# Problem 2. SoftUni Water Supplies

In the SoftUni office we drink lots of water (and other drinks). That’s why instead of using glasses, we use bottles.

Everyone is using different bottles, but with the **same capacity**. You are given an **array of floating-point numbers**, in which **each index represents someone’s bottle**.

We have **some water in stock**, but sometimes it is **not enough to fill our bottles**. That’s why we need you. Your job is to create a program that fills our bottles and **tells us if there is enough water or not**. If there **is enough water** to fill all bottles, you need to **tell us how much water we have left**. If there are **some bottles** that are not **completely filled**, you need to **tell us how many are those bottles**, and **how much more water we need in order to fill them**. We also need to know which members of the team need to wait for more water, and instead of using names, we will use **the indexes of the bottles**.

On the **first line** you are given the **total amount of water** that we have. On the **second line** you are given **all bottles that you need to fill**. There is a tricky part. If the **total water** is an **even number**, you will start filling the bottles from the **beginning** **of the array**, **until you run out of water, or fill all bottles**. If the **number is odd**, you will **traverse** **the** **array** **from the** **end to the beginning**, filling the bottles that way. **In case you run out of water**, you need to **print the indexes** of the empty bottles in the **same order in which you are traversing the array**. On the **last line** you will receive the **maximum capacity** that **each bottle** has.

## Input

* On the **first line** you will receive integer, which represents the **total amount of water** that we have.
* On the **second line** you will receive an **array**, **representing the bottles** that you need to fill.
* On the **last line** you will receive the **capacity for each bottle** of the array.

## Output

* If there is enough water to fill all bottles print the following lines:
  + “Enough water!”
  + “Water left: {amountOfWaterLeft}l.”
* If there isn’t enough water print the following lines:
  + “We need more water!”
  + “Bottles left: {amountOfBottles}”
  + “With indexes: {index}, {index}, {index}”
    - The order of the indexes must be the same, as the order in which you are traversing the array.
  + “We need {amountOfWaterNeeded} more liters!”

## Constraints

* The **total amount of water** will be **integer** in range [0…2,147,483,647].
* The **items in the array** will be **floating-point numbers** in range [0…1.7\*10308].
* The **bottle capacity** will be **integer** in range [0…2,147,483,647].

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 30  2 5 1 3 2 2  5 | Enough water!  Water left: 15l. | The total water is 30 liters. That is an even number, so we start traversing the array from index 0. We fill the bottles and have 15 liters left. |

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 15  0 0 0 0 0  4 | We need more water!  Bottles left: **2**  With indexes: **1**, **0**  We need **5** more liters! | The total water is odd number, so we start traversing the array from its last element (array.lenght - 1).  We are filling the bottles one by one until we run out of water.  That happens on the cell with index=1. We need 1 more liter to fill the bottle thereforе we count it towards the empty bottles.  Now we continue to traverse the array till the end and we count how many bottles we have to fill, and we save the indexes of those bottles in the order we traverse the array (from start to end). |