# Stewards

*"It's only when you are flying above that you realize how incredible the Earth really is."*

*As you know, stewards are needed for every single flight. Today you will be that one steward, and you will be assisting the passengers in finding their seats.*

You will be given a **sequence of 6 seats** - every seat is a **mix** of **a** **number and a letter** in the format **"{number}{letter}"**. You will also be given two more sequences of **numbers only**.

First, you have to take the **first** number of the **first** sequence and the **last** number of the **second** sequence. Next, take the **sum** of those **two numbers** and find its **ASCII character**.

* Compare **each** of the **two taken numbers** and the **found character** with the **seats**. If you find a match, the **passenger is seated**, and the seat is considered taken. **Remove** **both numbers** from their sequences.
* If there is **no equality**, the two numbers should be returned **at the end of their sequences** (first becomes last, last becomes first).
* If you **match an already taken seat**, you should just **remove both numbers** from their sequences.

**Each time** you take numbers from the sequences and try to match them, **you make one rotation**. You should **keep track of all rotations made**.

The **program should** **end** under the following circumstances:

* You have found **3** (three) **seat** **matches**
* You have made a total of **10 rotations**

### Input

* On the **first line,** you will be given a sequence of seats - **strings** separated by comma and space **", "**
* On the **second** and the **third line,** you will be given two more sequences - **integers** separated by a comma and a space **", "**

### Output

When the program ends, print the following on **two** **different lines**:

* **Seat matches: {matches separated by comma and space}**
* **Rotations count: {total rotations made}**

### Constraints

* All integers will be in the range **[1, 100]**
* All letters will be in the range **[A-Z]**
* You will **never** **run out of numbers** in your sequences before the program ends
* You will never have **more than one match** at a time

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 17K, 20B, 3C, 15D, 31Z, 28F  20, 35, 15, 3, 2, 10  1, 15, 64, 53, 45, 46 | Seat matches: 20B, 15D, 3C  Rotations count: 4 |
| **Comment** | |
| 1) Take the first number from the first sequence (20) and the last number from the second sequence (46). Then, sum the two numbers (66) - its ASCII character is "B". Check both combinations "20B" and "46B" with the seats. The seat "20B" matches the first combination, so remove both numbers (20 and 46) from the sequences and take the seat - it's no longer available for matching.  2) Take the next numbers - 35 and 45. Their sum matches the character "P". There are no matches. Return both numbers to the opposite side of the sequences. The sequences look the following way: "15, 3, 2, 10, 35" and "45, 1, 15, 64, 53"  3) Take the following numbers - 15 and 53. Their sum matches the character "D". The seat "15D" is matched. Remove numbers 15 and 53 from their sequences.  4) Take the following numbers - 3 and 64. Their sum matches "C". The seat "3C" is matched. Remove the numbers 3 and 64 from their sequences.  Three seats are matched - print the needed information and end the program. | |
| **Input** | **Output** |
| 25A, 16B, 44T, 49D, 27M, 44F  25, 3, 31, 49, 26, 13  10, 15, 44, 40 | Seat matches: 25A, 44F  Rotations count: 10 |
| **Input** | **Output** |
| 15C, 25C, 36C, 43P, 40E, 38G  15, 25, 80, 40, 15, 99, 52  15, 42, 29 | Seat matches: 25C, 40E, 15C  Rotations count: 7 |