Ninth Annual
University of Central Florida

ACM • UPE

High School Programming Tournament

Problems

Problem Name	Filename
Gone Phishin'	PHISH
Obedience School	OBEY
So, Who Won?	OTHELLO
Where's Ali?	WHERE
Dueling Goats	GOATS
I Can't Hear You	HEAR
The Annoying Salesman	SALESMAN
Fire! Fire! Fire!	FIRE
M&M Factory	FACTORY
Vhat's fur Dinner?	CHEF

Call your program file: *Filename*.PAS or *Filename*.C
Call your input file: *Filename*.IN

For example, if you are solving So, Who Won?:

Call your program file: OTHELLO.PAS or OTHELLO.C
Call your input file: OTHELLO.IN

Gone Phishin'

Filename: PHISH

Many Grateful Dead fans (Deadheads) have begun to follow a band called Phish. Both the Deadheads and the Phishheads are strange groups and have developed a language of their own. Phishheads, in particular, use PH instead of F in honor of the band's name. For example, the phrase "Vermont's Finest" from a popular brand of ice cream has been seen on T-shirts as "Vermont's Phinest". Many Deadheads have trouble using the Phishheads' lingo. They need you to write a program to help them fit in with the Phishheads.

The Problem:
Convert normal English into the Phishhead equivalent.

The Input:
You will be given several paragraphs of text. End of data is indicated by end of file. No line will exceed 80 characters.

The Output:
Output the text exactly as it appears in the input file except with every F changed to a PH. To preserve capitalization, replace each lower case f with ph and each upper case F with Ph.

Sample Input:
```
A Falafel Vendor: Fresh Fat Falafels!
Two for four-fifty! Can't find a better deal!
A Police Officer: Hey! You can't sell that! No vending, freak!
```

Sample Output:
```
A Phalaphel Vendor: Phresh Phat Phalaphels!
Two phor phour-phiphty! Can't phind a better deal!
A Police Ophphicer: Hey! You can't sell that! No vending, phreak!
```

Obedience School

Filename: OBEY

Guard dogs at a training school learn to respond to and protect an instructor. Like their human counterparts, all dogs do not respond equally well to training. Certain dogs fail to obey commands they are currently learning; this can cause the instructor serious headaches.

The Problem:

Write a program that will simulate the actions of dogs during training. The dogs are being taught three commands:

speak	Dogs that respond to speak will be heard only if the dog is near the instructor.
come	Dogs that respond to come will return to the instructor.
A sic B	If dog A responds to sic, dog A will chase dog B. As a result, both dogs will end up away from the instructor, regardless of where they are before the command is issued.

There are four dogs: Argus, Cerberus, Gromit, and Ren. Argus will respond to all the commands, Cerberus will not come, Gromit will not speak, and Ren will not sic. All dogs start off near the instructor.

The Input:

The input will consist of several lines representing commands to the dogs. Each line contains one command that starts with the dog's name followed by one of the three commands above. The command sic will be followed by a second dog's name, representing the dog that will be attacked. Commands will only be given to Argus, Cerberus, Gromit, and Ren and only these dogs may be attacked with the sic command. End of data is indicated by end of file.

The Output:

The output will consist of the dogs' responses to speak. When a dog is instructed to speak, he'll respond with the following output (if he is both near the instructor and understands the speak command):

Bark!

otherwise print:

No response.

Sample Input:
```
Cerberus sic Argus
Ren speak
Gromit speak
Cerberus speak
Cerberus come
Argus come
Argus speak
```

Sample Output:
```
Bark!
No response.
No response.
Bark!
```

So, Who Won?

Filename: OTHELLO

The judges of an Othello contest would like your help deciding game winners.

The Problem:
Given a final 8×8 Othello game board report who wins or notify the judges of a tie. The winner is the player with more pieces on the board.

The Input:
There will be exactly one game board in the input file. It will be specified as eight rows containing exactly eight characters chosen from B's, W's and .'s. B represents a black piece, W represents a white piece, and . represents an empty square.

The Output:
For the given board output:

x wins.

where *x* is either Black or White. If there is a tie then output:

Tie game.

Sample Input:
```
........
B.W.WB.W
........
........
........
......W..
........
BWBB...W
```

**Sample Output:**
```
White wins.
```

Where's Ali?

Filename: WHERE

I'm sure you've heard of the popular *Where's Waldo?* games where you must search through a picture containing lots of people and other distractions to try to find one tiny person named Waldo. A computerized version of this game is called "Where's Ali?". In this game you search through a piece of ASCII art to find Ali.

The Problem:
Write a program to find Ali. Ali looks like this:
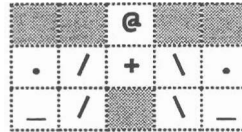
Note that all non-shaded squares represent parts of Ali and the shaded ones may contain any other characters as specified below.

The Input:
There will be multiple data sets. End of data is indicated by end of file.

Each data set will contain an ASCII picture with 10 rows and 40 columns for you to search. The picture may be composed of any standard ASCII characters: that is, any letters, numbers, spaces, and any of:

~`!@#$%^&*()_-=+[]{}:;'"<,.>/?|\

Each data set will contain exactly one Ali.

The Output:
For each data set, your program must output the following message:

```
Ali is hiding at x,y!
```

Where *x* and *y* are the column and row respectively of Ali's head (the @ character).

Sample Input:
```
|                           |        |       |        |
|+------------+ 8 8 88 888  |        |  Food> |        |
||High School| 8 8 8   8o   |        |        |        |
|| Contest    | 888 8   8   |        |    ___@____    |
|+------------+              |        / @,/*\,         |
|        !~~~!\        !~~~!\        /./+\./\          |
|   @  @ !C:_!/  @ !___!/ @|/  _/ \_            |
|  /X\ (=/###/   |=/###/ /+\         ,zz        |
|_/ \_||        / \     _/ \_ _/\=@__       |
|#######################################|
```

Sample Output:
```
Ali is hiding at 33,5!
```

Dueling Goats
Filename: GOATS

Billy-Bob and Bubba are amusing themselves by watching two goats that are tied to posts. They want to know whether these goats will kick each other. Not being the brightest of all characters, they look away whenever any exciting goat action takes place. They need you to tell them whether the goats kick each other.

Each goat is tied to a fixed-length rod which in turn is fastened to a post. Therefore the goats stay a constant distance from their posts. The goats will kick each other if they can pass within 0.0001 units of each other, as shown in the following diagram:
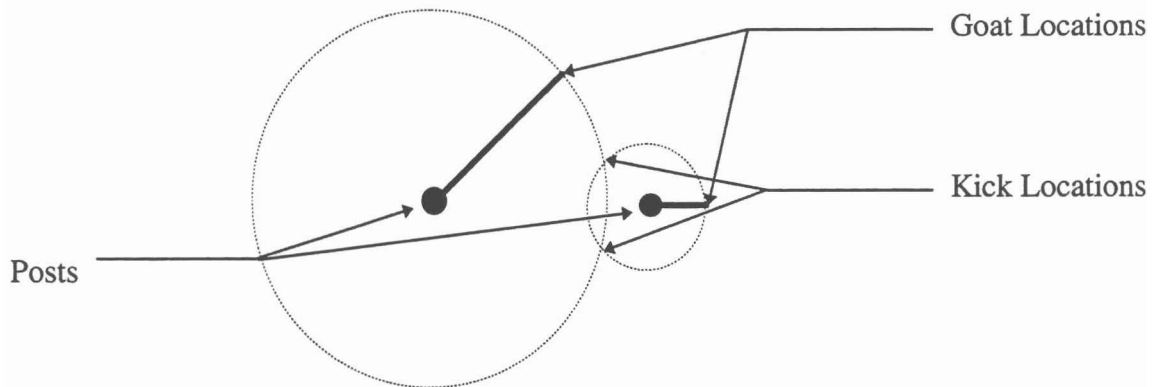
The Problem:
Given the locations of the posts and the length of the rod tied to each goat, determine all locations at which the goats could kick each other.

The Input:
The first line of the input file contains a single integer n; there are n data sets in the input file. Each data set consists of two lines, each specifying information for one of the goats. Each line contains three real numbers x, y, and l, separated by spaces; (x,y) is the location of the post and l is the length of the rod connected to the goat. End of data is indicated by end of file.

The Output:
For each data set, print the number of the data set followed by the points at which the goats kick each other. Output coordinates should be correct to three decimal places as shown below. If the goats cannot kick each other, print the message:

```
The goats can't kick each other.
```

Separate output sets with a single blank line. Follow the Sample Output for formatting details.

Sample Input:
```
3
0.0 0.0 1.0
2.0 0.0 1.0
-1.0 0.0 1.4142136
1.0 0.0 1.4142136
0.0 0.0 3.0
8.0 3.0 1.0
```

Sample Output:
```
Set #1
  The goats kick each other at 1.000 0.000

Set #2
  The goats kick each other at 0.000 1.000
  The goats kick each other at 0.000 -1.000

Set #3
  The goats can't kick each other.
```

Disclaimer: No goats were injured in the making of this problem.

I Can't Hear You

Filename: HEAR

Two telephone companies provide service to the remote island of Benet: First Phone Company (FPC) and Best Phone Company (BPC). Each telephone company owns a communication network; these networks are not connected to each other. To improve island unity, the local governor has asked FPC and BPC to connect their networks. Wanting to keep costs down, the phone companies agree that a single connection between two customers, one from each network, will suffice. They now seek the shortest distance between two such customers to further control costs. The following picture illustrates this idea for six of Benet's houses:
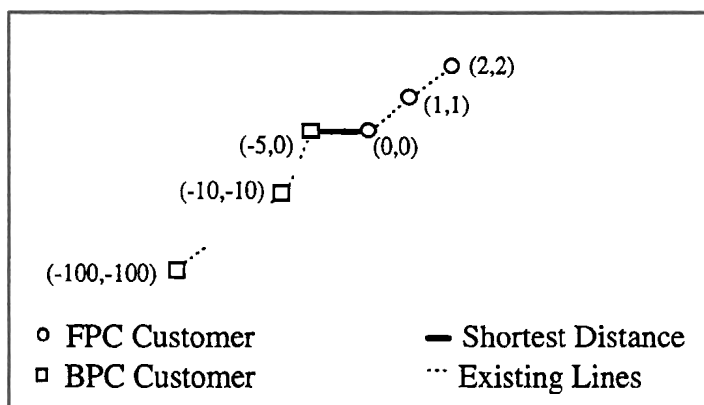
The Problem:
Given the locations of customers of FPC and of BPC, find the minimum distance between a customer of FPC and a customer of BPC.

The Input:
Each data set will begin with a line containing two integers $m \geq 1$ and $n \geq 1$; there are m customers of FPC and n customers of BPC. The next m lines will specify the locations of customers of FPC as pairs of floating point values separated by a single space. The next n lines will specify the locations of customers of BPC as pairs of floating point values separated by a single space. End of input will be signaled by a data set in which neither FPC nor BPC has any customers (i.e., $m = n = 0$). The total number of customers on Benet will not exceed 100.

The Output:
For each data set, output the following message:

```
The distance between the closest pair of customers is distance.
```

where *distance* is the appropriate value, expressed to two decimal places.

Sample Input:
```
3 3
0.0 0.0
1.0 1.0
2.0 2.0
-5.0 0.0
-10.0 -10.0
-100.0 -100.0
2 2
0.0 0.0
1.0 1.0
0.0 -1.0
-10.0 -20.0
0 0
```

Sample Output:
```
The distance between the closest pair of customers is 5.00.
The distance between the closest pair of customers is 1.00.
```

The Annoying Salesman

Filename: SALESMAN

Police have been trying to catch a traveling salesman who has very unusual selling habits. The salesman visits houses with "No Soliciting" signs. His strange pattern of visitation is what has stumped the police, who would like to catch him. Tired of this intrusion, one neighborhood in which no house wants solicitors has decided to protect themselves by having a police officer live in one of the houses.

Suppose the first house the salesman has not yet visited is house k. He then visits every k^{th} unvisited house starting with house k. For example, if the first unvisited house is house 2, the salesman visits houses 2, 4, 6, 8, etc. After he finishes talking to those people, he starts back at the beginning of the street again, this time starting at house 3 and visiting every third unvisited house, etc. The first house number is 2 and all the houses are numbered sequentially.

If the salesman tries to sell anything to the police officer, he will be caught.

The Problem:
Write a program to determine how many houses the salesman visits before he is caught.

The Input:
Each data set consists of a line containing two integers. The first integer, n ($1 < n < 5000$), is the last house number on the street. The second integer, p ($1 < p \le n$), is the house number that the police officer lives in. End of data is indicated by end of file.

The Output:
For each data set, output the number of people the traveling salesman visits before reaching the police officer's house. The format for each output line is

The salesman visits x people before being caught.

Sample Input:
```
5 4
10 7
```

Sample Output:
```
The salesman visits 1 people before being caught.
The salesman visits 8 people before being caught.
```

Fire! Fire! Fire!

Filename: FIRE

When watching TV with Butt-head, Beavis becomes excited whenever he hears a word that sounds like "fire". However, Beavis quickly becomes bored with less inflammatory dialogue.

The Problem:
Given an inspiring TV quote, supply Beavis' portion of the dialogue.

The Input:
There will be several lines of input. Each line will consist of several words that make up a TV quote. Words are considered to be any string of letters, numbers, dashes, or apostrophes delimited by other characters. Words will be less than 50 characters in length. Anything that is not a word should be disregarded. End of data is indicated by end of file.

The Output:
For each word that sounds like "fire" in an input line, your program should print that word three times on a separate line exactly as it appears in the input (see below). For the purposes of this problem a word sounds like "fire" if its last three characters are `ire`.

If there is no word that sounds like "fire" on a given input line, print:

```
Change the channel, Butt-head.
```

Sample Input:
```
Hire someone who will not tire you.
An ice-cold soft drink would be desired.
Where's the remote?
That has too much satire, irony, and symbolism.
```

Sample Output:
```
Hire! Hire! Hire!
tire! tire! tire!
Change the channel, Butt-head.
Change the channel, Butt-head.
satire! satire! satire!
```

M&M Factory

Filename: FACTORY

The ACM contest problem factory uses a unique system of gears. The contest judges have found a way to use M&M candies as gears in their factory by adding little teeth to the edges. The neat thing about these candies is that they have the letter M printed on each one. If you rotate an M&M gear clockwise 90°, the M looks like a 3. If you rotate it again, you get a W, a third rotation gives you an E, and a fourth rotation gives you an M again.

The M&M gears are arranged in a long row at the factory. Turning a gear causes the two gears on either side to turn in the opposite direction, the gears on the other side of those two will turn in the same direction as the originally turned gear, and this continues all the way out to the gears on each end. The problem is that the judges like to eat the M&Ms. When a judge eats one of the M&Ms the factory ceases to function properly because the hole in the line of gears keeps some groups of gears from touching.
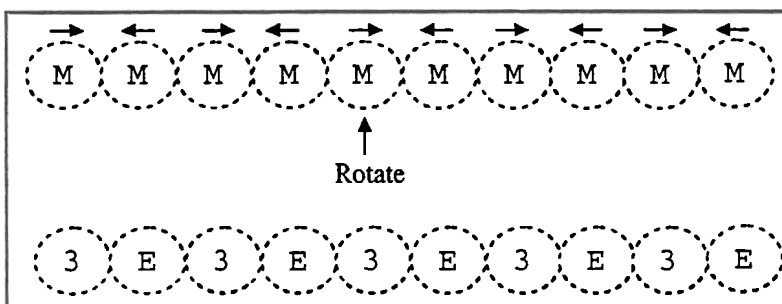
The Problem:
You will be given the number of M&M gears in a row at the start, and a sequence of events that occur. Each event will either be that a single M&M gear was rotated 90° clockwise, or that a single M&M gear was eaten by one of the judges. Your task is to write a program that will read in this information and print out what letter each M&M gear in the row looks like after the last event. Note that at the beginning all the letters will look like Ms.

The Input:
The first line of the input file will contain a single integer indicating the number of data sets to be processed. Each data set begins with a line containing a single integer m ($1 \le m \le 75$) that represents the number of M&M gears in the line at the start. The next line contains a single integer n ($1 \le n \le 5000$) that represents the number of events to be processed. This is followed by n lines listed in the order that the events occur. Each line describes one event. Each event line begins with a single letter T or E followed by a space and then an integer i ($1 \le i \le n$). A T means that the M&M gear in the ith position was turned clockwise 90°. An E means that the M&M gear in the ith position was eaten. In the event descriptions, the gear positions are considered to be numbered from 1 to n. The position number for a given gear **does not change** as gears are eaten.

The Output:
For each input set, print out a single line showing what the letter on each M&M gear looks like after all events have taken place. Print a space for gears that have been eaten.

Sample Input:
```
3
10
1
T 5
20
4
E 5
T 1
T 6
T 7
10
2
T 1
T 9
```

Sample Output:
```
3E3E3E3E3E
3E3E MMMMMMMMMMMMM
WWWWWWWWW
```

Vhat's fur Dinner?

Filename: CHEF

Because of the Swedish Chef's strong accent, Kermit and the other Muppets have a very difficult time understanding him. Kermit would like to have the other Muppets practice listening to Swedish-accented English. He needs your help.

The Problem:
Write a translator that converts English to Swedish-accented English. Do this by replacing substrings in sentences with other substrings. In the table below, the left column shows the original English substring and the right column shows what it should be converted into (ignore all quotation marks):

FROM	TO
"and"	"und"
"And"	"Und"
"Then"	"Den"
"then"	"den"
"the"	"der"
"The"	"Der"
"Yes"	"Ya"
"yes"	"ya"
"Which"	"Vich"
"which"	"vich"
"x."	"xy."
"x "	"xy "
"w "	"v "
"w"	"v"
"W"	"V"
"d."	"dy."
"d "	"dy "
"p."	"py."
"p "	"py "
"v"	"f"
"e "	"ey "
"e."	"ey."
"ir"	"ur"
"or"	"ur"
"ou"	"oo"
"ue"	"oo"
"Th"	"D"
"th"	"d"
"ick"	"ook"

Note that when applying substitutions, you should scan the input line from left to right trying the rules in the table from top to bottom at each character of the input line. Anything that does not match one of these rules should be left alone in the output. Furthermore, make sure that you neither apply a rule to text that is the output from the application of another rule nor overlap matching substrings (for example, "ue " should generate "oo " and not "ooey ").

The Input:
The input file will start with an integer *n* on a line by itself. On the following *n* lines there will be English text for you to translate.

The Output:
Each line of the output will be the Swedish-accented English translation of each line of the input. Although the output may take up to 150 characters per line, do not wrap the line after 80 columns.

Sample Input:
```
3
Welcome to the Ninth Annual High School Programming Tournament.
The judges think you will find the problem set both fun and
challenging.  We hope you actually do.
```

Sample Output:
```
Velcomey to der Nind Annual High School Programming Toornament.
Der judges dink yoo vill findy der problem set bod fun und
challenging.  Vey hopey yoo actually do.
```