# Cups and Bottles

You will be given a **sequence of integers** – each indicating a **cup's capacity**. After that you will be given **another sequence of integers** – a **bottle** **with** **water** in it. Your job is to try to **fill up** all of the cups.

Filling is done by picking **exactly one** bottle at a time. You must start picking from **the last received bottle** and start filling from **the first entered cup**. If the current bottle has **N** water, you **give** the **first entered cup N** water and **reduce** its integer value by **N**.

When a cup's **integer value** reaches **0 or less**, it **gets removed**. It is **possible** that the current cup's value is **greater** than the current bottle's value. **In that case** you **pick bottles until** you reduce the cup's integer value to **0 or less**. If a bottle's value is **greater** **or equal to** the cup's **current** value, you fill up the cup and **the remaining water** **becomes wasted**. You should **keep track of the wasted litters of water** and **print it at the end of the program**.

If you **have managed** to **fill up all of the cups**, print the **remaining water bottles**, from the **last entered** **– to the first**, otherwise you must print the **remaining cups**, by **order of entrance** – from the **first entered – to the last**.

### Input

* On the **first line** of input you will receive the integers, representing the **cups' capacity**, **separated** by a **single space**.
* On the **second line** of input you will receive the integers, representing the **filled** **bottles**, **separated** by a **single space**.

### Output

* On the first line of output you must print the remaining bottles, or the remaining cups, depending on the case you are in. Just **keep** the **orders of printing exactly as specified**.
  + **"Bottles: {remainingBottles}"** or **"Cups: {remainingCups}"**
* On the second line print the wasted litters of water in the following format: **"Wasted litters of water: {wastedLittersOfWater}.**

### Constraints

* All of the given numbers will be valid integers in the range [1, 500].
* It is safe to assume that there will be **NO** case in which the water is **exactly as much** as the cups' values, so that at the end there are no cups and no water in the bottles.
* Allowed time/memory: 100ms/16MB.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 4 2 10 5  3 15 15 11 6 | Bottles: 3  Wasted litters of water: 26 | We take the first entered cup and the last entered bottle, as it is described in the condition.  6 – 4 = 2 – we have 2 more so the wasted water becomes 2.  11 – 2 = 9 – again, it is more, so we add it to the previous amount, which is 2 and it becomes 11.  15 – 10 = 5 – wasted water becomes 16.  15 – 5 = 10 – wasted water becomes 26.  We've managed to fill up all of the cups, so we print the remaining bottles and the total amount of wasted water. |
| 1 5 28 1 4  3 18 1 9 30 4 5 | Cups: 4  Wasted litters of water: 35 |  |
| 10 20 30 40 50  20 11 | Cups: 30 40 50  Wasted litters of water: 1 |  |