PLU February 2023 Programming Contest

Advanced Division

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. Problems will have either no input or will read input from a specified file. All output should be to standard output (the monitor).
3. All input is as specified in the problem. Unless specified by the problem, integer inputs will not have leading zeros.
4. Your program should not print extraneous output. Follow the form exactly as given in the problem.

II. Names of Problems

Number	Name
Problem 1	Not my Forte
Problem 2	Metrognome
Problem 3	Chair Test
Problem 4	All About That Bass
Problem 5	Concerto in C Minor
Problem 6	Marching Orchestra
Problem 7	Quartet
Problem 8	Rosin for Sale
Problem 9	Bootleg Sheet Music
Problem 10	Perfect Pitch
Problem 11	Bach to the Future
Problem 12	Piano Movers

1. Not my Forte

Input File: forte.dat

After several frustrating rehearsals, Richard, an orchestra director, finally decided that it was time to come up with an easier system for the musical notation of forte (loud) and piano (soft). A simple forte is easy enough to understand, but there exist other variations, namely fortissimo (louder), fortississimo (even louder), pianissimo (softer), pianississimo (even softer), and so on. To make things less complicated for new students, Richard has come up with a numerical system to make memorizing the dynamics of a music piece a breeze. For every increase in number (beyond zero), the piece should be played louder and louder, and for every decrease in number (beyond zero), the piece should be played softer and softer. Since writing computer programs is not Richard's forte, help him out! Make a program that takes in a numerical value and prints out the corresponding dynamic (forte, piano, etc.).

Input

The input begins with the number `t` (`1 <= t <= 100`), the number of test cases to follow. For each of the following `t` test cases, a line with a single integer `d` (`1 <= |d| <= 20`), the dynamic value of the piece, will appear. (Note: `|x|` = the absolute value of `x`).

Output

For each of the t test cases, print out a single line with a string corresponding to the given dynamic value. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
5
-1
-2
1
2
5
```

Example Output to Screen

```
piano
pianissimo
forte
fortissimo
fortississississimo
```

2. Metrognome

Input File: metrognome.dat

Let me tell you, getting garden gnomes to dance in sync while listening to Moonlight Sonata is no easy task. The best we can hope for is that the garden gnomes eventually all line up on a single beat or two (since all garden gnomes dance at their own pace). More specifically, each garden gnome, G, has a certain value, Gi. At each integer multiple of Gi, (Gi, 2 * Gi, 3 * Gi, etc.), the garden gnome G will perform a dance move. Your task is to find out, for a given set of garden gnomes: what is the first point in time at which all the garden gnomes perform a dance move simultaneously?

Input

The input begins with the number t (1 <= t <= 100), the number of test cases to follow. For each of the following t test cases, a line with a single integer n (1 <= n <= 1000), the number of garden gnomes, will appear. The next line will consist of n space-separated integers G1, G2 … Gn (1 <= Gi <= 100).

Output

For each of the t test cases, print out a single integer: the first point in time during which all garden gnomes are dancing simultaneously. An answer is guaranteed to exist. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
3
5
1 2 3 4 5
2
1 2
3
7 8 3
```

Example Output to Screen

```
60
2
168
```

3. Chair Test

Input File: test.dat

The time is here, the most nerve-wracking moment ever in the life of an orchestra member: chair test day! Initially, all of the chairs in the orchestra are up for grabs (and are empty). Then, for each orchestra member starting from 1 up to the final member, the following operation is performed:

1. Find the longest subarray (continuous subsegment) consisting of empty seats (if there are multiple, choose the leftmost one).

2. Denote the subarray as [l : r]. if r – l + 1 is odd, then set the ((l + r) / 2)th chair to the number of the current orchestra member. If r – l + 1 is even, set the ((l + r – 1) / 2)th chair to the number of the current orchestra member.

This operation is repeated once for each member of the orchestra, meaning that a unique valid answer always exists. For example, consider the case that there are 5 orchestra members. The sequence of moves will look like this:

```
[0,0,0,0,0]->[0,0,1,0,0]->[2,0,1,0,0]->[2,0,1,3,0]->[2,4,1,3,0]->[2,4,1,3,5]
```

Input

The input will begin with a single integer, `t` (`1 <= t <= 100`), the number of test cases to follow. For each test case, there will be a single line containing the integer `n` (`1 <= n <= 100`), the number of orchestra members.

Output

For each test case, output a single line containing `n` space-separated integers denoting the final seating arrangement of the orchestra members.

Example Input File

```
6
6
5
4
3
2
1
```

Example Output to Screen

```
3 4 1 5 2 6
2 4 1 3 5
3 1 2 4
2 1 3
1 2
1
```

4. All About That Bass

Input File: bass.dat

After years of wondering why he is never appreciated for his contributions to the orchestra, Spencer believes that he has come up with a formula to explain why bassists are so underappreciated. Here is what he has come up with:

$$Appreciation = \frac{N^q}{\sqrt[2]{|42 * sin\, sin\, T|}}$$

Where N represents the number of notes played, q represents the overall quality of the performance, T represents the length in minutes of the performance, |x| represents the absolute value of x, (and 42 is Bach’s constant of linear musical congruence). Given the necessary information, can you help Spencer calculate the orchestra’s appreciation for his music?

Input

The input will begin with a single integer, t (1 <= t <= 100), the number of test cases to follow. For each test case, there will be a single line containing the integers N, q, and T (2 <= N, q <= 10, 1 <= T <= 6), described above.

Output

Output a single decimal value, the appreciation of the orchestra of Spencer’s music, rounded to two decimal places. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
3
3 1 1
2 3 2
9 9 5
```

Example Output to Screen

```
0.54
1.39
66073745.82
```

5. Concerto in C Minor

Input File: concerto.dat

Mr. Bey Toven is a big fan of songs written in minor keys. In fact, he sees it as an insult to his ears to hear the childish, playful melodies of songs written in major keys. After getting frustrated with the constant annoying tunes of his neighbor, Mr. Vee Valdee, Mr. Toven has come to ask you for help. Your task is to finish the last part of Mr. Toven's all-new Minor Counter 5001. With this new tool, Bey Toven wants to be able to count the number of times the word "minor" (case sensitive) appears in a particular sentence. If the count is greater than two, Mr. Toven deems it a worthy sentence. If not, he gets all upset that the music in this world is going downhill due to the "Fake Melodies" written in the major keys. Help Mr. Toven create this new calculator so that he can listen to his music in peace!

Input

The input will begin with a single integer, `t` (`1` <= `t` <= `100`), the number of test cases to follow. For each test case, there will be a single line containing a sentence to be scanned by the Minor Counter 5001. The sentence may contain punctuation, special characters, numbers, and English letters (`1` <= length of sentence <= `1000`).

Output

For each test case, you must print a single line of output. If the sentence is deemed worthy by the Minor Counter 5001 (it has a count greater than two), print "`It's over Vee Valdee! I have the high ground!`" (without the quotation marks). If the sentence is not up to par, print "`You underestimate my music!`" (without the quotation marks). Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
2
The best word in the English language is minor. Minorminorminor!
Major chords please my ears. So harmonious and beautiful!
```

Example Output to Screen

```
It's over Vee Valdee! I have the high ground!
You underestimate my music!
```

6. Marching Orchestra

Input File: marching.dat

Unsurprisingly, the idea of creating the world's first "marching orchestra" was not popular with the cellists and bassists at Quirky Academy. Despite the moans and groans saying "Oh, I can barely lift my instrument!" and "How am I supposed to play, walk, and carry my cello?!", the marching orchestra was created. Now, you need to help the orchestra get into formation. Given a bird's-eye view of a rectangular football field, help the marching orchestra compute the minimum time it takes for each member to get to their specified destinations. It is allowed for multiple orchestra members to occupy the same square at the same time. Each orchestra member will be denoted in the input by a capital letter (A, B, C, etc.) and each destination will be denoted by a lowercase letter (a, b, c, etc.). For each orchestra member, the contribution to the total time it takes is the time it takes them to go from their starting point to their respective destination (for example, A needs to go to a). The total time is the sum of all of the times for each orchestra member. Please help these disgruntled musicians find their way around on the field at halftime! Note: musicians can only move in the four cardinal directions (NESW). Also, each musician will have a valid, reachable destination. It is easy to see that an answer will always exist. Musicians cannot exit the football field.

Input

The input will begin with a single integer, `t` (`1 <= t <= 10`), the number of test cases to follow. For each test case, the first line will contain two integers, `n` and `m` (`1 <= n, m <= 10`), denoting the number of rows and columns, respectively. The next `n` lines will contain `m` characters each, the description of the field. Each character will be one of the following: an uppercase letter (`A-Z`), denoting a musician, a lowercase letter (`a-z`), denoting a destination, a '`.`' character, denoting an empty space on the field (musicians can move through these), or a '`#`' character, denoting a cheerleader (musicians can never occupy the same square as a cheerleader; cheerleaders remain stationary in their squares). Musicians are allowed to walk through the destinations of other musicians.

Output

Output a single line containing a single integer, the total time it takes for the marching orchestra to get into position (meaning the sum of all the times it takes for each musician to get into position). Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
2
5 5
#####
ECdDa
eA##B
Fbf#c
#####
1 4
AaBb
```

Example Output to Screen

```
19
2
```

7. Quartet

Input File: quartet.dat

Every year, the orchestra at The School for Extremely Intelligent Young Musicians and Performing Artists puts on a quartet (a musical performance utilizing four members). Of course, they want to choose the four best musicians with only one restriction: they must choose exactly one violinist, one violist, one cellist, and one bassist. Each student at TSEIYMPS plays exactly one instrument and has a distinct (unique) skill level. Given this, can you construct the best possible quartet while following the rules and also print out the total skill level (the sum of the four members' skill levels) of the quartet? As always, the orchestra director has left this task to you, the member of the orchestra who happens to know a thing or two about algorithms.

Input

The input begins with the number `t` (`1 <= t <= 100`), the number of test cases to follow. For each of the following `t` test cases, a line with four integers `a`, `b`, `c`, and `d` (`1 <= a`, `b`, `c`, `d <= 1000`), the number of violinists, violists, cellists, and bassists, respectively, will appear. The next four lines will consist of `a` space-separated integers, then `b` space-separated integers, then `c` space-separated integers, then `d` space-separated integers, the skill levels of all students at TSEIYMPS. It is guaranteed that the total skill level of the best quartet will fit within a standard signed 32-bit integer. It is also guaranteed that the skill level of each student is a positive integer value.

Output

Print a single integer, the total skill level of the best possible quartet. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
2
1 1 1 1
5
5
5
5
1 2 3 4
2
9 10
70 3 2
100 42 212 3
```

Example Output to Screen

```
20
294
```

8. Rosin for Sale

Input File: rosin.dat

The local instrument shop is having its annual sale on premium rosin and naturally you are the first in line to check out! As you are an amateur coin collector when you aren't playing in the orchestra, you want to pay for your rosin while using the fewest number of coins possible (so as to not damage your collection). You have an unlimited supply of each of a given set of denominations of coins and you also know how much the rosin costs. Now all that is left is to create a program that computes the minimum number of coins or determine that you cannot pay the exact amount you need to (no more, no less).

Input

The input begins with the number `t` (`1 <= t <= 100`), the number of test cases to follow. For each of the following `t` test cases, a line with a single integer `n` (`1 <= n <= 100`), the number of types of coins you have, will appear. The next line will consist of `n` space-separated integers, the denomination (value) of each coin (`1 <= value <= 100`, all values are pairwise distinct). Finally, for each test case, you will be given an integer, `c`, the cost of the rosin (`0 <= c <= 100`). You can use each type of coin as many times as necessary.

Output

Print a single line of output for each test case, the minimum number of coins such that you can exactly pay the cost of the rosin. If it is not possible, print "`No rosin for you! Come back in one year!`" (without the quotation marks) instead. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
3
5
1 5 10 25 50
30
3
1 2 3
0
4
3 5 7 9
2
```

Example Output to Screen

```
2
0
No rosin for you! Come back in one year!
```

9. Bootleg Sheet Music

Input File: none

After realizing that sheet music costs money (I know, shocking), you decided to pirate some off of the “Deep Dark Web 2.0 ™” instead of paying for it with your hard-earned money. The only problem is that the resolution of the files you downloaded was really, really low. After trying to decipher each note for countless hours, you finally give up and fall asleep at your music stand. Your friend, a professional ASCII artist, sees your sheet music and acquires some inspiration. Please recreate what he drew in his text editor.

Input

There is no input for this problem.

Output

Output the text found under the “Example Output to Screen” section exactly as it appears. Do not print trailing whitespace characters.

Example Output to Screen

```
   |~~~~~~~~~~~~~|
   |~~~~~~~~~~~~~|
 __|            __|
/~~\          /~~\
\__/          \__/
```

10. Perfect Pitch

Input File: pitch.dat

Being a true “Renaissance Woman”, Alexa has talents ranging from having a mental Java compiler (very helpful on written tests) to having perfect pitch: the ability to assign a certain pitch to its respective note through only hearing the pitch. Of course, Alexa’s pure intellectual superiority over her friends has made them quite jealous (and annoyed), so now you are stuck with her. The purist she is, Alexa will correct you for any mistakes you make while she listens to you practice viola. If you play just a tad bit sharp or flat, you will be painfully reminded by her shrill voice saying, “That’s supposed to be A FLAT, not G SHARP!” Being rightfully confused (after all, they are the same note), you have decided to come up with a virtual version of Alexa to help you uselessly correct yourself while you play. Given a musical note, write a program to return the “corrected” version of the note as Alexa would using the following chart:

Note	**“Corrected” Note**	**Note**	**“Corrected” Note**
A Flat	G Sharp	A Sharp	B Flat
B Flat	A Sharp	B Sharp	C Flat
C Flat	B Sharp	C Sharp	D Flat
D Flat	C Sharp	D Sharp	E Flat
E Flat	D Sharp	E Sharp	F Flat

Note: musical notes that are not listed in the chart will not appear anywhere in the input.

Input

The input will begin with a single integer, `t` (`1 <= t <= 100`), the number of test cases to follow. For each test case, a single note to be corrected will appear on its own line.

Output

For each given note, output the “corrected” note according to the given table. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
3
A Flat
C Sharp
E Sharp
```

Example Output to Screen

```
G Sharp
D Flat
F Flat
```

11. Bach to the Future

Input File: bach.dat

After filming was finally completed for Back to the Future XI (not as impressive now that time travel actually exists), Marty McFly decided to take a vacation to 1700 in order to visit his good friend, Johann Sebastian Bach. After fiddling with the calculations for a while in his DeLorean, he has come to you to help him figure out the order of certain events. Given a list of events (all of which happened on the same day), and a list of times (all of which are on the same day and are on the same day as the events), please help Marty determine which events happened at or before each given time.

Input

The input will begin with a line containing two integers, `n` (`1` <= `n` <= `100`) and `q` (`1` <= `q` <= `100`), denoting the number of events, and number of times that Marty wants to query, respectively. The next `n` lines will contain a string representing the name of an event and a time (in the format `HH:MM`, 24-hour style) representing when the event happened on the day. It is guaranteed that the name of each event will only be one word (meaning that it will not contain whitespace characters). Times could be single digits or double-digits. Finally, the next `q` lines will contain a time (in the format `HH:MM`, 24-hour style) representing an individual query by Marty. All times will be within the range `00:00` – `23:59` (inclusive).

Output

For each of the `q` queries, print out a single line containing a space-separated list of the names of the events that occur at or before the time given for the query. Make sure to print the names of the events in chronological order. If no events occur before or at a given time, print "`Why Don't You Make Like a Tree and Get Outta Here?`" (without the quotation marks). Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
5 5
Lunch-With-Bach 12:00
Playing-Cello 13:50
Teach-Bach-Cantina-Band 22:00
Chicken-Dinner 19:30
Go-Home 23:59
12:00
00:00
23:58
13:36
16:55
```

Example Output to Screen

```
Lunch-With-Bach
Why Don't You Make Like a Tree and Get Outta Here?
Lunch-With-Bach Playing-Cello Chicken-Dinner Teach-Bach-Cantina-Band
Lunch-With-Bach
Lunch-With-Bach Playing-Cello
```

12. Piano Movers

Input File: piano.dat

After earning millions from his piano concert at Carnegie Hall, Albert thought that it was time to move into a bigger home. The only problem with his idea was that his 24-karat gold piano (quite possibly the heaviest to ever exist) needed to be dragged slowly across a long path of 1-meter-long strips of road. Albert’s piano is exactly 2 meters long and moving it over a gap (an area with no road built) that is strictly longer than 2 meters is near impossible. Given a description of the road that Albert must move his piano along, please help him find the minimum number of 1-meter-long strips of road that Albert must fill in (covering up gaps) in order to safely move his piano to his new mansion.

Input

The input begins with the number `t` (`1 <= t <= 100`), the number of test cases to follow. For each of the following `t` test cases, a line with a single integer `n` (`1 <= n <= 50`), the length (in meters) of the road to Albert’s new home, will appear. The next line will consist of `n` characters (‘`.`’ denotes a gap, ‘`-`’ denotes a tile of road), the description of the road.

Output

For each of the `t` test cases, print out a single integer: the minimum number of additional road tiles that must be added in order for the piano to be able to be safely moved. Do not print trailing whitespace characters following and/or in between test cases.

Example Input File

```
5
10
.-.-.-.-.-
3
...
20
..-...-...-..-.....-
2
..
7
-......
```

Example Output to Screen

```
0
1
3
0
2
```