# Catapult Attack

**First**, you will be given a **number** equal to the **piles of rocks the Trojans have**. On the **second** line you will be given the **walls of the Spartan fortress**. Then, on each next line (**for each pile**), you receive the power of **each Trojan rock**. Additionally, on every **third wave**, the Spartans build a **new wall** (**extra** line with a single integer) **before** the Trojans toss a rock. In order to enter the city, the Trojans have to **destroy all the walls**.

**Until** there are **no more walls** or **rocks**, the **last Trojan rock** is tossed to the **the first wall**:

* If the **rock’s** value is **greater**, it **destroys** the wall and **lowers** its value by the wall’s value, then attacks the **next** wall, **until** its value reaches 0.
* If the **wall’s** value is **greater**, the rock **is destroyed** and the wall **decreases** its value by the rock’s value.
* If their values are **equal**, **both** the rock and the wall are **destroyed**.

### Input

* **First** line: integer- the number of **piles**
* **Second** line: integers, representing the **walls**, **separated by a single space**.
* For each **pile:** integers, representing the **rocks**, **separated by a single space**.
  + On every **third** wave, you will be given an **extra line** with a **single** integer, which will be the **wall you need to add**. **[!]** Add the wall **before** processing the attacks. **[!]**

### Output

* If there are rocks: “**Rocks left: {rock1}, {rock1}, {rock1},** **(…)**”
* If there are walls: “**Walls left: {wall1}, {wall2}, {wall3}, (…)**”

### Constraints

* All of the given numbers will be valid integers in the range [1, 100].
* **Not all piles** may be needed to destroy the defense.
* There will **always** be a **winning side**, meaning either no warriors or plates left.

### Examples

|  |  |  |
| --- | --- | --- |
| ****Input**** | ****Output**** | ****Comment**** |
| **3**  **10 20 30 40**  **4 5 1**  **10 5 5**  **10 10 10**  **4** | **Walls left: 4** | * First pile (4 5 1):   + Rock (1) attacks **Wall** (10) => destroyed and wall is now 9.   + Rock (5) attacks **Wall** (9) => destroyed and wall is now 4.   + Rock (4) attacks **Wall** (4) => destroyed and wall is gone. * Second pile (10 5 5):   + Rock (5) attacks **Wall** (20) => destroyed and wall is now 15.   + Rock (5) attacks **Wall** (15) => destroyed and wall is now 10.   + Rock (10) attacks **Wall** (10) => destroyed and wall is gone. * Third pile (10 10 10):   + Spartans build a new wall (4), walls are now: 30 4   + Rock (10) attacks **Wall** (30) => destroyed and wall is now 20.   + Rock (10) attacks **Wall** (20) => destroyed and wall is now 10.   + Rock (10) attacks **Wall** (10) => destroyed and wall is destroyed as well. * We have no more rocks and one wall left (4) => see the output. |
|  | **Rocks left: 1,7** | * First pile (3 3 4):   + Rock (4) attacks **Wall** (10) => destroyed and wall is now 6.   + Rock (3) attacks **Wall** (6) => destroyed and wall is now 3.   + Rock (3) attacks **Wall** (3) => destroyed and wall is gone. * Second pile (10 10 10):   + Rock (10) attacks **Wall** (30) => destroyed and wall is now 20.   + Rock (10) attacks **Wall** (20) => destroyed and wall is now 10.   + Rock (10) attacks **Wall** (10) => destroyed and wall is destroyed. * Third pile (5 5):   + Spartans build a new wall (5), walls are now: 10 5   + Rock (5) attacks **Wall** (10) => destroyed and wall is now 5.   + Rock (5) attacks **Wall** (5) => destroyed and wall is also destroyed. * Fourth pile (7 6):   + Warrior (6) attacks **Wall** (5) => the warrior is now 1 and the plate is destroyed. * We have no more walls, so the piles stop coming => see the output. Also, we stop the input. (8 6 7 is not proceeded, but is in the input because the piles are 5) |