Problem ID: coolestskiroute

John loves winter. Every skiing season he goes heli-skiing with his friends. To do so, they rent a helicopter that flies them directly to any mountain in the Alps. From there they follow the picturesque slopes through the untouched snow.

Of course they want to ski on only the best snow, in the best weather they can get. For this they use a combined condition measure and for any given day, they rate all the available slopes.

Can you help them find the most awesome route?

Input

The input consists of:

- one line with two integers n ($2 \le n \le 1000$) and m ($1 \le m \le 5000$), where n is the number of (1-indexed) connecting points between slopes and m is the number of slopes.
- m lines, each with three integers s, t, c ($1 \le s, t \le n$, $1 \le c \le 100$) representing a slope from point s to point t with condition measure c.

Points without incoming slopes are mountain tops with beautiful scenery, points without outgoing slopes are valleys. The helicopter can land on every connecting point, so the friends can start and end their tour at any point they want. All slopes go downhill, so regardless of where they start, they cannot reach the same point again after taking any of the slopes.

Output

Output a single number n that is the maximum sum of condition measures along a path that the friends could take.

Sample visualization

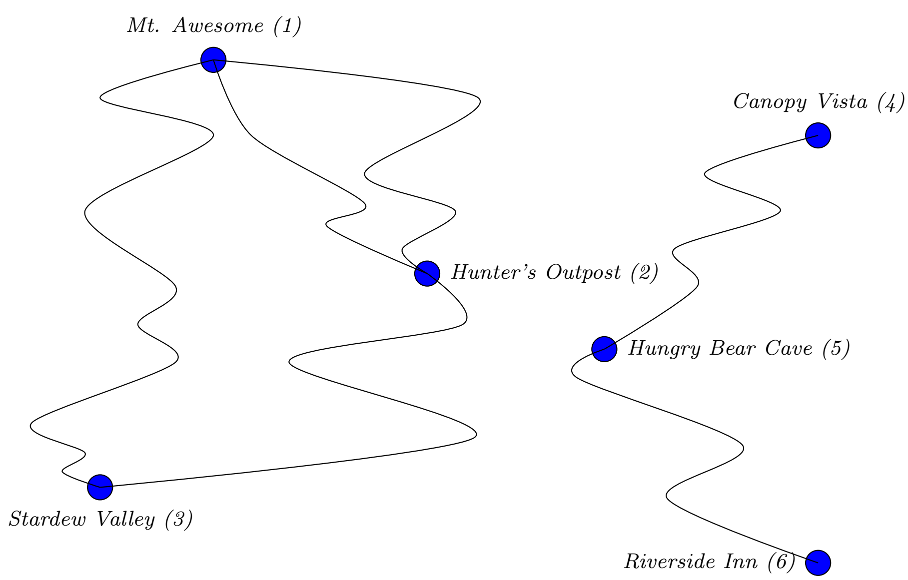


Figure 1: Map of the second sample case

Sample Input 1

```
5 5
1 2 15
2 3 12
1 4 17
4 2 11
5 4 9
```

Sample Output 1

```
40
```

Sample Input 2

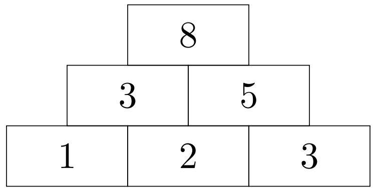
```
6 6
1 2 2
4 5 2
2 3 3
1 3 2
5 6 2
1 2 4
```

Sample Output 2

```
7
```

Problem ID: downthepyramid

Do you like number pyramids? Given a number sequence that represents the base, you are usually supposed to build the rest of the "pyramid" bottom-up: For each pair of adjacent numbers, you would compute their sum and write it down above them. For example, given the base sequence $[1, 2, 3]$, the sequence directly above it would be $[3, 5]$, and the top of the pyramid would be $[8]$:



However, I am not interested in completing the pyramid – instead, I would much rather go underground. Thus, for a sequence of n non-negative integers, I will write down a sequence of $n + 1$ non-negative integers below it such that each number in the original sequence is the sum of the two numbers I put below it. However, there may be several possible sequences or perhaps even none at all satisfying this condition. So, could you please tell me how many sequences there are for me to choose from?

Input

The input consists of:

- one line with the integer n ($1 \leq n \leq 10^6$), the length of the base sequence.
- one line with n integers $a_1, \ldots, a_n$ ($0 \leq a_i \leq 10^8$ for each i), forming the base sequence.

Output

Output a single integer, the number of non-negative integer sequences that would have the input sequence as the next level in a number pyramid.

Sample Input 1

```
6
12 5 7 7 8 4
```

Sample Output 1

```
2
```

Sample Input 2

```
3
10 1000 100
```

Sample Output 2

```
0
```

Problem ID: expiredlicense

Paul is an extremely gifted computer scientist who just completed his master's degree at a prestigious German university. Now he would like to culminate his academic career in a PhD. The problem is that there are so many great universities out there that it is hard for him to pick the best. Because some application deadlines are coming up soon, Paul's only way to procrastinate his decision is by simply applying to all of them.

Most applications require Paul to attach a portrait photo. However, it seems like there does not exist an international standard for the aspect ratio of these kinds of photos. While most European universities ask Paul to send a photograph with aspect ratio 4.5 by 6, some Asian countries discard the applications immediately if the photo does not have an aspect ratio of 7.14 by 11.22, precisely.

As Paul has never been interested in photo editing, he never had a reason to spend a lot of money on proper software. He downloaded a free trial version some months ago, but that version has already expired and now only works with some funny restrictions. The cropping tool, for example, no longer accepts arbitrary numbers for setting the aspect ratio, but only primes. This makes Paul wonder whether the desired aspect ratios can even be properly expressed by two prime numbers. Of course, in case this is possible, he would also like to know the primes he has to enter.

Input

The input consists of:

- one line with an integer n ($1 \leq n \leq 10^5$), the number of applications Paul has to file;
- n lines, each with two real numbers a and b ($0 < a, b < 100$), where $a \times b$ is the desired aspect ratio of one application.

All real numbers are given with at most 5 decimal places after the decimal point.

Output

For each application, if it is possible to represent the desired aspect ratio by two prime numbers p and q, output one line with p and q. Otherwise, output `impossible`. If multiple solutions exist, output the one minimizing $p + q$.

Sample Input 1

```
3
4.5 6
7.14 11.22
0.00002 0.00007
```

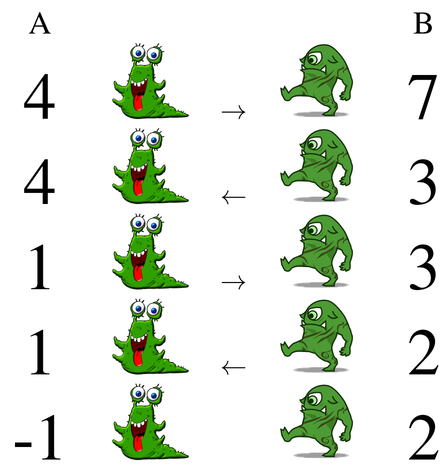
Sample Output 1

```
impossible
7 11
2 7
```

Problem ID: fightingmonsters

Emma just discovered a new card game called *Gwint: A wizard's game*. There are two types of cards: monster cards and spell cards. Monster cards are used to score points, while spell cards typically interact with the monsters in some way.

On each monster card there is an integer value, the *power* of the monster. Monsters can fight each other, and during these fights the power acts as both the strength and the health of the monster. The monsters take turns hitting each other until one of them dies. Whenever a monster A hits a monster B, this causes B to lose an amount of power equal to the power of A. Conversely, if B hits A, A loses power equal to the power of B (see the example below). This continues until one of the two monsters has a power of zero or less, at which point this monster is considered dead.



Images by OpenClipart-Vectors on Pixabay and from PhantomOpenEmoji.

Figure 1: A fight between monsters A and B, starting with powers of 4 and 7, respectively. A hits first. B wins with a remaining power of 2.

One of Emma's most beloved cards in the game is a spell called *Fight!* which states:

> Pick two monsters. They fight each other to the death. If the surviving monster has a power of exactly 1 left, return this card to your hand.

Of course, Emma would like to play as efficiently as possible by picking two monsters such that *Fight!* is returned to her hand. However, there are often a lot of monsters on the board, which makes it very time consuming to figure out whether this can be done or not. Can you help her find two monsters she can pick so that she gets the card back?

Input

The input consists of:

- one line with an integer n ($2 \leq n \leq 10^5$), the number of monsters;
- one line with n integers $m_1, \ldots, m_n$ ($1 \leq m_i \leq 10^6$), giving the power of each monster.

Output

If there is no pair of monsters that Emma can pick, output `impossible`. Otherwise, output two distinct integers i, j ($1 \leq i, j \leq n$), where i is the index of the monster that starts the fight and j is the index of the other monster. If multiple solutions exist, any of them will be accepted.

Sample Input 1

```
4
1 12 67 8
```

Sample Output 1

```
impossible
```

Sample Input 2

```
5
1 1 12 67 8
```

Sample Output 2

```
2 1
```

Sample Input 3

```
6
1 5 6 7 90 8
```

Sample Output 3

```
2 6
```

Problem ID: itstimeforamontage

The heroes of your favorite action TV show are preparing for the final confrontation with the villains. Fundamentally, there are two rivals who will fight each other: a very important main hero who wants to save the universe and an equally important main villain who wants to destroy it. However, through countless recursive spin-offs, they may have slightly less important sidekicks (a hero and a villain who are rivals themselves), who in turn may also have their own (even less important) sidekicks, and so on. Note that there is an equal number of heroes and villains, and each rival pair has at most one sidekick pair.

Initially, every character will fight their rival, with the winner being determined by who has the higher *Power Level*. If a hero and their corresponding villain have the same Power Level, their battle will be determined by their sidekicks' battle, as the winning sidekick can help as a sort of tiebreaker. (If rivals of equal Power Level do not have sidekicks, the hero character will win with the help of random passersby.) However, whenever a battle is won by either side, there is nothing the sidekicks can do about it – this is because the people behind the show believe some fans might get upset if a character were to get defeated by a bunch of less important characters, so they would lose regardless of the Power Levels.

After the battles between rivals (and possible tiebreakers) are done, the most important character remaining will defeat the rest of the opposing side and determine the fate of the universe. Fortunately, the heroes can ensure victory through hard, rigorous training. For each day they spend training, the Power Level of each hero increases by 1, while the villains' Power Levels remain constant.

But you already knew all this. The question plaguing your mind is how long the training is going to take.

Input

The input consists of:

- one line with an integer n ($1 \leq n \leq 1\,000$), giving the number of rival pairs.
- one line with n integers $h_1, \ldots, h_n$ ($1 \leq h_i \leq 1\,000$ for each i), the i-th value giving the Power Level of the i-th most important hero.
- one line with n integers $v_1, \ldots, v_n$ ($1 \leq v_i \leq 1\,000$ for each i), the i-th value giving the Power Level of the i-th most important villain.

Output

Output a single integer, the minimum number of days the heroes need to spend training in order for their side to win.

Sample Input 1

```
4
5 3 1 1
8 6 9 1
```

Sample Output 1

```
4
```

Sample Input 2

```
1
2
1
```

Sample Output 2

```
0
```

Sample Input 3

```
2
4 2
7 5
```

Sample Output 3

```
3
```

Problem ID: logicpuzzle

While browsing a kiosk at a recent trip, you bought a magazine filled with various kinds of logic puzzles. After a while of solving, however, you start to get a bit bored of the puzzles. Still wanting to complete all the puzzles in the magazine, you start wondering about ways to solve some of them algorithmically.

The puzzle you are currently trying to solve is called *Mosaic*, and it is quite similar to the classic *Minesweeper* video game:

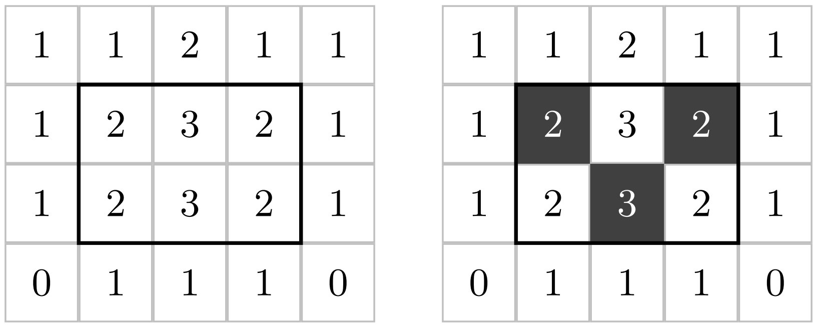


Figure 1: Illustration of the first sample

You are given a two-dimensional grid of cells, initially all white, and you have to color some of the cells in black. You are also given a grid of clue numbers, which extends beyond the borders of the puzzle grid by one cell in each direction. The number in a cell indicates (exactly) how many cells in the 3×3 block centered at this cell need to be colored in black. You may not color any cells outside of the original grid.

Input

The input consists of:

- one line with two integers h, w ($1 \le h, w \le 100$), the height and width of the puzzle;
- $h + 2$ lines, each with $w + 2$ integers $c_1, \ldots, c_{w+2}$ ($0 \le c_i \le 9$), the clue numbers.

Output

If the given clue numbers are inconsistent, output `impossible`. Otherwise, output h lines with w characters each, the solution to the puzzle. Use `X` for black cells and `.` for white cells. If there are multiple solutions, any of them will be accepted.

Sample Input 1

```
2 3
1 1 2 1 1
1 2 3 2 1
1 2 3 2 1
0 1 1 1 0
```

Sample Output 1

```
X.X
.X.
```

Sample Input 2

```
1 2
0 0 1 1
0 1 1 1
0 1 1 1
```

Sample Output 2

```
impossible
```