# Problem 3 - Messages Manager



Create a program that manages **messages** **sent** and **received** by **users**. You need to keep information about **username**, their **sent** and **received** messages.

On the first line, you will **receive** the **capacity** of **possible** **messages** (total of send and received) **kept** **at once per user**. Next, you will be receiving **lines** with commands until you receive the **"****Statistics"** command. There are **three** **possible** commands:

* **"Add={username}={sent}={received}"**:
  + **Add** the **username** and the number of already **sent** and **received** messages to your **records**.
  + If the person with the given **username** already **exists, ignore** **the line**.
* **"Message={sender}={receiver}"**:
  + If both usernames **exist**, **increase** the **sender's** **sent messages** by **1** and the **receiver's received messages** by **1**.
  + If anyone **reaches** the **capacity** (**first check the sender**), he/she should be **removed** fromthe **record,** and you should **print** the following message:
    - **"****{username} reached the capacity!"**
* **"Empty={username}"**:
  + Delete **all** records of the **given user** **if** he **exists**.
  + If **"All"** is **given as username** - delete **all records** you have.

Finally, **print the total number of users. On the following lines, print each person** with the **sum** of their **sent** and **received** messages.

## Input

* On the **first** **line**, you will **receive** the **capacity** - an **integer** number in the range **[1-10000]**.
* You will be receiving linesuntil you receive the **"Statistics"** command.
* The **initial messages** (**sent** and **received**)will **always** be **below** the **capacity**.
* The input will **always** be **valid**.

## Output

* Print the appropriate message after the **"Message"** command if someone **reaches the capacity**.
* Print the users with their **messages** in the following format:

**"Users count: {count}**

**{username1} - {number of messages}**

**{username2} - {number of messages}**

**…**

**{usernameN} - {number of messages}"**

## Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 10 Add=Berg=9=0  Add=Kevin=0=0  Message=Berg=Kevin  Add=Mark=5=4  Statistics | Berg reached the capacity!  Users count: 2  Kevin - 1  Mark - 9 |
| 20 Add=Mark=3=9  Add=Berry=5=5 Add=Clark=4=0 Empty=Berry Add=Blake=9=3 Add=Michael=3=9 Add=Amy=9=9 Message=Blake=Amy Message=Michael=Amy Statistics | Amy reached the capacity!  Users count: 4  Mark - 12  Clark - 4  Blake - 13  Michael - 13 |
| 12  Add=Bonnie=3=5 Add=Johny=4=4  Empty=All  Add=Bonnie=3=3 Statistics | Users count: 1  Bonnie - 6 |

## JS Examples

The input will be provided as an array of strings.

|  |  |
| --- | --- |
| **Input** | **Output** |
| (["10",  "Add=Berg=9=0",  "Add=Kevin=0=0",  "Message=Berg=Kevin",  "Add=Mark=5=4",  "Statistics"]) | Berg reached the capacity!  Users count: 2  Kevin - 1  Mark - 9 |
| (["20",  "Add=Mark=3=9",  "Add=Berry=5=5",  "Add=Clark=4=0",  "Empty=Berry",  "Add=Blake=9=3",  "Add=Michael=3=9",  "Add=Amy=9=9",  "Message=Blake=Amy",  "Message=Michael=Amy",  "Statistics"]) | Amy reached the capacity!  Users count: 4  Mark - 12  Clark - 4  Blake - 13  Michael - 13 |
| (["12",  "Add=Bonnie=3=5",  "Add=Johny=4=4",  "Empty=All",  "Add=Bonnie=3=3",  "Statistics"]) | Users count: 1  Bonnie - 6 |