Computer Science Competition
Invitational A 2022
Programming Problem Set

I. General Notes

1. Do the problems in any order you like. They do not have to be done in order from 1 to 12.

2. All problems have a value of 60 points.

3. There is no extraneous input. All input is exactly as specified in the problem. Unless specified by the problem, integer inputs will not have leading zeros. Unless otherwise specified, your program should read to the end of file.

4. Your program should not print extraneous output. Follow the form exactly as given in the problem.

5. A penalty of 5 points will be assessed each time that an incorrect solution is submitted. This penalty will only be assessed if a solution is ultimately judged as correct.

II. Names of Problems

Number	Name
Problem 1	Artur
Problem 2	Carolina
Problem 3	Dhruv
Problem 4	Eva
Problem 5	Harold
Problem 6	Ishika
Problem 7	Krish
Problem 8	Mia
Problem 9	Paloma
Problem 10	Renata
Problem 11	Sheal
Problem 12	Tom

1. Artur

Program Name: Artur.java **Input File: None**

Artur is studying the Energy Pyramid. He is creating a chart about how the energy flows from the "Producers" at the bottom of the pyramid, to the "Decomposers" at the top. As he begins his course of study, can you produce an exact copy of the pyramid below so that he can fill in the details later.

Input: There is no input for this problem.

Note: The only characters used in the output of this problem are:

```
/ (forward slash)
\ (backslash)
- (minus sign)
> (greater than symbol)
% (percent symbol)
. (decimal point)
0 (zero)
1 (one)
```

Exact Output:

```
                   /\
                  /  \--->0.01%
                 /    \
                /------\
               /        \
              /          \--->0.1%
             /            \
            /--------------\
           /                \
          /                  \--->1%
         /                    \
        /----------------------\
       /                        \
      /                          \--->10%
     /                            \
    /------------------------------\
   /                                \
  /                                  \--->100%
 /                                    \
/--------------------------------------\
```

2. Carolina

Program Name: Carolina.java **Input File: carolina.dat**

Carolina is researching costs of computer-related purchases and would like some help to calculate her final costs for each item. She understands that each item will be taxed at 8.25% and would like to see that tax applied to each item individually as well as a single overall purchase. She wonders if there might be a difference in individual purchases versus one large purchase.

Write a program that produces a report for Carolina, formatted as shown in the sample below. For each item in her list, the report displays the individual cost, the sales tax for that single item, their sum, and the item description. All amounts must be rounded to 2 decimal places as shown.

Input: A list with no more than 15 items. Each line will contain the individual cost of a single item followed by its item description, separated by a single space. The individual costs will be between $5.00 and $1500.00 and the length of item descriptions will not exceed 50 characters.

Output: For each item, output a single line containing the following elements formatted as shown in the sample below.

- Individual cost: $76.53
- Sales Tax for individual item: $76.53 * 8.25% = $6.31
- Total individual item cost: $76.53 + $6.31 = $82.84
- Item’s description: Wireless Keyboard and Mouse

Below the list, display the subtotal cost of a combined purchase for all items below $689.66 plus sales tax $56.90 and the total purchase $746.56 in the same format with “Combined Single Purchase” as the description. The final line of output is the sum of the individual purchases with “Sum of Individual Purchases” as the description.

Sample input:
```
76.53 Wireless Keyboard and Mouse
117.75 HD LED Monitor 24"
453.39 Color Laser Printer
41.99 Wireless Headset with Microphone
```

Sample output:
```
$76.53 + $6.31 = $82.84 Wireless Keyboard and Mouse
$117.75 + $9.71 = $127.46 HD LED Monitor 24"
$453.39 + $37.40 = $490.79 Color Laser Printer
$41.99 + $3.46 = $45.45 Wireless Headset with Microphone
$689.66 + $56.90 = $746.56 Combined Single Purchase
$746.54 Sum of Individual Purchases
```

3. Dhruv

Program Name: Dhruv.java **Input File: dhruv.dat**

Dhruv loves to listen to AM radio, and who doesn't? There's sports talk, news talk, talk-talk, oldies, and just enough static to make it a challenge. Your job is to help him with his AM radio.

In AM radio, stations have frequencies which range from 535 to 1605, all values being whole numbers. Every station has a frequency and call letters. The call letters represent the station's "name."

When Dhruv types in a frequency into his AM radio, it takes him to the programmed station with the closest value to his input. There are 8 stations programmed into his radio.

First, you will read in the 8-programmed stations from a data file. Then, you will read in a series of inputs that Dhruv makes. Your job is to display the call letters of the station that the radio will select.

If the input is exactly halfway between two programmed stations, the radio will select the station with the higher frequency.

Input: The first 8 lines will be 8 sets of radio stations consisting of a 4-character string representing the station's call letters followed by the integer AM frequency of the station. Line 9 will be an integer representing N, the number of data sets to follow. N will be in the range of [1,25]. Each of the N data sets will consist of an integer representing the number that Dhruv entered as his station choice. The station choice will be an integer in the range of [0,9999].

Note: The AM stations exist in a range of 535-1605. Any input Dhruv makes outside of this range should generate the error "BAD INPUT."

Output: Each output will be a 4-character string representing the station the radio will select or will be the message "BAD INPUT"

Sample input:

```
KEYS 1410
KTTB 600
KFST 860
KTEM 1400
KBME 790
WOAI 1200
KRLD 1080
KKSA 1260
5
1200
880
1110
1700
1230
```

Sample output: *(Note: the output for each regular expression will all be on the same line.)*

```
WOAI
KFST
KRLD
BAD INPUT
KKSA
```

4. Eva

Program Name: Eva.java **Input File: eva.dat**

Ever since Eva started taking geometry, her favorite shape has been the triangle. She has the idea that she would like to write a program that given a height H, will output a triangle with H rows. For example, if 5 is given as the height, her code should output the below triangle:

```
    *
   ***
  *****
 *******
*********
```

Notice, the top row of the triangle has exactly one asterisk (*), and each subsequent row has two more asterisks than the row before it. The top row has 4 leading spaces, and each subsequent row has one fewer leading space than the row before it. There are no spaces after the last asterisk in any given line. There are no leading spaces on the bottom row of the triangle, either.

Can you help Eva write a program to output a triangle given an arbitrary height?

Input: Input will begin with a number N, the number of test cases. N will be in the range of [1,20]. Each of the following N rows will contain a single integer H, the height of the triangle to be output. H will be in the range of [1,30].

Output: Each output case will begin with “Start of Triangle #X” where X is the case number. Following, beginning on the next line, will be a triangle output to H lines. Each output case will end with “End of Triangle #X”.

Sample input:

```
5
5
3
10
15
9
```

See next page for sample output

Eva continued

Sample output:

```
Start of Triangle #1
    *
   ***
  *****
 *******
*********
End of Triangle #1
Start of Triangle #2
  *
 ***
*****
End of Triangle #2
Start of Triangle #3
         *
        ***
       *****
      *******
     *********
    ***********
   *************
  ***************
 *****************
*******************
End of Triangle #3
Start of Triangle #4
              *
             ***
            *****
           *******
          *********
         ***********
        *************
       ***************
      *****************
     *******************
    *********************
   ***********************
  *************************
 ***************************
*****************************
End of Triangle #4
Start of Triangle #5
        *
       ***
      *****
     *******
    *********
   ***********
  *************
 ***************
*****************
End of Triangle #5
```

5. Harold

Program Name: Harold.java **Input File: harold.dat**

In Harold's physics class, his teacher has introduced the idea of writing numbers in scientific notation. Scientific notation is a way to express numbers that are too large in a more conveniently written decimal form. In scientific notation, nonzero numbers are written in the form $m \times 10^n$or m times 10 raised to the power of n where n is an integer and the coefficient m is a nonzero real number. For example the number 30 000 in scientific notation would be $3 * 10^4$ and the number 987 in scientific notation would be $9.87 * 10^2$.

Can you help Harold write a program that if given a positive, non-zero whole number, can output the number in scientific notation?

Input: Input will begin with a number N, the number of test cases. N will be in the range of [1,10]. Each of the following N rows will contain a single, positive integer, I, representing the number that is to be converted to scientific notation. The number will be in range [1, 2147483647].

Output: For each test case, you are to output *I*=*m**10^*n* where m is the coefficient and n is the power to raise m to. The value of m is to be in the range of [1,10). For the purposes of this programming question, the caret character ^ (shift+6) will be used to denote an exponent. It will not denote the XOR. It is important to note that if no decimal places are needed for the coefficient m, no decimal places should be present in the output. The output should not have any blank or white space present. See the sample output below.

Sample input:

```
7
30000
987
123000
45000000
2
76
123456
```

Sample output:

```
30000=3*10^4
987=9.87*10^2
123000=1.23*10^5
45000000=4.5*10^7
2=2*10^0
76=7.6*10^1
123456=1.23456*10^5
```

6. Ishika

Program Name: Ishika.java **Input File: ishika.dat**

Ishika has gone to her local lumber yard looking for different wooden planks. She is working on a special project. Because she is on a tight budget, she makes her selection from the scrap wood pile that consists of lumber left over from previous cuts.

She must buy exactly two separate pieces, and she must get at least 20 feet in all. Her goal is to select the two pieces that give her the smallest total length.

Input: The first line consists of a number N, representing the number of lines of data to follow. N will be in the range of [1,50]. The next N lines of data consist of an integer T, followed by T integers representing the lengths of each piece of wood in that batch. Each input of T will be an integer in the range of [1,100].

Note: The length of each piece of wood will always be a whole number. If no two pieces have a combined length of at least 20 feet, output NOT POSSIBLE.

Output: Each output will be the combined length of the two planks that Ishika selects.

Sample input:

```
8
3 15 8 13
2 9 10
5 3 5 7 9 11
5 15 21 7 4 8
2 12 50
5 20 9 10 11 2
6 10 12 14 16 18 20
5 18 1 1 1 3
```

Sample output:

```
21
NOT POSSIBLE
20
22
62
20
22
21
```

7. Krish

Program Name: Krish.java **Input File: krish.dat**

Krish enjoys anything that involves numbers and has been playing with integers. He started with just summing the digits that make up an integer. He then moved on to finding the product of the non-zero digits of the same integer after he realized that multiplying by zero will always produce a zero product and that that just was not very exciting. Visually, reversing the digits creates another integer and Krish wonders what the products of the original integer and its “reverse” would look like. He decides it would be nice to have a program that verifies his results and has asked you for some help creating the program.

Write a program for Krish that produces the following transformations of a single non-zero integer like 3805 or -3805:

- Sum of the digits would be 3 + 8 + 0 + 5 = 16 with -3805 as -3 + -8 + 0 + -5 = -16
- Product of the non-zero digits would be 3 * 8 * 5 = 120 with -3805 as -3 * -8 * -5 = -120
- “Reverse” digits integer would be 5083 and be same sign, e.g., -3805 would be -5083
- Product of original and “reverse” integers would be 3805 * 5083 = 19340815

Input: The first line is a positive integer T ≤ 20, the number of test cases in the data file. That will then be followed by T lines, each containing a single integer N, -999999 ≤ N ≤ 999999 with N ≠ 0.

Output: For each N, output a single line containing the following items, separated by single spaces:

- Original integer
- Sum of digits
- Product of non-zero digits
- “Reverse” digits integer
- Product of original and “reverse” digit integers

Sample input:

```
5
3805
-127
123456
-802
40091
```

Sample output:

```
3805 16 120 5083 19340815
-127 -10 -14 -721 91567
123456 21 720 654321 80779853376
-802 -10 16 -208 166816
40091 14 36 19004 761889364
```

8. Mia

Program Name: Mia.java **Input File: mia.dat**

Mia works for the Universal Identity Lab in the identity matrix division. An identity matrix is an $n \times n$ matrix with all 1's along the diagonal, and 0's everywhere else. The general form of an identity matrix is below:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

So, given a 3×3matrix, the resulting identity matrix would be:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

As part of her job, Mia is responsible for shipping identity matrices to mathematicians and computer scientists all over the world! On her way to the post office with a package full of identity matrices, she accidentally dropped it, causing changes to some of the matrices. Some of the matrices were jumbled up, but still salvageable as an identity matrix, some matrices were altered such that they could no longer serve as an identity matrix, and the remaining matrices were unchanged.

For example the below jumbled matrix can be transformed back into an identity matrix as shown below:

$$\begin{array}{r} row\ 1 \\ row\ 2 \\ row\ 3 \end{array} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \rightarrow \text{Swap rows 1 and 2} \rightarrow \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \rightarrow \text{Swap rows 1 and 3} \rightarrow \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The below matrix, however, can not be transformed back into an identity matrix no matter the combination or number of swaps used:

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Mia needs your help determining if a given matrix is an identity matrix or not. If the matrix is an identity matrix, Mia needs to know whether swaps are needed to transform the original matrix into an identity matrix, or if no swaps are needed at all. Only allowed operations are row swaps. Column swaps are NOT allowed.

Input: Input will begin with a number N, the number of test cases. N will be in the range of [1,15]. Each test case will begin with a single integer D, the dimension of the square matrix. D will be in the range of [1,25]. The following D lines will each contain D numbers. The numbers will either be a 0 or a 1, separated by a space. This represents the matrix to be evaluated for the given case.

Mia continued

Output: For each case you are to output either: “`Matrix X: Identity Matrix - No swaps needed`” if the original matrix was an identity matrix in its original form (ie, no swaps were needed), “`Matrix X: Identity Matrix - Swaps needed`” if the original matrix could be transformed into an identity matrix with the use of swaps, or “`Matrix X: This is not an Identity Matrix`” if the matrix can not be transformed into an identity matrix, no matter the combination of swaps used.

Sample input:

```
6
3
1 0 0
0 1 0
0 0 1
3
0 1 0
0 0 1
1 0 0
3
1 0 1
0 1 0
0 0 1
4
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
4
0 0 1 0
0 1 0 0
0 0 0 1
1 0 0 0
5
0 0 1 0 0
0 1 0 0 0
0 0 0 1 0
1 0 0 0 0
0 1 0 0 0
```

Sample output:

```
Matrix 1: Identity Matrix - No swaps needed
Matrix 2: Identity Matrix - Swaps needed
Matrix 3: This is not an Identity Matrix
Matrix 4: Identity Matrix - No swaps needed
Matrix 5: Identity Matrix - Swaps needed
Matrix 6: This is not an Identity Matrix
```

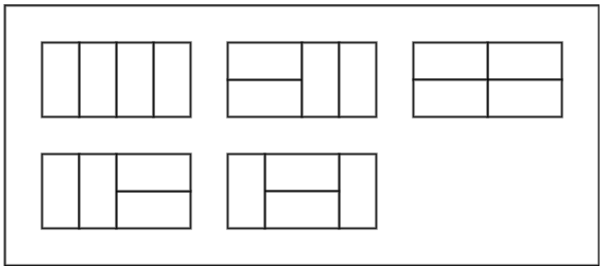
9. Paloma

Program Name: Paloma.java **Input File: paloma.dat**

Paloma is taking a trip to the beach today. Unfortunately, it is very crowded. So crowded that there is just barely enough room for all N beach-goers to lay their towels down. This makes her wonder - how many different ways are there to arrange the beach-goers' towels?

There is a 2 by N meter area where beach towels can go, and each beach towel is 1 by 2 meters, and can be laid either vertically or horizontally.

For example, when N = 4, the possible towel arrangements are as follows:



Input: Input will begin with a single integer T, the number of test cases to follow. The following T lines each contain a single integer N (0 < N <= 20), the number of beach towels that can fit on the beach for that test case.

Output: For each test case, output a single line containing the number of possible ways to arrange N towels on the beach.

Sample input:

```
3
1
2
4
```

Sample output:

```
1
2
5
```

10. Renata

Program Name: Renata.java **Input File: renata.dat**

Renata enjoys archery and has created a scoring process using a square grid. She has assigned values for each of the squares in the grid and wants to be able to record the positions and later convert them into scores. Here is a sample scoring grid:

Columns → Rows ↓	1	2	3	4	5
1	7	8	8	8	7
2	8	9	9	9	8
3	8	9	10	9	8
4	8	9	9	9	8
5	7	8	8	8	7

An arrow in position (2,4) would score 9 points and position (5,1) would score 7 points. An arrow that completely misses the target would be recorded as (0,0) and would score 0 points. Renata has asked for your help to create a scoring program.

Input: The first line is a positive integer T ≤ 10, the number of test cases in the data file. That will then be followed by T sets of data. For each set, the first line will contain 1 integer (N) the number of rows and columns in the scoring grid, with 5 ≤ N ≤ 15. The dataset continues with N rows, each containing N integers (S) to populate the scoring grid with 0 ≤ S ≤ 15. The next line will be the number of positions recorded. Each position appears on a separate line as two integers, the row and column position of the grid.

Output: For each test case, output the test case number followed by a colon, a space, and the total score.

Sample input:

```
2
5
7 8 8 8 7
8 9 9 9 8
8 9 10 9 8
8 9 9 9 8
7 8 8 8 7
2
2 4
5 1
6
6 8 8 8 8 6
7 8 9 9 8 7
8 9 10 10 9 8
8 9 10 10 9 8
7 8 9 9 8 7
6 8 8 8 8 6
5
2 4
5 1
0 0
2 2
3 4
```

Sample output:

```
1: 16
2: 34
```

11. Sheal

Program Name: Sheal.java **Input File: sheal.dat**

Sheal plans to open a candy shop, and his first big idea to draw in customers is to make a big sign with the name of his shop. The only problem is, he can't decide on what to name it. He's narrowed down his choices since he only has a few available letters to put on his sign. Sheal's plan to pick a name is to choose some number K, and then set the store's name to the K'th smallest name possible in lexicographical order.

For example, if the letters available were "AABC", all possible names in order would be:

1: A	2: AA	3: AAB
4: AABC	5: AAC	6: AACB
7: AB	8: ABA	9: ABAC
10: ABC	11: ABCA	12: AC
13: ACA	14: ACAB	15: ACB
16: ACBA	17: B	18: BA
19: BAA	20: BAAC	21: BAC
22: BACA	23: BC	24: BCA
25: BCAA	26: C	27: CA
28: CAA	29: CAAB	30: CAB
31: CABA	32: CB	33: CBA
34: CBAA		

Write a program to determine Sheal's choice of shop name.

Input: Input will begin with a single integer T(T <= 10), the number of test cases to follow. The following T lines will each contain one test case. Each test case consists of a string of available letters S (|S| <= 8), and Sheal's choice of K. K is guaranteed to be less than or equal to the number of possible names.

Output: For each test case, output a single line containing the name of Sheal's shop.

Sample input:

```
3
AABC 17
AABC 34
AEHLSS 836
```

Sample output:

```
B
CBAA
SHEALS
```

12. Tom

Program Name: Tom.java **Input File: tom.dat**

Tom likes words and is reading the dictionary for fun. More specifically, Tom really likes words that have lots of the same character. In order to decide on what his favorite words are, Tom has come up with a point-based system to assign a value to every word.

Each word's value starts at zero, and Tom reads each character in word from left to right. The value of each character is equal to 1 plus the number of times he has seen this character to the left already.

For instance, the word "dictionary" would be granted 1 + 1 + 1 + 1 + 2 + 1 + 1 + 1 + 1 + 1 points for a total of 11.

Calculate Tom's score for a given word.

Input: Input will begin with a single integer T, the number of test cases to follow. The following T lines each contain a single word consisting of only lowercase characters a-z.

Output: For each test case, output Tom's word score.

Sample input:

```
3
tom
dictionary
mississippi
```

Sample output:

```
3
11
24
```