

Dennis MV Programming Contest
October 23, 2009

Sponsored by:
Dennis Matveyev

Rules:

1. There are five questions to be completed in two and a half hours.
2. All questions require you to read the test data from standard input and write results to standard output. You cannot use files for input or output. Additional input and output specifications can be found in the General Information Sheet.
3. The allowed programming languages are C, C++, Java and C#.
4. All programs will be re-compiled prior to testing with the judges' data.
5. Non-standard libraries cannot be used in your solutions. The Standard Template Library (STL) and C++ string libraries are allowed. The standard Java API is available, except for those packages that are deemed dangerous by contestant officials (e.g., that might generate a security violation).
6. Programming style is not considered in this contest. You are free to code in whatever style you prefer. Documentation is not required.
7. All communication with the judges will be handled by the PC2 environment.
8. Judges' decisions are to be considered final. No cheating will be tolerated.

A. Watch Out!!

Mr. Christopher S. K. Eezy is a crazy scientist who loves time travel. In fact, Eezy is not just crazy. He is a complete lunatic. After bumping his head on a washing machine during a bad laundry day, he invented the Flaks Capacitor, a device that makes time travel impossible. At this time his device is not powerful enough and can only cause nearby watches and clocks to go backwards in time. This has created a big problem for S.P.K.Eezy. You see, it's hard to tell the time when your watch is going backwards. Even though Mr. S.Koozy is incredibly smart and handsome, he has trouble with basic math. He needs your help to figure out the actual time when all the clocks are affected by his time device.

S. Koazy does his time experiments exactly at midnight. Thus, for example, if an hour had passed after the activation of the Flaks Capacitor, the backwards watch will be showing "23:00:00", when the actual time is really "1:00:00". When the watch is exactly on the hour, it is easy for S. Kroozy to figure out the real time. But how about telling the time when the backwards watch is showing "12:13:55"? This is where I. M. Kreazy needs your help.

Input

Input will consist of multiple cases. Each line will contain a time string in a format of "[H]H:MM:SS", where valid time will be in a 24 hour format ranging from "0:00:01" to "23:59:59". Note that while minutes and seconds will always have the leading 0 to pad them to two digits, the hours will not. When hours have one digit only, the time string will be right-justified by having a leading space before the time string. The last line will contain the string "0:00:00" to indicate the end of input, and is not to be processed.

Output

For each test case output the actual time that you obtain by interpreting the backwards clock. Output format will be the same as the input format "[H]H:MM:SS", where minutes and seconds may have a leading 0, and hours do not. If hours have a single digit, right-justify the time string by printing out a space before it. If the backwards watch is showing an impossible time, display "I broke time-space continuum!" on a line by itself.

Sample Input

```
23:00:00
12:13:55
 7:42:69
0:00:00
```

Sample Output

```
 1:00:00
11:46:05
I broke time-space continuum!
```

B. The Broken Record of DJ Zhzyatslya

DJ Zhzyatslya is in trouble. You see, after a successful career as a DJ, Zhzyatslya decided to try his hand at computing. Unfortunately that did not work so well. After failing his Final Exam on History of Computing and Computation of All Things Computable using Computers, DJ Zhzyatslya got very upset and smashed up his latest favorite one of a kind record collection. After he came back to sanity, he saw his life in pieces in front of him. Will you be so kind and help him put his records and his life back together?

DJ Zhzyatslya is very fond of his name. To ensure you pronounce it correctly, he provided you with his IPA (International Phonetic Alphabet) transliteration: [z̥z̥ ʲa f̩ l̩ ʲa]. The DJ is too sad to care that he had wasted your time having you read this entire problem statement block when it gave you virtually no information about the actual problem. Will you cheer him up anyways?

Input

Input will consist of multiple test cases. Each line will be in format $A / B + C / D$, where the letters represent numbers between 1 and 1000 inclusively. The last line will have all zeroes in place of the numbers. That line will not be processed.

Output

Sum up the fractions of DJ Zhzyatslya's Record Collection and print them out in a format A / B , if the sum is less than one. Use the format A and B / C , if the fractional sum is larger than one. If the sum turns out to be a whole number, output only the whole part without the fraction, in a format of A . Always reduce the fractions to their lowest terms, when possible. When it's impossible, don't do it! When you are all done with the input, print out just one more line that says "Zhzyatslya, I did this for YOU!" This is your note to DJ Zhzyatslya to let him know that his life is whole again!

Sample Input

```
1 / 1 + 2 / 2
1 / 1 + 2 / 3
1 / 3 + 1 / 3
0 / 0 + 0 / 0
```

Sample Output

```
2
1 and 2 / 3
2 / 3
Zhzyatslya, I did this for YOU!
```

C. Not a Composite Grid

Curiously Strong Engineers consortium (CSE) is in the process of developing new grid-based games to compete with Sudoku. Matt “This Cool” Wolfenschnitzel, lead designer of CSE and head of the Extensively Employed Creative Synergy department (EECS) is a creator of the game that is the best things since Portal.

Matt “This Cool” will give you the size of the grid and a number between 1 and 100, which he puts in the top left corner of the grid. Using numbers 1 through 100, your task is to fill the rest of the grid according to the following rules: each number in the grid must be unique; each consecutive pair of numbers both vertically and horizontally must sum to a non-composite number; and the sum of all the numbers in the grid must be as small as possible.

Note that this way there are many grids possible for any particular grid size. Matt “This Cool” wants you to pick the one that’s first lexicographically. This means that if there are two grids satisfying the rules, pick the one that has the least first square (1st row 1st column). If there is a tie, pick one that has the least second square (1st row 2nd column), and so on. When the first row is over, and there is still a tie, use the second row, and then the third one, and so on until and including the last row.

Matt guarantees that grids with sizes over 4 by 4 will either have an optimal solution, or none at all. By optimal solution, Matt means that only numbers 1 through $g * g$ can be used, where g is the size of the grid.

Input

Input will consist of multiple test cases. Each test case consists of an integer number g ($1 \leq g \leq 10$), indicating the size of g by g grid, following by an integer number n , ($1 \leq n \leq 100$) that will go in the top left corner of the grid to get you started. Fill in the rest of the grid according to the rules above.

Output

For each test case, output a line “Matt, here is puzzle # i :”, where i is the number of the puzzle, starting with 1, followed by the print out of the grid. Each grid consists of g rows and g columns of numbers. Space-justify each number according with the max number in the grid. If the max number has N digits, space-pad the first column to N places and the remaining column to $N+1$ places. There will be no spaces after the last number in each row. For visual clarity, separate each pair of consecutive print-outs by a single blank line. Do not leave any extra blank lines after the last print-out. See Sample Output for an example. If there are no grids possible, output “Yo, Matt! That puzzle # i , it was not cool!” on a line by itself, where i is the puzzle counter. Because ya know .. Matt is a really cool guy, and giving you impossible grids .. it's just not like him at all.

Sample Input

```
6 1
3 1
5 2
10 23
0 0
```

Sample Output

Matt, here is puzzle #1:

```
1 2 3 4 7 6
10 21 16 13 24 5
19 22 25 18 23 14
12 31 36 35 8 15
29 30 17 26 33 28
32 11 20 27 34 9
```

Matt, here is puzzle #2:

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

Yo, Matt! That puzzle #3, it was not cool!

Matt, here is puzzle #4:

```
23 6 1 2 3 4 7 10 9 8
14 5 12 11 20 27 16 13 28 15
17 24 19 18 41 26 21 40 31 22
30 29 42 25 48 35 32 39 58 45
37 60 47 36 53 44 57 50 51 38
34 49 54 43 84 65 74 33 46 63
55 52 85 64 73 66 83 68 81 86
96 97 94 87 76 91 90 89 92 71
67 82 69 62 75 88 61 78 59 80
100 99 98 95 56 93 70 79 72 77
```

D. Dogs with Large Eyes

Hans, a little boy, had many dogs. Some of his dogs had eyes as big as tea saucers. Others had eyes as big as windmills, and some of the more ferocious dogs had eyes as big as skyscraper towers. Recently Hans befriended another boy named Andersen. Andersen had a lot of dogs and each dog had eyes of different sizes. Some dog's eyes were as big as electric scooters, some had eyes as big as Suzuki motorcycles, and some had eyes as large as rotating yellow Ferraris. Hans realize that some of these dogs had eyes of identical sizes. He also realized that the dogs with the same eyes had magical powers when they stood next to each other. Hans just had to know how many magical dogs he and Andersen had together. Since the dogs are wild and ferocious, they are running all over Hans' yard. Can you locate the dogs with identical eye sizes and bring them back to Hans to let the magic happen?

Input

Each input line will begin with a number n . You will be given a list of n integer numbers 1 through 100 inclusively, where the number represents the size of dog's eyes in alens. Alen is a Danish unit of length equaling to 62.77 centimeters. Hans and Andersen together will not have more than 200 dogs. The last line will contain the number 0 by itself, indicating the end of input and should not be processed.

Output

For each line of input, you will find all groups of dogs with the same eye sizes and print out the sizes in sorted order. If you cannot find any dogs with identical eyes, print out the message "I want more dogs with large eyes!" on a line by itself.

Sample Input

```
5 1 2 3 2 4
8 1 2 2 3 3 4 4 5
5 2 1 3 4 5
0
```

Sample Output

```
2
2 3 4
I want more dogs with large eyes!
```

E. Reversible Primes

Your friend, Honorable Sir Filthy Richard Moneybags III, enjoys two things in life – playing the lottery and using prime numbers for fun and profit. He profits from his interests by playing the lottery and only betting on prime numbers. So far, he has been using any regular prime numbers and his luck at winning has not been very good. He has decided to start betting only on reversible prime numbers, hoping that his luck will improve. A reversible prime is a prime number whose 'reverse' is also a prime. To reverse a number, write it down on paper, and then read it backwards. For example, the reverse of 12345 is 54321.

One such reversible prime is 17, since its reversal, 71 is also a prime. Another example is 13, as 31 also shares the prime number property. Note that while 41 is a prime, 14 is not!

F. Moneybags III quickly found out that calculating these numbers takes too long. You see, Rich only uses pencil and paper for calculations, so he doesn't get very far and he quickly gets tired. He has summoned you to generate a list of reversible primes for him within a given range. He expects a fast turn-around time so that he can go back to playing his lottery. We are sorry to say, but Sir Filthy Rich Moneybags he won't be paying you for your efforts. But perhaps he will be generous to you when he wins a jackpot using the numbers you've generated? I wouldn't bet on it ...

Input

Input will consist of multiple test cases. Each test case consists of a pair of integers $[A, B]$, where $1 \leq A \leq 4,000,000$ and $A \leq B \leq 4,000,000$. The last line will contain two zeroes, to indicate the end of input and should not be processed.

Output

For each test case, output the number of reversible primes within the given range. Each case should appear on a single line by itself.

Sample Input

```
1 23
1 101
4 6
0 0
```

Sample Output

```
7
14
1
```