Problem 1

On the first line, you will receive a sequence of **pizza orders**. **Each order contains a different number of pizzas**, separated by comma and space **", "**. On the **second line**, you will receive a sequence of **employees with pizza-making capacities** (how much pizzas an employee could make), separated by comma and space **", "**.

Your task is to check if **all pizza orders are completed**.

To do that, you should take the **first order** and the **last employee** and see:

- If the number of pizzas in the order is **less than or equal to** the employee's pizza making capacity, the order is completed. **Remove both** the order and the employee.
- If the number of pizzas in the order is **greater than** the employee's pizza making capacity, the **remaining pizzas** from the order are going to be made by **the next employees** until the order is **completed**.
    - If there are **no more employees** to finish the order, consider it **not completed**.
- The restaurant **does not take** orders for more than **10 pizzas at once**.
- If an order is **invalid** (less than or equal to 0), you need to **remove it before** it is taken by an employee.

You should keep track of the **total pizzas that are being made**.

Input

- On the **first line** you will be given a sequence of **pizza orders** each represented as a number – **integers** separated by comma and space **", "**
- On the **second line** you will be given a sequence of **employees** with pizza-making capacities – **integers** separated by comma and space **", "**

Output

- If all orders are **successfully** completed, print:
  All orders are successfully completed!
  Total pizzas made: {total count}
  Employees: {left employees joined by ", "}
- Otherwise, if you **ran out** of **employees** and there are still some **orders left** print:
  Not all orders are completed.
  Orders left: {left orders joined by ", "}

Constraints

- You will always have **at least one order** and **at least one employee**
- All integers will be in range **[-100, 100]**

Examples

Input	Output
11, 6, 8, 1 3, 1, 9, 10, 5, 9, 1	All orders are successfully completed! Total pizzas made: 15 Employees: 3, 1
Comment	

1) The restaurant do not take the first order for 11 pizzas.

2) The first employee (1) takes an order for 6 pizzas but could only make 1. 5 pizzas left.

3) The next employee (9) continues the same order for 5 pizzas. The order is completed. Remove both.

4) The next employee (5) takes an order for 8 pizzas but could only make 5. 3 pizzas left.

5) The next employee (10) continues the same order for 3 pizzas. The order is completed. Remove both.

6) The next employee (9) takes an order for 1 pizza. The order is completed. Remove both.

7) All orders are completed.

Input	Output
10, 9, 8, 7, 5 5, 10, 9, 8, 7	Not all orders are completed. Orders left: 2, 5

Comment

1) The last employee (7) takes an order for 10 pizzas but could only make 7. 3 pizzas left.

2) The next employee (8) continues the same order for 3 pizzas. The order is completed. Remove both.

3) The next employee (9) takes an order for 9 pizzas. The order is completed. Remove both.

4) The next employee (10) takes an order for 8 pizzas. The order is completed. Remove both.

5) The next employee (5) takes an order for 7 pizzas but could only make 5. 2 pizzas left.

6) Orders are not completed.

Input	Output
12, -3, 14, 3, 2, 0 10, 15, 4, 6, 3, 1, 22, 1	All orders are successfully completed! Total pizzas made: 5 Employees: 10, 15, 4, 6

Problem 2

Two players bare-handedly throw small sharp-pointed missiles known as darts at a round target known as a dartboard. Who is going to win this game?

You will be given a **matrix with 7 rows and 7 columns** representing the dartboard. For example:

1	2	3	4	5	6	7
24	D	D	D	D	D	8
23	D	T	T	T	D	9
22	D	T	B	T	D	10
21	D	T	T	T	D	11
20	D	D	D	D	D	12
19	18	17	16	15	14	13

Each of the **two players** starts with a **score of 501** and they **take turns** to throw **a dart – one throw for each player**. The score for each turn is **deducted** from the **player's total** score. The **first** player who reduces their **score to zero or less wins** the game.

You are going to receive the information for every throw on a **separate line**. The **coordinate** information of a hit will be in the format: **`"({row}, {column})".`**

- If a player **hits outside the dartboard**, he does **not score** any points.
- If a player **hits a number**, it is **deducted from his total**.
- If a player hits a **`"D"`** the **sum** of the **4 corresponding numbers** per **column** and **row** is **doubled** and **then deducted from his total**.
- If a player hits a **`"T"`** the **sum** of the **4 corresponding numbers** per **column** and **row** is **tripled** and **then deducted from his total**.
- **`"B"`** is the **bullseye**. If a player hits it, he **wins the game,** and the **program ends**.

For example, if Peter hits position with coordinates (2, 1), he wins (23 + 2 + 9 + 18) * 2 = 104 points and they are deducted from his total.

Your job is to find who won the game and with how many turns.

Input

- The **name** of the **first** player and the **name** of the **second** player, separated by **`", "`**
- **7 lines** – the dartboard (**separated by single space**)
- On the next **lines** - the **coordinates** in the format: **`"({row}, {column})"`**

Output

- You should print only one line containing the winner and his count of throws:
  `"{name} won the game with {count_turns} throws!"`

Constrains

- There will always be **exactly 7 lines**
- There will **always** be a winner
- The points will be in range **[1, 24]**
- The coordinates will be in range **[0, 100]**

Examples

Input	Output	Comment
Ivan, Peter 12 21 18 4 20 7 11 9 D D D D D 10 15 D T T T D 3 2 D T B T D 19 17 D T T T D 6 22 D D D D D 14 5 8 23 13 16 1 24 (3, 3)	Ivan won the game with 1 throws!	Ivan hits the Bullseye and wins the game. The program ends.
George, Hristo 17 8 21 6 13 3 24 16 D D D D D 14 7 D T T T D 15 23 D T B T D 2 9 D T T T D 22 19 D D D D D 10 12 18 4 20 5 11 1 (1, 0) (2, 3) (0, 0) (4, 2) (5, 1) (3, 1) (0, 0) (2, 3)	Hristo won the game with 4 throws!	George 1st throw: 501 – 16 = 485 Hristo 1st throw: 501 – 144 = 357 George 2nd throw: 485 – 17 = 468 Hristo 2nd throw: 357 – 168 = 189 George 3rd throw: 468 – 110 = 358 Hristo 3rd throw: 189 – 102 = 87 George 4th throw: 358 – 17 = 341 Hristo 4th throw: 87 – 144 = -57 Hristo wins the game. The program ends.

Problem 3

Create a function named **`flights`** that receives a different number of arguments representing the information **about the flights for a day**:

- the **destination** of **each flight**
- the count of **passengers** that are boarding the plane
- a string **"`Finish`"**

You need to take **each argument** and make a **dictionary** with the plane's **destination as a key** and the **passengers as a value** of the corresponding key.

If there are **more than one** flight to the **same destination**, you should count **all the passengers** that flew to the destination.

You should modify the dictionary until the current argument is equal to "`Finish`".

Note: Submit only the function in the judge system

Input

- There will be **no input**, just parameters passed to your function

Output

- The function should **return the final dictionary**

Constrains

- All numbers will be valid integers in the range **[0, 300]**
- There will be no flight without given number of passengers

Examples

Test Code	Output
`print(flights('Vienna', 256, 'Vienna', 26, 'Morocco', 98, 'Paris', 115, 'Finish', 'Paris', 15))`	`{'Vienna': 282, 'Morocco': 98, 'Paris': 115}`
`print(flights('London', 0, 'New York', 9, 'Aberdeen', 215, 'Sydney', 2, 'New York', 300, 'Nice', 0, 'Finish'))`	`{'London': 0, 'New York': 309, 'Aberdeen': 215, 'Sydney': 2, 'Nice': 0}`
`print(flights('Finish', 'New York', 90, 'Aberdeen', 300, 'Sydney', 0))`	`{}`