PSCSTA Programming Competition

April 25th, 2015

INTERMEDIATE & ADVANCED DIVISIONS

- Complete the problems in any order.
- All problems have a value of 60 points.
- Your program should not print extraneous output. Follow the format exactly as given in the problem.
- There is a 5-point penalty for each incorrect submission for problems that are eventually judged correct.
- Time will be used to break point ties.

Page #	Problems	Points	Notes
1	Determined1		
2	Bank Street Writer		
3	Splatter		
4	Drought		
5	Cross		
6	Buoyancy		
7	Accounting		
8	Cursed		
9	Data Count		
10	HistoNum		
11	Say Hello		
12	Shuffle		
13	Sine Up		
14 - 15	Determined2		

Good luck!

Determined1

Input file: determined1.dat

You are determined to find the determinant of a 2x2 matrix. A matrix is a set of numbers in rows and columns. A 2x2 matrix determinant can be found by:

$$det\begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$$

Given 2 rows and 2 columns of numbers, find the determinant.

Input: The first line (N) indicates the number of data sets. Each data set contains 2 rows and 2 columns of integers.

Output: Print out each determinant on one line.

Example Input File:

```
3
1 2
3 3
10 10
10 10
-2 3
-4 -5
```

Output to screen:

```
-3
0
22
```

Bank Street Writer

Input File: bank.dat

Bank Street Writer was a word processing tool designed in 1981 by a team of educators at the Bank Street College of Education in New York City. Because of its wide adoption in elementary schools in the following years, it was generally thought at the time to have changed the way students learned to write by allowing them to concentrate on the creative parts of writing, and making the process of editing easier than with pencil, paper and eraser.

Your job is to simulate this older style word-processor by implementing a few commands: SEARCH, DELETE, REPLACE, and INSERT. You'll be given an input string, followed by a list of commands.

SEARCH N C	Search for the first occurrence of the character C appearing at or after position N and print the position where C is found. If C is not found, output -1.
DELETE N M	Starting at position N, delete M characters from the string and print the new string.
REPLACE N C	Replace the Nth character with the character C, and print the new string.
INSERT N S	Insert the new string S followed by a space to the left of the Nth position, and print the new string.

Input:
An initial value N, followed by N sets of data. Each data set consists of a string to be edited, an integer value M, followed by M editing commands. The string to be edited will contain only upper-case letters and single space blanks (which count as characters), and will begin and end with a letter. Indexing is 1-based (not zero-based).

Output:
The result of applying each command to the given string.

Sample Data File:

```
1
I LIKE DOING THESE PROBLEMS
5
SEARCH 8 E
REPLACE 16 O
DELETE 8 6
INSERT 20 CSTA
INSERT 1 WHY DO
```

Output to Screen:

```
16
I LIKE DOING THOSE PROBLEMS
I LIKE THESE PROBLEMS
I LIKE DOING THESE CSTA PROBLEMS
WHY DO I LIKE DOING THESE CSTA PROBLEMS
```

Splatter

Input File: splatter.dat

When I last painted the rooms in my house, my youngest daughter was 3 years old. I thought it would be funny if she helped by "throwing" paint on the wall, like Jackson Pollock. Let's write a program simulating this! You will be given a wall size (matrix) and some paint splatters. When the paint hits the wall, it spreads out (2 up, down, left, right and 1 diagonally).

Here are examples of 2 splatters at (3,4) and (0,8)

top-left = (0,0)

						X	X	X	X
				X			X	X	X
			X	X	X			X	
		X	X	X	X	X			
			X	X	X				
				X					

bottom-right = (5,9)

A wall is considered covered if there is a splatter on every square foot.

Input: The first line (N) consists of the number of data sets. Each data set starts with the height and width of the wall (R and C). The next line contains the number of paint splatters, T, followed by T coordinates of the splatters (X and Y).

Constraints:
1<=R<=20
1<=C<=20
1<=T<=R*C

Output: Print out "YES" or "NO" to indicate if the wall is covered or not.

Example Input File

```
2
6 10
2
3 4
0 8
4 4
4
1 1
3 3
3 1
1 3
```

Output to screen:

```
NO
YES
```

Drought

Input File: drought.dat

Much of Texas has been under drought conditions for the last 4-5 years. Write a program to keep track of monthly rain totals. Given an average yearly rainfall amount and 24 months of data, indicate if the drought is continuing or if the weather pattern is improving.

For this program:

- The drought is over if both years' data show at least twice the average.
- The weather pattern is improving only if both years' rainfall is at least average.
- The drought is continuing otherwise.

Input: The first line (N) consists of the number of data sets. Each data set contains 25 numbers on one line. The first is the average yearly rainfall, followed by 24 months of rainfall data.

Output: For each data set, print out "drought over", "improving", or "continuing" on a single line.

Example Input file

```
3
21 1.2 2.0 0.0 1.0 1.0 1.5 0.5 0.6 0.8 0.1 0.0 0.0 1.0 2.0 1.0 2.0 2.5 3.5 3.0 3.0 2.0 2.5 3.0 3.0
21 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0
22.5 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8 3.8
```

Output to screen:

```
continuing
improving
drought over
```

Cross

Input File: cross.dat

Given 2 words, you will determine if they can be perpendicular in a crossword puzzle. If they can, you will show them intersecting. The first word will be the "across" and the second word will be the "down" word. Here are some examples:

cross and green (the r's intersect)
playing and dog (the g's intersect)

```
 g                d
cross             o
 e          playing
 e
 n
```

Input: The first line (N) will contain the number of data sets. Each data set consists of two words separated by "and." The longest word will be 15 characters long.

Output: For each data set, print out either "none" or the intersecting words. The words should intersect at the first shared character. Separate data sets with a blank line.

Example Input file

```
4
cross and green
playing and dog
meet and non
school and love
```

Output to screen:

```
 g
cross
 e
 e
 n

      d
      o
playing

none

   l
school
   v
   e
```

Buoyancy

Input File: buoyancy.dat

Some schools have an “egg-drop” project. The task is to build a device to protect a raw egg when dropped from a certain height. When my daughter did this project, she decided to use helium balloons to hold up the eggs. She was worried that the balloons would float away!

Using the fact that the buoyant force is equal to the weight of the air displaced, we can calculate how many balloons it would take to lift an egg. The mass of an egg is an average of 55 g, so the weight is 0.54 newtons. Each liter of balloon volume provides a buoyant force of 0.011 newtons. Given the volume of each balloon in liters, determine how many balloons it would take to equal or exceed the weight of one egg.

Input: The first line contains the number of data sets (N). Each subsequent line contains an integer, Y, the balloon size in liters.

Output: Print out the number of balloons required.

Example Input file

```
3
1
5
6
```

Output to screen:

```
50
10
9
```

Accounting

Input File: accounting.dat

In accounting currency formatting, very large values are separated by commas, and negative values are expressed inside parentheses. Below are some examples:

$250.34
$(500.19)
$2,343,555.55
$(59,216.99)

As each transaction is accounted for, the balance is updated by adding the credits and subtracting the debits, so the running balance for the numbers above would be as shown below, with negative balances shown in parentheses, and all values $1,000 or more separated out using commas:

$250.34
$(249.85)
$2,343,305.70
$2,284,088.71

Your job is to write a program that shows the credits, debits, and running balance, formatted as shown, complete with the top and bottom "****." lines and column headings (exactly as shown).

Input: Several dollar amounts in accounting currency format as shown above, each on one line. It is guaranteed that all transaction values will be less than 1 billion dollars.

Output: A formatted running balance as shown below, complete with the header and footer lines.

Sample Input

$250.34
$(500.19)
$2,343,555.55
$(59,216.99)

Example Output to Screen

```
****.****.****.****.****.****.****.****.
       Transaction : Balance
           $250.34 : $250.34
         $(500.19) : $(249.85)
     $2,343,555.55 : $2,343,305.70
      $(59,216.99) : $2,284,088.71
****.****.****.****.****.****.****.****.
```

Cursed

Input File: cursed.dat

Nigel has invented a time machine, however he has noticed that he has been randomly having bad days. A variety of horrible things can happen to him on these days, which he calls his ***"cursed"*** days. After several of these days, Nigel noticed these days only happen when the date in MMDDYYYY form is a palindrome (read the same forward and backward). Because he is a young time traveler, he needs your help in recognizing which of these days he should not travel on.

Input: A date with 3 parts, the month, the day followed by a comma, and the year between 0 and 9999.

Output: For each date print it in MMDDYYYY format, and the message "DON'T TRAVEL" if the date is a palindrome or "OK TO TRAVEL" if it is not a palindrome.

Assumptions: The input will be a valid date in AD. Note that the MMDDYYYY form has 2 characters for month, 2 for day, and 4 for year, if the provided date does not contain enough characters for these fields, they should be padded with leading '0's. (For example the year 109 becomes 0109 to fill 4 characters in year).

Sample Input
June 1, 1998
January 9, 109
December 31, 1321

Example Output to Screen
06011998: OK TO TRAVEL
01090109: OK TO TRAVEL
12311321: DON'T TRAVEL

Data Count

Input File: All the data files!

This problem is fairly trivial, but will give you a chance to exercise your file input technique. Your job is simply write a program to count how many lines of data are in all of the data files of this packet. Have fun!

Input: All data files for this packet.

Output: The total number of lines of data combined in all the data files for this packet. *Note: Since the judges' data will be different than the student data, you must input each data file and make your program count it. It is not sufficient to just count the lines by hand and output the integer you count. You may assume that the judges files will be named the same as the student files.*

Sample Input (this is only two of the files)
//data from accounting.dat
$250.34
$(500.19)
$2,343,555.55
$(59,216.99)
//data from cursed.dat
June 1, 1998
January 9, 109
December 31, 1321

Example Output to Screen *(output for only these two data files)*
THERE ARE 7 LINES OF DATA FOR THIS PACKET

HistoNum

Input File: histonum.dat

Histograms are bar graphs, and this problem is an example of one. Given a series of integers, count the number of occurrences of each digit (0-9) and output a horizontal bar graph for each digit.

Input: Several input integers, each on one line.

Output: Output the corresponding histogram, as shown below. Each line should end with the number of occurrences for each digit surrounded by curly braces ({}). If no digit occurs in the number series, do not output that bar.

Sample Input

35987
176253
20293805
2387
3981

Example Output to Screen

0|** {2}
1|** {2}
2|**** {4}
3|***** {5}
5|*** {3}
6|* {1}
7|*** {3}
8|**** {4}
9|*** {3}

Say Hello

Input File: none

Say "Hello" to the Judges with each person's first name, and the total number of letters combined in all the first names. You can manually count your letters, or use some other, more elegant technique if you wish.

Input: None.

Output: A unique, interesting and school appropriate message saying "Hello" to the judge, including all first names of the team members, and a number that represents how many letters combined are in all the first names.

Example Output to Screen

Hello Esteemed Judges! We are Juan, Nancy, and Amanze! There are 15 letters in all of our names. Have fun grading our programs, and have a wonderful day!

Shuffle

Input File: shuffle.dat

Given a sentence, output all unique words in alphabetical order.

Input: Several sentences, each on one line, all words in uppercase letters, with no other characters of any kind included.

Output: All unique words from each sentence in alphabetical order.

Sample Input

PRINT ONLY THE UNIQUE WORDS IN ALPHABETICAL ORDER
JINGLE BELLS JINGLE BELLS JINGLE ALL THE WAY
MARES EAT OATS AND DOES EAT OATS AND LITTLE LAMBS EAT IVY
SHE SELLS SEASHELLS ON THE SEASHORE
HELLO WORLD

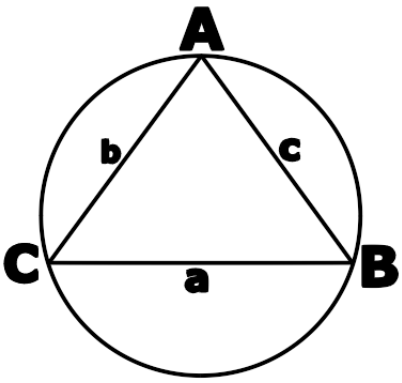
Example Output to Screen

ALPHABETICAL IN ONLY ORDER PRINT THE UNIQUE WORDS
ALL BELLS JINGLE THE WAY
AND DOES EAT IVY LAMBS LITTLE MARES OATS
ON SEASHELLS SEASHORE SELLS SHE THE
HELLO WORLD

Sine Up

Input File: sineup.dat

The Law of Sines for Trigonometry can be used in several different ways. It goes like this:



Given the diameter D of the circumcircle of a triangle ABC, with sides a, b, and c opposite their corresponding angles, the ratios ***a/(sin A), b/(sin B), and c/(sin C)*** *are all equal to the length of the diameter D.*

In this problem you are given the diameter D of the circumcircle of the triangle ABC, and the degree measures of the two angles A and B. You are to find the third angle, and the lengths of the three sides, and output all of the information as shown below.

Note: A circumcircle of a triangle is a circle that contains all three vertices of the triangle.

Input: Several lines of data, each line consisting of three integer values: the diameter of a circumcircle and two angle degree measures of the corresponding triangle.

Output: All of the information for the situation, formatted exactly as shown below. All integers should be round to the nearest whole number.

Hint: Keep in mind that most sin() evaluation functions require that the angle is in radians (not degrees). There are 360 degrees (or 2π radians) in a circle.

Sample Input

100 40 60
50 45 45
70 90 60

Example Output to Screen

Circumcircle diameter = 100
Angles are 40, 60 and 80
Corresponding sides are 64, 87 and 98
Circumcircle diameter = 50
Angles are 45, 45 and 90
Corresponding sides are 35, 35 and 50
Circumcircle diameter = 70
Angles are 90, 60 and 30
Corresponding sides are 70, 61 and 35

Determined2

Input File: determined2.dat

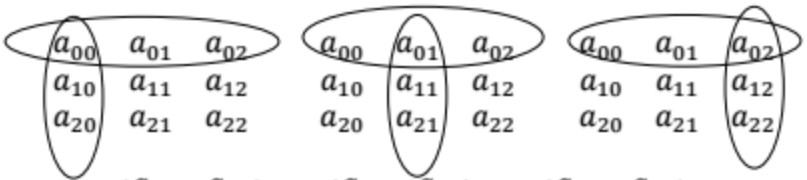
Even if you haven't seen this in math class, don't let that *deter* you from trying this program!

In this program, find the determinant of a square matrix TxT, where T=3,4,5…10. Each matrix element will be an integer -100<x<100.

A 2x2 matrix determinant can be found by:

$$\det(A) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

Given a larger matrix (3x3, 4x4…) it is required to find the smaller 2x2 determinants of a submatrix (minor) using a process called expansion by minors.
By expanding along the top row, cover up each row and column containing element a_{00} and find the determinant of the minor. Alternately add and subtract the matrix element times minor.



$$= a_{00}\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} - a_{01}\begin{vmatrix} a_{10} & a_{12} \\ a_{20} & a_{22} \end{vmatrix} + a_{02}\begin{vmatrix} a_{10} & a_{11} \\ a_{20} & a_{21} \end{vmatrix}$$

Here is an example of a 3x3 determinant:

$$\begin{vmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 5 & 6 & 5 \end{vmatrix} = 1\begin{vmatrix} 3 & 4 \\ 6 & 5 \end{vmatrix} - 2\begin{vmatrix} 2 & 4 \\ 5 & 5 \end{vmatrix} + 3\begin{vmatrix} 2 & 3 \\ 5 & 6 \end{vmatrix} = 1(-9) - 2(-10) + 3(-3) = -9 + 20 - 9 = 2$$

The following equation can be used to find the determinant of any general square matrix larger than 2x2:

$$\det(A) = \sum_{i=0}^{n-1} (-1)^i \left(a_{0,i}\right) \det\left(M_{0i}\right)$$

Where M_{0i} is the minor (submatrix created when removing row 0 and column i) and $a_{0,i}$ is the coefficient element of the matrix row 0 and column i.

(Continued on next page…)

(... Determined2 continued)

Here is an example of a 4x4 matrix:

$$\begin{matrix}1&2&3&4\\2&3&4&4\\5&6&5&2\\8&9&9&4\end{matrix} = 1\begin{pmatrix}3&4&4\\6&5&2\\9&9&4\end{pmatrix} - 2\begin{pmatrix}2&4&4\\5&5&2\\8&9&4\end{pmatrix} + 3\begin{pmatrix}2&3&4\\5&6&2\\8&9&4\end{pmatrix} - 4\begin{pmatrix}2&3&4\\5&6&5\\8&9&9\end{pmatrix} = \ldots$$

$$= 1(18) - 2(8) + 3(-12) - 4(-9) = 2$$

You can see why a program is the way to go for this project, especially for matrices larger than 3x3.

Input: The first line (N) consists of the number of data sets. Each data set contains a T by T matrix of integers.

Output: For each data set, print out the determinant.

Constraints:
1<=N<=10
3 <= T <= 10

Example Input File
3
1 2 3
2 3 4
5 6 5
1 2 3 4
2 3 4 4
5 6 5 2
8 9 9 4
1 2 3 9 8 7
2 3 4 4 4 3
5 6 5 2 9 8
8 9 9 4 6 5
3 2 1 5 4 3
9 8 7 5 4 5

Output to screen:
2
2
408