Consortium for Computing Sciences in Colleges
Southeastern Virtual Programming Competition
Saturday, January 23rd, 2021, 10 AM – 1 PM EST
Hosted by University of North Carolina at Asheville

There are ten (10) problems in this packet. Each team member should have a copy of the problems. These problems are NOT necessarily sorted by difficulty. You may solve them in any order. Remember input/output for the contest will be from `stdin` to `stdout`. `Stderr` will be ignored. Do not refer to or use external files in your source code. Extra white space at the end of lines is ignored, but extra white space at the beginning or within text on a line is not ignored. Reminders for our virtual competition:

- Teams may be comprised of 1 – 3 students.
- Internet access is allowed.
- Video, audio, or text conferencing between team members is allowed, as is any file transfer.
- Any team found to be communicating with someone other than their teammates for assistance will be disqualified.
- Four programming languages are allowed: C#, C++, Java 11, Python 3.8.
- An awards ceremony will be held at 1:30 PM EST on the general conference Zoom link.
- Have a lot of fun and good luck! ☺

Problem 1

Military Time

Military time is a method of measuring the time based on the full twenty-fours of the day rather than two groups of twelve hours. For example, 0530 is 5:30 AM and 1730 is 5:30 PM. You are writing a program that inputs two military times for a particular soldier. The first time input represents when the soldier first logs in for work, and the second time input is when he or she logs out. Your program should calculate the total hours and minutes they worked. Many members of the military work during the third shift so they may log in one evening, and log out the following day.

Input

The first line of input will be a positive integer *n* representing the number of test cases. This will be followed by *n* lines of input consisting of two integers. The first integer at the start of the line is the time when someone logs in and the second integer is the time is when he or she logs out. The two integers will each be four digits and separated by a single space. They will always be valid times in the range from 0000 up to 2359.

Output Corresponding to Sample Input

The output should be formatted exactly as shown below where each line indicates the hours and minutes worked between the two times. Use the singular word `hour` and/or `minute` when you only have one of either.

Sample Input

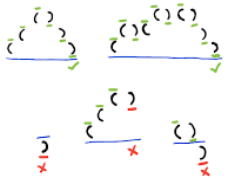
```
5
0900 1730
1730 0900
0030 0031
0031 0030
1231 1430
```

Sample Output

```
8 hours 30 minutes
15 hours 30 minutes
0 hours 1 minute
23 hours 59 minutes
1 hour 59 minutes
```

Problem 2

Balancing Delimiters



Many encoded strings contain delimiters. A delimiter is a non-empty string that acts as a boundary between different parts of a larger string. The delimiters involved in this problem occur in pairs that must be balanced with each pair having an open delimiter and a close delimiter. The open and close delimiters will always be different.

Input

The first line of input will contain a single integer *n*, which represents the number of test cases. You may assume that $1 \leq n \leq 100$. This will be followed by *n* test cases. Each test case consists of two lines. The first line contains the open and close delimiters. Each delimiter is a sequence of nonspace characters of length less than ten consisting of letters, digits, and/or symbols. The second line will be a string of one or more tokens to be checked for balanced delimiters. Assume all delimiters and other tokens are preceded by at least one space unless they occur at the start of a line.

Output

For each test case, you should print the string checked followed by whether or not it is balanced in the format below.

Sample Input

```
6
( )
( x + ( y )
{ }
{ { 3 + 4 } }
<B> </B>
<B> Make this text bold </B>
<B> </B>
<B> Make this text bold </UB>
[ ]
[ [ [ UNCA ] ] ]
fish fish2
red fish blue fish2 green fish yellow fish2
```

Output Corresponding to Sample Input

```
( x + ( y ) is not balanced
{ { 3 + 4 } } is balanced
<B> Make this text bold </B> is balanced
<B> Make this text bold </UB> is not balanced
[ [ [ UNCA ] ] ] is balanced
red fish blue fish2 green fish yellow fish2 is balanced
```

Problem 3

Millennial Millinery

Melissa is opening a new millinery business in downtown Asheville. It is a hat store called Millennial Millinery. For now, Melissa has decided to sell just one type of hat, but it will vary by size and material. She would like your help in determining the prices of hats. The type of material will determine the price of a hat and the total area of material used to make the hat.

In this problem, we will assume that the human head and the hat will be circular in shape rather than elongated. For a given hat, you will need to determine the total area of material used. Let *r* be the radius of the wearer's head in inches. Each hat consists of three parts:

- The top of the hat, which is a circle of radius *r*.
- The middle of the hat, which is a cylinder without its two bases. This cylinder has a circumference equal to that of the top of the hat, and a height of 4 inches.
- The brim of the hat, which is 3 inches wide. Thus, the brim occupies the annular region between two concentric circles having radii of *r* and *r + 3* inches.

Each hat is made of one kind of material. Melissa makes hats of three possible materials, and each type has a different price per square inch:

- Velvet, 25 cents
- Straw, 35 cents
- Wool, 45 cents

The hats come in various sizes. Hat sizes range from 6 to 8, in increments of one-eighth. Thus, the second largest hat size is $7\frac{7}{8}$ and the second smallest is $6\frac{1}{8}$. The relationship between the hat size and the size of the wearer's head is as follows. If h is the hat size, then the circumference of the head and of the hat is $\frac{25}{8}h$. For example, a size 7 hat has a circumference of $21\frac{7}{8}$ inches.

Input

The first line of input will contain a single integer *n*, indicating the number of hats to price. Each of the remaining *n* lines of input will contain information about one hat. The format of such a line is as follows:

```
Model <number> <material> <size>
```

Where `<number>` is a two-digit positive integer, `<material>` is "velvet", "straw", or "wool"; and `<size>` is either a whole number 6, 7, or 8, or a mixed numeral between 6 and 8 rounded to the nearest one-eighth in lowest terms. A mixed numeral will have one space between the whole number and fraction.

Output
Your program will print one line for each hat. The output format is as follows:
`Model <number> $ <price>`

Where `<price>` is an amount of dollars expressed to exactly two decimal places. Be sure to print exactly one space on either side of the dollar sign.

Sample Input
```
3
Model 70 wool 6
Model 97 velvet 7 1/2
Model 86 straw 6 5/8
```

Output Corresponding to Sample Input
```
Model 70 $ 84.38
Model 97 $ 59.01
Model 86 $ 72.56
```

Problem 4

The Typing Robot

Maurice has built a robot, and now he wants to teach it how to type on a keyboard. The first task for the robot is to be able to type numbers with one hand. As a preliminary experiment, Maurice wants the robot to type numbers that fit a specific pattern. The digits of the number are arranged so that it is only necessary for the robot's hand to move in a single direction, from left to right, and never right to left, while typing the number. The left-to-right motion also allows the robot's hand to become momentary stationary while typing several consecutive instances of the same digit.

From left to right, the digits on the robot's keyboard are just like yours: 1,2,3,4,5,6,7,8,9,0. In Maurice's experiment, if the robot depresses the digit 5, the next digit of the number may be 5, 6, 7, 8, 9, or 0, but it cannot be 1, 2, 3, or 4.

Maurice needs help in determining how many numbers fit this monotonic left-to-right pattern. Given the number of digits, the starting digit, and the final digit, Maurice wants to know how many positive integers exist such that the robot's hand should never move left.

For example, here is a list of all five-digit numbers beginning with one and ending with three that fit this pattern: 11113, 11123, 11133, 11223, 11233, 11333, 12223, 12233, 12333, 13333. Note that while typing a number it is possible to skip a digit. 13333 skips the digit 2.

An instance of this problem consists of three integers:

- *n*, the number of digits
- *A*, the starting digit
- *B*, the final digit

You may assume that either $A = B$, or B is located to the right of A on the keyboard. You may also assume that $2 \leq n \leq 40$. You need to determine how many n-digit positive integers whose leftmost digit is A and rightmost digit is B can by typed by the robot without having to move its hand ever to the left.

Input

The first line of the input will contain a single integer, indicating the number of test cases. Each test case appears on its own line. Each test case will indicate the values of *n*, *A*, and *B*, in that order.

Output
The output should display the solution to each test case, expressed as a single integer, one per line.

Sample Input

```
8
7 7 7
3 4 5
10 8 0
5 1 3
40 1 0
4 2 9
2 4 6
37 4 0
```

Output Corresponding to Sample Input

```
1
2
45
10
1362649145
36
1
4496388
```

Problem 5

Evil Villains Identify Links

The Union of Underground Supervillains includes many nefarious evildoers, including Devil Digger, Dirty Trickster, Terror Tunneller, and the Underwhelmer. These villains have decided to band together for their common defense from over-eager superheroes. Each villain has an underground lair. In total, there are $1 < L < 100$ lairs. To more conveniently travel between lairs, the villains will dig tunnels connecting their lairs. However, the process of digging tunnels is extremely difficult and time-consuming, so they would like to minimize the amount of digging required. Write a program to determine minimum tunneling distance needed to connect the underground lairs so that it is possible to travel from any lair to any other (possibly passing through other lairs in the process). To simplify, you may assume all tunnels are in a straight-line directly connecting one lair with another.

Input

The first line of input will contain a single integer *N*, which will indicate the number of test cases. You may assume that $1 \leq N \leq 100$. Each of the *N* test cases will start with a line containing a positive integer *L* representing the number of lairs. The following *L* lines each consists of a series of *L* numbers representing the distances to the other lairs.

Output

For each test case, the program should print the case number in the format below followed by a single number representing the minimum total distance of tunnels so that travel between the lairs is possible. In the first test case below between the four lairs, the minimum distance of tunnels to be dug, 15, occurs by connecting the first and second lairs (distance 3), the second and third lairs (distance 7), and the third and fourth lairs (distance 5).

Sample Input

```
2
4
0 3 10 8
3 0 7 12
10 7 0 5
8 12 5 0
2
0 2
2 0
```

Output Corresponding to Sample Input

```
Case #1: 15
Case #2: 2
```

Problem 6

Efficient Tasks

Cliff has a lot of things to do, and not a lot of time to do them. Each task on his to-do list can be modelled with tuple (a, b) of two durations of time: The first is the duration for Cliff's hands-on involvement of that task, and the second the duration of time he can be hands off (the task can finish by itself). For instance, washing clothes might take 5 minutes for Cliff to load the washing machine, but the cycle once he starts, it takes 40 minutes (in which he can do another task). This task would be represented as (5, 40).

Given a set of tasks, Cliff wants to accomplish them as quickly as possible. For instance, consider two tasks: task A (10, 5) and task B (5, 40). If Cliff does task A first, and task B afterwards he can complete all tasks in 55 minutes. However, if Cliff starts task B first, he can complete all tasks in 45 minutes.

Cliff has asked you to figure out the correct order in which to do his to-do list. However, there is one more wrinkle: some tasks depend on others. For instance, in Cliff's previous example task B may depend on task A to fully complete before it can be started. In that case, the time to complete A and B would be 60 minutes.

Given a set of tasks and their dependencies, determine the shortest amount of time it would take to do them.

Input

The input will start with a single integer $n \leq 10$ denoting how many cases will follow. Each case will start with a number $T \leq 1000$ denoting the number of tasks to follow. Each task will be formatted as a list of numbers. The first number will be a non-negative integer $k \leq 1000$ denoting the task's id. The next two numbers are the hands-on and hands-off times of that task respectively. The next number D, which is nonnegative, denotes how many tasks this task depends on. Following D, there will be D numbers denoting task ids that this task depends on

Output

For each case, print the minimum time in minutes it would take Cliff to complete all of the tasks in the case.

Sample Input

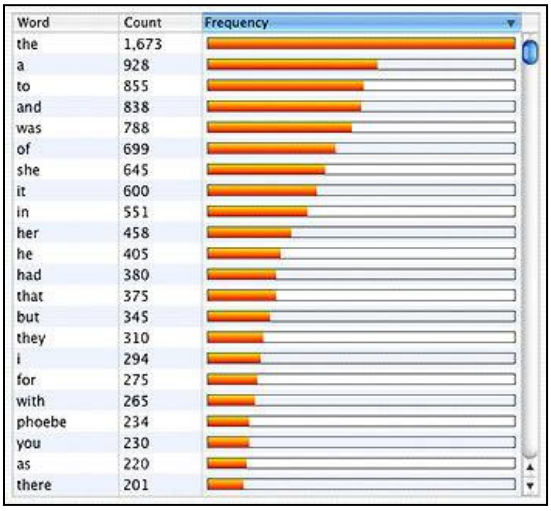
```
2
2
0 10 5 0
1 5 40 0
2
0 10 5 0
1 5 40 1 0
```

Output Corresponding to Sample Input

```
45
60
```

Problem 7

Word Stats



Word	Count	Frequency
the	1,673	
a	928	
to	855	
and	838	
was	788	
of	699	
she	645	
it	600	
in	551	
her	458	
he	405	
had	380	
that	375	
but	345	
they	310	
i	294	
for	275	
with	265	
phoebe	234	
you	230	
as	220	
there	201	

We all know how to crunch statistics with numbers. But, let us crunch some statistics with words! That is your job with this program. You want to input words, convert them to lowercase, remove any punctuation from them, sort them, then calculate the median of your word distribution along with the mode.

The median is defined as the word in the middle of your sorted distribution. If you have an odd number of words, there is only one median. But, if you have an even number of words, you will have two words that define your median.

The mode is the word that occurs more often than any others. In the event of a tie (multiple words with the same largest frequency), you will print all words with the largest frequency.

Input

Your program will take one or more words as input. A word is defined as one or more contiguous characters followed by white space (space, tab, or newline). All words should be converted to lowercase, and any non-alphabetic characters that start, end, or are contained within the word should be removed.

Output

Your median should be output first on a single line as shown below. If you have an odd number of words in your input, a single median word will be output. If you have an even number of words, you will output two words with a single comma between them, and brackets surrounding them. For example, "`My median=[morrow,night]`".

Your mode should be output on a single line following the median as shown below. Include the number of occurrences of the mode in parentheses following. Enclose your answer with brackets like you did for median. In the event of a tie, use a single comma to separate your words & their occurrences. For example, "`My mode=[an (23),the (23)]`". With ties, all words should be listed in alphabetical order for both the median and mode.

Sample Input

```
"Good night, good night!  Parting is such sweet sorrow,
That I shall say good night till it be morrow.
If love be rough with you, be rough with love.  Prick
love for pricking and you beat love down.
These violent delights have violent ends."
```

Sample Output

```
My median=[morrow,night]
My mode=[love (4)]
```

Problem 8

Snow to Water Ratios

Parts of Western North Carolina including Asheville were singing Bing Crosby's "White Christmas" last month as many in the area saw snow on December 25th. Snow can be found all around Asheville throughout January too as the region averages 4.6 inches of snow for the month. On the UNC Asheville campus, you should definitely check out their impressive Atmospheric Science Data Center, one of the nation's largest weather data centers. Here, you will learn that the amount of snow from a storm can look impressive when it covers your house and cars, but if you melted the snow, you would discover that little water is actually involved.

The snow to ice ratio or snow ratio expresses how much volume of snow you get for a given volume of water. Typically, a ratio of 10:1 (ten to one) means that every 10 inches of snowfall equals one inch of liquid water. However, the snow ratio actually depends on the low temperature of the air as the table below indicates. For example, 24 inches of snow at 28 degrees is equal to 24 * (1 inch water/10 inches of snow) = 2.4 inches of water. 28 degrees is closer to 30 degrees than it would be for 25 degrees, and so a ratio of 10:1 is used.

Temp (F)	**30°**	**25°**	**18°**	**12°**	**5°**	**-10°**
Ratio	**10:1**	**15:1**	**20:1**	**30:1**	**40:1**	**50:1**

For this problem, you will write a computer program using weather data from *n* different locations on a particular date to indicate the location where the most water has fallen. If a temperature is equal distance from two different values, use the ratio of the higher temperature. For example, for 15 degrees, use a ratio of 20:1. The sample input below shows four locations on December 3rd, 1971. That date set an all-time record for snowfall in Asheville.

Input

The first line of input will be a positive integer *n* representing the number of locations. This is followed by *n* lines each with three pieces of information in the following order separated by a single space: a floating-point number representing total snowfall, an integer representing the low temperature of the air, and a string for the location in the format: `city, state/province`.

Output

Output a single line in the exact format below indicating the location and amount of water. The inches should be rounded to nearest tenth. Assume only one location will have the most water.

Sample Input

```
4
30.5 18 Calgary, Alberta
25.0 25 Denver, Colorado
16.3 28 Asheville, North Carolina
22.1 16 Syracuse, New York
```

Output Corresponding to Sample Input

```
More water fell in Denver, Colorado (1.7 inches)
```

Problem 9

Where there's a Will, there's Rabbits

There is an old puzzle about a man who raised rabbits, his will, and how a lawyer made everyone happy.

The old man raised rabbits as a hobby. When he died, he left a will, in which he stated that his three grandsons should receive his rabbits, to be divided as follows:

One grandson was to receive ½ of the total number of rabbits, the next grandson was to receive ⅓ of the total, and the last grandson was to receive ⅑ of the rabbits.

The lawyer counted the rabbits and discovered that there were exactly 17 rabbits. Hence, the first grandson should receive 8½ rabbits, the second grandson should get 5⅔ rabbits, and the last grandson should get $1\frac{8}{9}$ rabbits! What could he do? Fortunately, the lawyer, himself, owned a rabbit. He brought it to the reading of the will and added his to the other 17 rabbits. He then proceeded to carry out the terms of the will. He gave the first grandson 9 rabbits, the second one 6 rabbits, and the third one 2 rabbits. He thus gave out 17 rabbits. He took his own rabbit home, and everyone received as the will specified. Your task is to write a program to solve this problem in a more general setting.

Input

Each case will consist of the number of rabbits owned by the man, N, and C the number of people in the will (on one line) and then a list of fractions describing the amounts that each person is to receive (on the next line). Input will be terminated by a case with $N = 0$ and $C = 0$ which is not to be processed. Otherwise, $N, C \geq 1$. You may assume that $N + k$ will fit in a 64-bit integer.

Output

If the sum of the fractions exceeds one, output "Too much!" If the total of the fractions is one, and no rabbit needs to be subdivided, output "Just right." Otherwise, you need to find the smallest number, $k \geq 0$, such that $N + k$ rabbits can be distributed without harm to any rabbit. In this case, output "Bring k rabbits, distribute $N+k$ rabbits, take j rabbits home." If either k or j are 1, then use the singular "rabbit" instead of "rabbits."

Sample Input

```
4 3
1 2 1 3 1 4
8 3
```

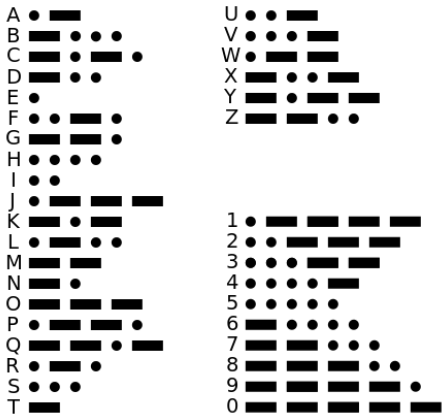
```
1 2 1 4 1 4
11 2
1 3 1 5
17 3
1 2 1 3 1 9
1 2
1 4 1 4
0 0
```

Output Corresponding to Sample Input

```
Too much!
Just right!
Bring 4 rabbits, distribute 15 rabbits, take 7 rabbits home.
Bring 1 rabbit, distribute 18 rabbits, take 1 rabbit home.
Bring 3 rabbits, distribute 4 rabbits, take 2 rabbits home.
```

Problem 10

Morse Code Decoder



Back in 1844, Samuel Morse sent out the first message "What Hath God Wrought?" in Morse code over the newly invented telegraph. In many ways, it was a pioneering form of texting.

Each character (letter or numeral) is represented by a unique sequence of dots and dashes. Letters which are used more frequently like E and T have the shortest encoding. All letters are separated from one another by exactly three spaces, and all words are separated from other words by seven spaces. Your job is to write a program that decodes a line of Morse code.

Input

You may assume only capital letters and digits are used. Each input file starts out with 36 lines representing the encoding of letters 'A'…'Z' followed by digits '0'… '9'. Each letter or digit has three spaces following it in the input file. A positive integer *n*, and then *n* lines of test cases to decode will follow.

Output

For each of the *n* test cases in Morse code, output the word(s) it corresponds.

Sample Input

```
A .-
B -...
C -.-.
 ●
 ●
 ●

X -..-
Y -.--
Z --..
0 -----
1 .---
2 ..---
3 ...--
4 ....-
5 ....
6 -...
7 --...
8 ---..
9 ----.
2
-..   ..   ...   -.   .   -.--
--   ..   -.-.   -.-   .   -.--      --   ---   ..-   ...   .
```

Sample Output

```
DISNEY
MICKEY MOUSE
```