University of Michigan-Dearborn
2009 CIS Programming Contest
February 11, 2009

Sponsored by ACM:
Association of Computing Machinery

Rules:

1. There are five questions to be completed in three 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++, C# and Java.
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. Save the Prince!

Once upon a time in a kingdom, there lived a king. He was prosperous, powerful and he loved hard puzzles. King's puzzles were so hard that the people were afraid to deal with the king, as before he would even talk to them, they would have to solve one of his hard puzzles. It was not until a handsome young prince – you! – showed up at the king's doorstep and asked for the hand of the king's daughter.

"Before I allow you to see my daughter", said the king, "you must complete a challenging task. Your task is as follows: as you know, my kingdom uses the currency called rupees and I have coins of denominations 1, 7, and 12. I have an enormous treasury, but I have a limited number of coins. I would like to keep as many coins in my treasury as I can. When I pay my servants, I want you to figure out the least number of coins necessary to pay them off".

Thus, for example, if king gives you the bill of 14 rupees, there are quite a few ways to pay it off by using the denominations of the coins. For example, some of these ways are:
14 = 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 (14 coins total)
14 = 7 + 1 + 1 + 1 + 1 + 1 + 1 + 1 (8 coins total)
14 = 12 + 1 + 1 (3 coins total)
14 = 7 + 7 (2 coins total)

Only the last option 14 = 7 + 7 gives you the least number of coins in this case. Your task is to figure out that least number of coins to get the hand of king's daughter.

Input
Input will consist of multiple test cases. Each line of input will contain an integer n between 1 and 1,000,000 inclusively, where n is the bill the king has to pay. Last line will start with number 0, and should not be processed.

Output
For each line of input the king expects you to print out the number of the coins required to form the sum as such "The bill of n rupees can be paid by using k coins.", where n is the sum of the bill and k is the minimum number of coins. Use "rupee" or "coin" in your output, if the bill consists of 1 rupee or if it can be paid by one coin respectively. The number of coins must be the least possible, or else the king will throw you in a snake pit with a prime number of snakes in it, or worse – fail you in all of your classes! You know how ruthless kings can be on a bad day. But don't worry, king's daughter is confident enough that you will succeed in your quest and she is waiting for you.

Sample Input
```
14
85
7
0
```

Sample Output
```
The bill of 14 rupees can be paid by using 2 coins.
The bill of 85 rupees can be paid by using 8 coins.
The bill of 7 rupees can be paid by using 1 coin.
```

B. Your Final Sequence

You are taking a brand new course at the Universal Madness school of Damien (UMD). The course is called CIS 666. Consider yourself lucky that you got Mr. Tor Tür as your teacher, for he is the best one in the school.

On your very first lecture, Mr. Tür writes down n integer numbers on the board and tells you to pick two numbers a and b at random from the sequence, while making sure that a does not divide b. Professor tells you that a divides b, only when there is an integer k, so that $a * k = b$. Then, he tells you to replace the number a with $\gcd(a, b)$, where gcd stands for the greatest common divisor of a and b, and then replace the number b with $\mathrm{lcm}(a, b)$, where lcm stands for the least common multiple of a and b. The result will form a new sequence. If that was not evil enough, professor tells you to do this process all over again with each new resulting sequence until you cannot do it anymore. The good thing, he says, is that eventually you will have to stop, and when you stop, the final sequence will not depend on the choices of a and b that you've made at each step. Professor knows the final sequence and he wants you to find it. Knowing professor Tür, you better find it fast, before he decides to make this problem any harder or worse – assign it as homework! Don't worry, he will do it anyway.

As an example professor tells you that $\gcd(6,15) = 3$ and $\mathrm{lcm}(6,15) = 30$. Here, 6 does not divide 15, because there is no such integer k, where $6 * k = 15$.

If you think that professor Tür is truly an evil incarnate, you are right! You can tell by the horns. But don't worry: you can always express your opinion later on the teacher evaluation sheets given out at the end of the semester. Just know that it will not matter as professor Tür already has tenure.

Input

Input will consist of multiple cases. Each line of input will start with an integer n between 1 and 1,000 inclusively, followed by n integer numbers, each between 1 and 30,000 inclusively. The last line will start with number 0, and should not be processed.

Output

For each line of output, print out the final sequence that you will get after applying the process described above. Each sequence should be on a line by itself, where the numbers in the sequence are separated by a blank space. Note that there should be no spaces after the last number in the sequence.

Sample Input

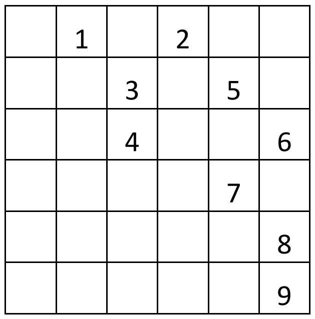
```
7 1 2 3 4 5 6 7
3 24 21 52
0
```

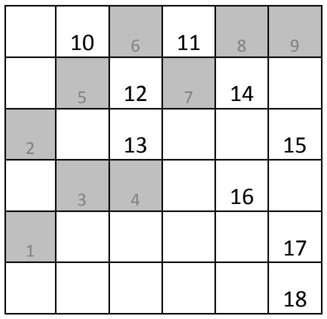
Sample Output

```
1 1 1 1 2 6 420
1 12 2184
```

C. This is Not Your Momma's Cipher

There is a time in every kid's life when they start asking questions about cryptography. When Alice started asking these questions, her mom knew right away that it was the time to pass on her family cipher. Here is what she told Alice: all messages are written into a grid of six by six squares, according to the following rules: first, remove all the spaces from the message and split the message into groups of nine letters. Then, write the first group of letters into the grid as shown below on the left:





Now, turn the grid 90 degrees anti-clockwise and repeat the pattern. The image above on the right shows how to write the second group of letters. Repeat the rotation and the pattern two more times until the entire grid is filled out. If the message is shorter than the grid, use random letters to complete the rest of the grid. If the message is longer, use a second grid. After explaining the family's cryptography secret to Alice, her mom knew that it was just a matter of time before Alice will start asking questions about non-Euclidean Geometry. By then, her mom will be ready with more answers.

For now, Alice is busy deciphering her family's manuscripts using the new secret she has acquired. After doing a few ciphers by hand, she has decided to write a program to help her out. She is not a good programmer yet, since she is still a little kid. Will you help her out?

Input

Input will consist of multiple cases. Each line will contain a cipher string consisting of thirty six lower case letters with no spaces between them. The cipher is transcribed into the string by reading each row of the grid left to right from top to bottom. The last line will have number 0, indicating the end of input.

Output

For each cipher input, print out the decoded message on a line by itself in lower case letters.

Sample Input

```
otuhmxtoetuaetjeedutyegrgolrvenaaids
itdhrxfrieiaduspysohanasnybarnelcome
0
```

Sample Output

```
thejudgesareevilandareouttogetyoutmx
thisisaneasyproblemcanyoufindharderx
```

D. Prime Brothers

Once upon a time there were two brothers, Khishtaki and Saritanur. They loved to play games and they loved prime numbers to bits and pieces. When they were bored they liked to play this game: one brother would think up two prime numbers, multiply them in his head and then tell the result to Saritanur. Saritanur would then tell Khishtaki the original prime numbers used to make up the product.

Hishtaki and Saritanur make this game look too easy, but that's because they are so skilled at it. Would you like to give it a try? "It's easy", the brothers tell you. "If we give you 15, you give us 3 and 5, if we give you 143, you give us 11 and 13. Would you like to try?" "Yes", you say, and the brothers give you this number: 187. "Hmm, you think. This is not so easy anymore", but after a minute of thought you come up with the answer: 11 and 17. Now the brothers give you number 11021. "Holy Primes!", you think, "this is more difficult than I thought. I better write a program to figure this out for me".

Input

Input will consist of multiple cases. Each line will contain an integer number between 6 and 4,294,967,294 inclusively, which will be a product of two prime numbers. The last line will have number 0, indicating the end of input.

Output

For each line of input, output the two prime numbers separated by a space, where the prime numbers form the product given to you by the brothers. The smaller prime number must be first, followed by the larger prime number. If the numbers are the same, their ordering does not matter.

Sample Input

```
15
143
187
11021
0
```

Sample Output

```
3 5
11 13
11 17
103 107
```

E. Ukulele Accounting

The accountant of Ukulele Manufacturing Department (UMD) is in trouble! In the current state of economy, he needs to minimize the payroll cost by figuring out the least amount of wages the company has to pay employees.

UMD has a lot of employees and each employee earns a different dollar amount per ukulele made, depending on the employee's rank in the company. The payroll cost for each employee is computed by multiplying their salary rank by the number of ukuleles they've made.

So, for example, if the customer sends out four orders for 3, 5, 2, 4 ukuleles respectively, and UMD has access to employees with wages of $40, $20, $30, and $20 dollars per ukulele, there are many ways to assign the employees to the orders. One way is to assign the order of 3 ukuleles is to the employee earning $40, and then it will cost UMD $120 to complete the order. However, if you assign the same order to the employee earning $20 dollars per ukulele, it will only cost UMD $60 to complete the order. UMD's accountant has to figure out a way to assign an employee to each order in such a way as to minimize the total wages UMD has to pay out. The trouble comes from the Corporate Headquarters, which has an unbreakable rule. The rule states that the orders must be taken care of in the exact order they were received as to not inconvenience the waiting customers. In fact, this rule is so vital to the company, that whenever headquarters sends out an email with the orders, at the bottom of each email there is a reminder that says "ALL ORDERS MUST BE COMPLETED IN THE ORDER GIVEN IN THIS EMAIL. FAILURE TO COMPLY WITH THIS RULE WILL RESULT IN TERMINATION". Of course, the accountant still has complete control over which employee is assigned to each order.

Since UMD's accountant has spent too much time slacking off and playing Ukulele of Time, a computer game written by UMD students, he now desperately needs your help. Could you help the accountant by writing a program to figure out the minimum payroll?

Input

Input will consist of multiple cases. First line will start with an integer n, indicating the number of orders and salaries to follow. The next row will contain n integers, indicating the number of ukuleles for each order. The row after that will indicate the number of salaries of n employees. There will be multiple cases of orders and employees. The last line will contain 0, which indicates the end of input.

Output

For each case, help the accountant to come up with the minimum payroll required to pay all the employees for all the orders that they complete.

Sample Input

```
4
3 5 2 4
40 20 30 20
4
5 2 4 5
1 1 1 1
0
```

Sample Output

```
350
16
```