

Codeforces Round #102 (Div. 2)

A. Help Vasilisa the Wise 2

time limit per test: 2 seconds

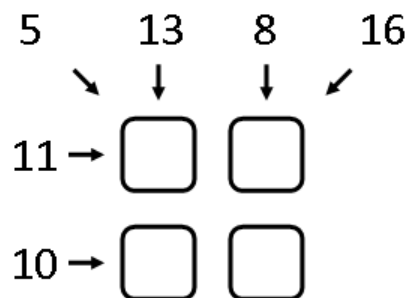
memory limit per test: 256 megabytes

input: standard input

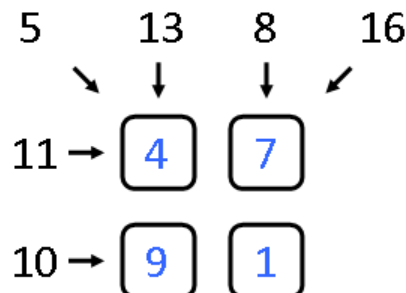
output: standard output

Vasilisa the Wise from the Kingdom of Far Far Away got a magic box with a secret as a present from her friend Hellawisa the Wise from the Kingdom of A Little Closer. However, Vasilisa the Wise does not know what the box's secret is, since she cannot open it again. She hopes that you will help her one more time with that.

The box's lock looks as follows: it contains 4 identical deepenings for gems as a 2×2 square, and some integer numbers are written at the lock's edge near the deepenings. The example of a lock is given on the picture below.



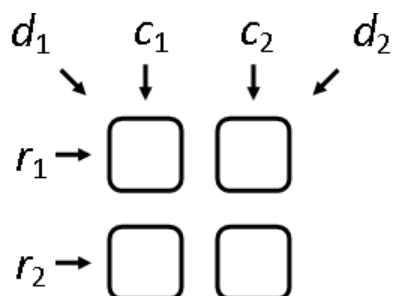
The box is accompanied with 9 gems. Their shapes match the deepenings' shapes and each gem contains one number from 1 to 9 (each number is written on exactly one gem). The box will only open after it is decorated with gems correctly: that is, each deepening in the lock should be filled with exactly one gem. Also, the sums of numbers in the square's rows, columns and two diagonals of the square should match the numbers written at the lock's edge. For example, the above lock will open if we fill the deepenings with gems with numbers as is shown on the picture below.



Now Vasilisa the Wise wants to define, given the numbers on the box's lock, which gems she should put in the deepenings to open the box. Help Vasilisa to solve this challenging task.

Input

The input contains numbers written on the edges of the lock of the box. The first line contains space-separated integers r_1 and r_2 that define the required sums of numbers in the rows of the square. The second line contains space-separated integers c_1 and c_2 that define the required sums of numbers in the columns of the square. The third line contains space-separated integers d_1 and d_2 that define the required sums of numbers on the main and on the side diagonals of the square ($1 \leq r_1, r_2, c_1, c_2, d_1, d_2 \leq 20$). Correspondence between the above 6 variables and places where they are written is shown on the picture below. For more clarifications please look at the second sample test that demonstrates the example given in the problem statement.



Output

Print the scheme of decorating the box with stones: two lines containing two space-separated integers from 1 to 9. The numbers should be pairwise different. If there is no solution for the given lock, then print the single number "-1" (without the quotes).

If there are several solutions, output any.

Sample test(s)

input
3 7 4 6 5 5
output
1 2 3 4

input
11 10 13 8 5 16
output
4 7 9 1

input
1 2 3 4 5 6
output
-1

input
10 10 10 10 10 10
output
-1

Note
Pay attention to the last test from the statement: it is impossible to open the box because for that Vasilisa the Wise would need 4 identical gems containing number "5". However, Vasilisa only has one gem with each number from 1 to 9.

B. Help Kingdom of Far Far Away 2

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

For some time the program of rounding numbers that had been developed by the Codeforces participants during one of the previous rounds, helped the citizens of Far Far Away to convert numbers into a more easily readable format. However, as time went by, the economy of the Far Far Away developed and the scale of operations grew. So the King ordered to found the Bank of Far Far Away and very soon even the rounding didn't help to quickly determine even the order of the numbers involved in operations. Besides, rounding a number to an integer wasn't very convenient as a bank needed to operate with all numbers with accuracy of up to 0.01, and not up to an integer.

The King issued yet another order: to introduce financial format to represent numbers denoting amounts of money. The formal rules of storing a number in the financial format are as follows:

- A number contains the integer part and the fractional part. The two parts are separated with a character "." (decimal point).
- To make digits in the integer part of a number easier to read, they are split into groups of three digits, starting from the least significant ones. The groups are separated with the character ",", (comma). For example, if the integer part of a number equals 12345678, then it will be stored in the financial format as 12,345,678
- In the financial format a number's fractional part should contain exactly two digits. So, if the initial number (the number that is converted into the financial format) contains less than two digits in the fractional part (or contains no digits at all), it is complemented with zeros until its length equals 2. If the fractional part contains more than two digits, the extra digits are simply **discarded** (they are not rounded: see sample tests).
- When a number is stored in the financial format, the minus sign is not written. Instead, if the initial number had the minus sign, the result is written in round brackets.
- Please keep in mind that the bank of Far Far Away operates using an exotic foreign currency — snakes (\$), that's why right before the number in the financial format we should put the sign "\$". If the number should be written in the brackets, then the snake sign should also be inside the brackets.

For example, by the above given rules number 2012 will be stored in the financial format as "\$2,012.00" and number -12345678.9 will be stored as "(\$12,345,678.90)".

The merchants of Far Far Away visited you again and expressed much hope that you supply them with the program that can convert arbitrary numbers to the financial format. Can you help them?

Input

The input contains a number that needs to be converted into financial format. The number's notation length does not exceed 100 characters, including (possible) signs "-" (minus) and "." (decimal point). The number's notation is correct, that is:

- The number's notation only contains characters from the set {"0" - "9", "-", ".", ""}.
- The decimal point (if it is present) is unique and is preceded and followed by a non-zero quantity on decimal digits
- A number cannot start with digit 0, except for a case when its whole integer part equals zero (in this case the integer parts is guaranteed to be a single zero: "0").
- The minus sign (if it is present) is unique and stands in the very beginning of the number's notation
- If a number is identically equal to 0 (that is, if it is written as, for example, "0" or "0.000"), then it is not preceded by the minus sign.
- The input data contains no spaces.
- The number's notation contains at least one decimal digit.

Output

Print the number given in the input in the financial format by the rules described in the problem statement.

Sample test(s)

input
2012
output
\$2,012.00

input
0.000
output
\$0.00

input
-0.00987654321
output
(\$0.00)

input

-12345678.9
output
(\$12,345,678.90)

Note

Pay attention to the second and third sample tests. They show that the sign of a number in the financial format (and consequently, the presence or absence of brackets) is determined solely by the sign of the initial number. It does not depend on the sign of the number you got after translating the number to the financial format.

C. Help Farmer

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

Once upon a time in the Kingdom of Far Far Away lived Sam the Farmer. Sam had a cow named Dawn and he was deeply attached to her. Sam would spend the whole summer stocking hay to feed Dawn in winter. Sam scythed hay and put it into haystack. As Sam was a bright farmer, he tried to make the process of storing hay simpler and more convenient to use. He collected the hay into cubical hay blocks of the same size. Then he stored the blocks in his barn. After a summer spent in hard toil Sam stored $A \cdot B \cdot C$ hay blocks and stored them in a barn as a rectangular parallelepiped A layers high. Each layer had B rows and each row had C blocks.

At the end of the autumn Sam came into the barn to admire one more time the hay he'd been stacking during this hard summer. Unfortunately, Sam was horrified to see that the hay blocks had been carelessly scattered around the barn. The place was a complete mess. As it turned out, thieves had sneaked into the barn. They completely dissembled and took away a layer of blocks from the parallelepiped's front, back, top and sides. As a result, the barn only had a parallelepiped containing $(A - 1) \times (B - 2) \times (C - 2)$ hay blocks. To hide the evidence of the crime, the thieves had dissembled the parallelepiped into single $1 \times 1 \times 1$ blocks and scattered them around the barn. After the theft Sam counted n hay blocks in the barn but he forgot numbers A , B и C .

Given number n , find the minimally possible and maximally possible number of stolen hay blocks.

Input

The only line contains integer n from the problem's statement ($1 \leq n \leq 10^9$).

Output

Print space-separated minimum and maximum number of hay blocks that could have been stolen by the thieves.

Note that the answer to the problem can be large enough, so you must use the 64-bit integer type for calculations. Please, do not use the %lld specifiator to read or write 64-bit integers in C++. It is preferred to use cin, cout streams or the %I64d specifiicator.

Sample test(s)

input
4
output
28 41

input
7
output
47 65

input
12
output
48 105

Note

Let's consider the first sample test. If initially Sam has a parallelepiped consisting of $32 = 2 \times 4 \times 4$ hay blocks in his barn, then after the theft the barn has $4 = (2 - 1) \times (4 - 2) \times (4 - 2)$ hay blocks left. Thus, the thieves could have stolen $32 - 4 = 28$ hay blocks. If Sam initially had a parallelepiped consisting of $45 = 5 \times 3 \times 3$ hay blocks in his barn, then after the theft the barn has $4 = (5 - 1) \times (3 - 2) \times (3 - 2)$ hay blocks left. Thus, the thieves could have stolen $45 - 4 = 41$ hay blocks. No other variants of the blocks' initial arrangement (that leave Sam with exactly 4 blocks after the theft) can permit the thieves to steal less than 28 or more than 41 blocks.

D. Help General

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Once upon a time in the Kingdom of Far Far Away lived Sir Lancelot, the chief Royal General. He was very proud of his men and he liked to invite the King to come and watch drill exercises which demonstrated the fighting techniques and tactics of the squad he was in charge of. But time went by and one day Sir Lancelot had a major argument with the Fairy Godmother (there were rumors that the argument occurred after the general spoke badly of the Godmother's flying techniques. That seemed to hurt the Fairy Godmother very deeply).

As the result of the argument, the Godmother put a rather strange curse upon the general. It sounded all complicated and quite harmless: " *If the squared distance between some two soldiers equals to 5, then those soldiers will conflict with each other!*"

The drill exercises are held on a rectangular $n \times m$ field, split into nm square 1×1 segments for each soldier. Thus, the square of the distance between the soldiers that stand on squares (x_1, y_1) and (x_2, y_2) equals exactly $(x_1 - x_2)^2 + (y_1 - y_2)^2$. Now not all nm squad soldiers can participate in the drill exercises as it was before the Fairy Godmother's curse. Unless, of course, the general wants the soldiers to fight with each other or even worse... For example, if he puts a soldier in the square $(2, 2)$, then he cannot put soldiers in the squares $(1, 4)$, $(3, 4)$, $(4, 1)$ and $(4, 3)$ — each of them will conflict with the soldier in the square $(2, 2)$.

Your task is to help the general. You are given the size of the drill exercise field. You are asked to calculate the maximum number of soldiers that can be simultaneously positioned on this field, so that no two soldiers fall under the Fairy Godmother's curse.

Input

The single line contains space-separated integers n and m ($1 \leq n, m \leq 1000$) that represent the size of the drill exercise field.

Output

Print the desired maximum number of warriors.





Sample test(s)

input
2 4
output
4







input
3 4
output
6

Note

In the first sample test Sir Lancelot can place his 4 soldiers on the 2×4 court as follows (the soldiers' locations are marked with gray circles on the scheme):

In the second sample test he can place 6 soldiers on the 3×4 site in the following manner:

E. Help Caretaker

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Autumn came late to the kingdom of Far Far Away. The harvest was exuberant and it is now time to get ready for the winter. As most people celebrate the Harvest festival, Simon the Caretaker tries to solve a very non-trivial task of how to find place for the agricultural equipment in the warehouse.

He's got problems with some particularly large piece of equipment, which is, of course, turboplows. The problem is that when a turboplow is stored, it takes up not some simply rectangular space. It takes up a T-shaped space like on one of the four pictures below (here character "#" stands for the space occupied by the turboplow and character "." stands for the free space):

```
###      . .#      .#.      #. .  
.#.      ###      .#.      ###  
.#.      . .#      ###      #. .
```

Simon faced a quite natural challenge: placing in the given $n \times m$ cells warehouse the maximum number of turboplows. As one stores the turboplows, he can rotate them in any manner (so that they take up the space like on one of the four pictures above). However, two turboplows cannot "overlap", that is, they cannot share the same cell in the warehouse.

Simon feels that he alone cannot find the optimal way of positioning the plugs in the warehouse that would maximize their quantity. Can you help him?

Input

The only line contains two space-separated integers n and m — the sizes of the warehouse ($1 \leq n, m \leq 9$).

Output

In the first line print the maximum number of turboplows that can be positioned in the warehouse. In each of the next n lines print m characters. Use "." (dot) to mark empty space and use successive capital Latin letters ("A" for the first turboplow, "B" for the second one and so on until you reach the number of turboplows in your scheme) to mark place for the corresponding turboplows considering that they are positioned in the optimal manner in the warehouse. The order in which you number places for the turboplows does not matter. If there are several optimal solutions for a warehouse of the given size, print any of them.

Sample test(s)

input
3 3
output
1 AAA .A. .A.
input
5 6
output
4 A..C.. AAAC.. ABCCCD .B.DDD BBB..D
input
2 2
output
0