (2, 3) has quality 13

Data set 2: grove (1, 1) (2, 6) has quality 8
```

Cornell Programming Competition

Fall 1999
Problem C – Take a Rest

Description

When wandering through his forest, Adam occasionally wants to sit down and take a rest, so he decides to place a bench on one of the empty spots. He really likes his trees, and wants to see as many of them as possible, but realizes that from some places, his view of one tree is obscured by another. More precisely, if tree Z stands behind tree Y in a straight line from Adam's position, he can't see tree Z.

(For the purposes of this problem, assume that trees have an extremely small but nonzero diameter. This means that tree Y blocks tree Z if and only if tree Z is directly behind tree Y.)

He wants to place his bench on one of the empty grid positions such that as few trees as possible are obscured by other trees. He does not, however, discriminate between different kinds of trees. You are to write a program to determine the best possible bench positions.

Input / Output

The input format is described in the introduction. In addition, we will assume **for this problem only** that there is always at least one empty space in the garden.

The output for each data set is as follows: The first line contains the sentence "Data set n: He can see k trees in the following places", where k is the maximum number of trees visible from any point. This is followed by the coordinates of all of the best bench positions, sorted first by non-decreasing row and then by non-decreasing column. A blank line should follow each data set.

Sample

```
Data set 1: He can see 7 trees in the following places
2 2
3 2
4 2

Data set 2: He can see 10 trees in the following places
1 4
2 3
2 5
4 3
4 5
5 4
```

Cornell Programming Competition

Fall 1999
Problem D – Stop the Invasion

Description

After a short time, Adam discovers that the beavers from the nearby lake enjoy his forest at least as much as he does, but in a less peaceful way. He therefore decides to surround it with a fence. Because fences are expensive, he wants to buy as little of it as possible (grad students don't get paid very much). He even decides not to fence the entire property, but rather as small as possible an area that contains all of his trees.

For the purposes of this problem, assume that all trees have zero diameter.

You will determine how much fence is needed.

Input / Output

See the introduction for the input format. No further assumptions are made on the input.

The output file contains a single line with the sentence "He will need x yards of fence." Here, x is the minimum length of fence needed to fence all of the trees, and it should be rounded to two decimals.

Sample

```
Data set 1: He will need 92.36 yards of fence.
Data set 2: He will need 182.61 yards of fence.
```

Cornell Programming Competition

Fall 1999
Problem E – Clearings

Description

Adam has noticed during his walks in the garden that from some grid points it is possible to reach other grid points by only walking along grid lines. The set of all grid points that he can reach by walking this way defines clearings in the forest (at least if we assume that a clearing can be bounded on one or more sides by the end of his property).

Now, he gets curious and wants to know how many clearings there are in his garden. Write a program that will find it out for him.

Input / Output

See the introduction for the input format. No further assumptions are made on the input.

For each data set, output the line "Data set n: There are k clearings." where n is the data set number, and k is the number of clearnings in that data set.

Sample

```
Data set 1: There are 1 clearings.
Data set 2: There are 3 clearings.
```

Don't be worried about the poor grammar in data set 1's output.

Cornell Programming Competition

Fall 1999
Problem F – Sunny Spots

Description

While trees are really beautiful, Adam decides that it is about time he planted some flowers as well. So, he looks for a sunny spot to use for a flowerbed. How sunny a spot is depends of course mostly on the shade it gets from trees, in particular from the tree nearest to it. He therefore looks for the spot that is as far away from its nearest tree as possible.

Write a program to determine all candidates for him.

Input / Output

The input format is the one described in the introduction. In addition, we will assume **for this problem only** that there is always at least one tree and at least one empty spot in the garden.

For each data set, output the line "Data set n: The following are all d yards from the closest tree", where n is the data set number and d is rounded to two decimal places. Then, output the coordinates of the best flowerbed positions. They should be sorted first by non-decreasing row and then by non-decreasing column. There should be a blank line between data sets.

Sample

```
Data set 1: The following are all 10.00 yards from the closest tree
2 2
3 2
3 3
4 1
4 2

Data set 2: The following are all 20.00 yards from the closest tree
1 4
5 4
```