# 01.Birthday Celebration

**Stack – LIFO – last in, first out – dishes**

**Queue – FIFO – first in, first out store**

*The big day is almost here! Tomorrow is your 18 birthday and you have to plan your party on your own.*

You will be given a **sequence of integers** – each indicating **the eating capacity of a single guest**. After that you will be given **another sequence of integers -** **the** **plates**. Your job is to make sure everyone is full, so you decide to serve them yourself.

Serving is done **exactly one** plate at a time. You must start picking from **the last stacked plate** and start serving from **the first entered guest**.

* If the current plate has **N** grams of food, you **give** the **first entered guest N** grams and **reduce** its integer value by **N**.
* When a guest's **integer value** reaches **0 or less**, it **gets removed**.
* It is **possible** that the current guest's value is **greater** than the current food's value. **In that case** you **pick plates until** you reduce the guest's integer value to **0 or less**.
* If a plate's value is **greater** **or equal to** the guest's **current** value, you fill up the guest and **the remaining food** **becomes wasted**. You should **keep track of the wasted grams of food** and **print it at the end of the program**.
* If you **have managed** to **fill up all of the guests**, print the **remaining prepared plates of food**, from the **last entered** **– to the first**, otherwise you must print the **remaining guests**, by **order of entrance** – from the **first entered – to the last**. Stack

### Input

* On the **first line** of input you will receive the integers, representing the **guests' capacity**, **separated** by a **single space**.
* On the **second line** of input you will receive the integers, representing the **prepared plates of food**, **separated** by a **single space**.

### Output

* On the first line of output you must print the remaining plates, or the remaining guests, depending on the case you are in. Just **keep** the **orders of printing exactly as specified**.
  + "**Plates: {remainingPlates}**" or "**Guests: {remainingGuests}**"
* On the second line print the wasted grams of food in the following format: "**Wasted grams of food: {wastedGramsOfFood}"**

### 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 food is **exactly as much** as the guests' values, so that at the end there are no guests and no food on the plates.
* Allowed time/memory: 100ms/16MB.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 4 2 10 5  3 15 15 11 6 | **Plates: 3**  **Wasted grams of food: 26** | **We take the first entered guest and the last entered plate, as it is described in the condition.**  **6 – 4 = 2 – we have 2 more so the wasted food 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 food becomes 16.**  **15 – 5 = 10 – wasted food becomes 26.**  **We've managed to fill up all of the guests, so we print the remaining plates and the total amount of wasted food.** |
| **1 5 28 1 4**  **3 18 1 9 30 4 5** | **Guests: 4**  **Wasted grams of food: 35** |  |
| **10 20 30 40 50**  **20 11** | **Guests: 30 40 50**  **Wasted grams of food: 1** |  |