Fifteenth Annual
University of Oregon
Eugene Luks Programming Competition

Saturday, April 9, 2011

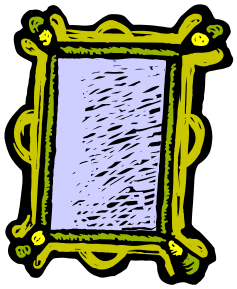
Problem Contributors

Jim Allen
David Atkins
Gene Luks

Prizes and food provided by Pipeworks

a Foundation 9 Entertainment Studio

REFLECTIONS



In plain fonts like Arial, the lowercase b and d letters are mirror images of each other, as are lower case p and q. The lowercase letters i, l, o, v, w, and x are mirror images of themselves, as are the uppercase letters A, H, I, M, O, T, U, V, W, X, and Y, and the digits 8 and 0. This means that a word like boxwood would appear as boowxod when viewed in a mirror. The word HOOT would appear as TOOH. Some strings might even appear the same in the mirror, e.g., the license plate p808q.

.

For this problem, you are given sequences of letters and numbers, and for each you must determine the reflection of the sequence. The first line of the input will be the number of lines that follows. Each subsequent line will contain a string of letters and numbers. For each string, output "Identical reflection" if the string appears the same in a mirror. If the string contains any letters or digits which do not reflect to a valid letter or digit, then output "Does not reflect well". Otherwise, output the reflection of the string.

Sample Input

```
4
boxwood
HOOT
p808q
Hello
```

Sample Output

```
boowxod
TOOH
Identical reflection
Does not reflect well
```

WAKE UP AND SMELL THE BACON

A popular game among thirty-something's is six degrees of Kevin Bacon. The trick is to take any movie star and find a series of connections with other stars that lead to Kevin Bacon in exactly six steps. For example, to find Julie Andrews' six degrees of Kevin Bacon, we observe that Julie Andrews was in "Thoroughly Modern Millie" with Pat Morita, who was in "The Karate Kid" with Ralph Macchio, who was in "My Cousin Vinnie" with Marrisa Tomei, who was in "Loverboy" with Travis Bacon, who was also in "Loverboy" with Kevin Bacon. According to the Oracle of Bacon website, approximately 12% of all actors cannot be linked to Kevin Bacon at all. IMDB is contacting you to ask if you will figure out which stars are connected in six degrees to Kevin Bacon.

As input, you will have the entire IMDB database and a set of queries. The first line contains two space-separated numbers n and m where $0<n<10{,}000$ and where $0<m<1000$. This is followed by n lines of the IMDB database and m query lines. The database lines begin with the movie name in quotes, followed by a space, followed by a comma-separated list of actors in the movie. Movie names can contain any character except double quotes. Names may contain any number of space-delimited parts, but only contain alphabetic and space characters. Each query line contains exactly one valid name.

You should output one line per query. If the person on the query line cannot be linked in exactly six degrees to Kevin Bacon, your program should output "`no`". If the person can be linked to Kevin Bacon in exactly six degrees, your program should output "`yes`". No actor can be in the same six degree arc twice, although the same movie might be used to link several actors.

Sample Input

```
12 5
"Thoroughly Modern Millie" Julie Andrews,Pat Morita,Mary Tyler Moore
"The Karate Kid" Pat Morita,Ralph Macchio,Elisabeth Shue
"My Cousin Vinnie" Ralph Macchio,Marrisa Tomei,Joe Pesci,Fred Gwynne
"The Musters Revenge" Al Lewis,Yvonne DeCarlo,Fred Gwynne,K C Martel
"The 10 Commandments" Charlton Heston,Edward G Robinson,Yvonne DeCarlo
"Key Largo" Humphrey Bogart,Edward G Robinson,Lauren Becall,Lionel Barrymore
"Charlie's Angels" Cameron Diaz,Drew Barrymore,Lucy Lui,Bill Murray
"Groundhog Day" Bill Murray,Chris Elliot,Andie MacDowell,Brian Doyle-Murray
"Tootsie" Dustin Hoffman,Jessica Lange,Teri Garr,Bill Murray,Dabney Coleman
"Loverboy" Kyra Sedgwick,Kevin Bacon,Marrisa Tomei,Travis Bacon
"It's a Wonderful Life" Jimmy Stewart,Lionel Barrymore,Donna Reed
"Vertigo" Jimmy Stewart,Kim Novak,Barabara Bel Geddes
Yvonne DeCarlo
Julie Andrews
Kyra Sedgwick
Teri Garr
Barabara Bel Geddes
```

Sample Output

```
yes
yes
no
no
no
```

HATCHING THE NEST EGG

As many of the Computer Science faculty approach retirement, they are thinking about how to make their savings last. For example, savings of $10,000 could be withdrawn in the amount of $2,000 each year for five years. However, things are not that simple – the savings is invested and will grow with the return on investment, so the withdrawals could be larger. On the other hand, inflation means that each subsequent year's withdrawal of the same dollar amount will actually be worth less. The faculty wants you to write a program that will calculate the right amount to withdraw to use up a nest egg without running out of money.

Starting with $10,000 assuming 10% return and 3% inflation.

Year	Withdrawal	Balance
1	$2271.25	$7728.75
2	$2339.39	$6162.24
3	$2409.57	$4368.89
4	$2481.86	$2323.92
5	$2556.31	$0.00

Suppose you started with $10,000 and assumed a 10% investment return and 3% inflation rate. If you withdrew $2,271.25 at the start, then after a year the remaining $7,728.75 would grow by 10% to $8,501.63. Increasing the second withdrawal by the 3% inflation rate to $2,339.39 would then leave $6162.24 at the beginning of the second year. The table shows the withdrawals at the beginning of each of the five years, each increasing by 3%, and the amount remaining after the withdrawal, and factors in the 10% growth.

For this problem, you must write a program that determines the amount of the first withdrawal such that increasing this amount each year by the inflation rate, and taking into account the investment return rate, will exactly use up the original savings over the given number of years.

Input to the program consists of a number of problems to solve. The first line indicates how many problems follow. Each problem consists of one line. The line begins with a decimal number that is the starting amount of the nest egg, followed by an integer that is the number of years. The line ends with two decimals that represent the rate of return and the rate of inflation, as percentages. For each problem, your program must print out, on one line, the initial withdrawal as a dollar amount.

Sample Input

```
3
1000.00 10 0.0 0.0
10000.00 5 10.0 3.0
10000.00 5 2.0 6.0
```

Sample Output

```
$100.00
$2271.25
$1849.17
```

INTELLECTUAL PROPERTY DISPUTE

MacroHard, a leader in Digital Rights Management (DRM), wants to create a program that will sort through a user's midi files and make sure that any music based on something MacroHard owns is paid for. They have hired a team of ninja programmers to handle downloading their software to unsuspecting users, finding midi files and checking whether the person paid royalties to MacroHard. Similarly, they have corporate artists who are able to pick out the melody from a sequence of chords. What they want you to do is take two melodies and see if one is based on the other.

1	C B#
2	C# D-
3	D
4	D# E-
5	E F-
6	F E#
7	F# G-
8	G
9	G# A-
10	A
11	A# B-
12	B C-

Melody *A* can be legally shown to be based on Melody *B* if *A* can be obtained from B by some combination of:

1. Transposing *B* into a new key
2. Removing the introductory notes from *B*
3. Adding bridges to the melody of *B*

Putting in a bridge means to insert additional notes between two adjacent notes of the melody, but adding notes to the beginning or end of a melody is not a bridge.

The key of a melody is the last note. The process of transposing to a new key means to add or subtract the same number of half-steps from each note. Although the notes are labeled A through G with sharps and flats, the number of half-steps between notes is tricky. The chart shows the twelve notes, each a half step apart, in the chromatic scale used in Western music along with the possible names of each note. Many notes have two names. (We use '-' for a flat instead of ♭.)

The scale wraps around, so the 13th note is just the 1st note in the next octave. Octaves and timing don't matter to this process, so all of the melodies are given as lists of notes with no octave or timing information.

The input begins with a single number $0 < n < 1000$, followed by $2n$ lines of melodies. Each melody has between 3 and 1000 notes. For each pair of melodies, you are to decide whether the first melody, A, can be derived from the second melody, B, using steps 1-3 above. If the first melody can be derived from the second melody, you should output "`sue!`" If not, you should output, "`continue scan`".

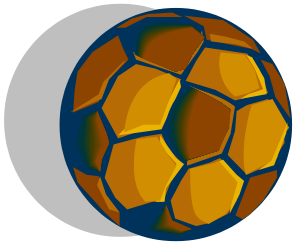
Sample Input

```
7
CDEFGABC
CEGC
CDEFGABC
ABCEGC
B-CDE-FGAB-
CEGC
DEF#GABC#D
CEGC
CBAGFECD
CEGC
CDEFGABCCBAGFECD
CEGC
CEFGCEFGCEFGECEDEDC
DABF#GDGAF#EDC#BABCDDA
BF#GDGAF#EDC#BABCD
```

Sample Output

```
sue!
sue!
sue!
sue!
continue scan
continue scan
continue scan
```

SCATTER BALL



The summer-camp game SCATTER BALL is played on an enormous field. In each round, one camper is designated as "IT" and has possession of a ball. To start, IT yells *"SCATTER*" and counts to 10 while all the other campers disperse in random directions. When finished counting, IT yells *"FREEZE"*. and the others must stop where they are. IT will have to throw the ball and hit a camper to be designated by the group as *"TARGET"*. But *before* TARGET is selected and revealed, IT is given the opportunity to move to a position from which to throw.

Now, the other campers will always designate TARGET to be the farthest camper from IT's new position. Hence, IT's goal is to select the position that minimizes the maximum distance to the other campers.

In this problem, you will be given a list of positions of the scattered campers and must determine IT's optimum position.

The first line of input to this problem will contain an integer *n* indicating the number of SCATTER BALL rounds to follow. Each round will start with a line containing an integer *m*, where $m \leq 25$, that indicates how many campers have scattered on the field. Each of the succeeding *m* lines will contain a pair of integers (x,y) giving the coordinates of a scattered camper.

For each round, the output should be the coordinates of IT's optimum position, correct to 2 decimal places. The output for each round should start on a new line.

Sample Input

```
4
3
-3 -3
1 1
2 2
5
0 4
0 0
4 0
1 2
1 4
4
-3 5
5 5
1 9
1 14
3
-3 6
4 -5
4 9
```

Sample Output

```
-0.50 -0.50
2.00 2.00
1.00 8.61
2.86 2.00
```