Exercise: Objects and Classes

1. Employees

You're tasked to create a list of employees and their personal numbers.

You will receive an array of strings. Each string is an employee **name** and to assign them a personal number you have to find the **length of the name** (whitespace included).

Try to use an object.

At the end print all the list employees in the following format:

`"Name: {employeeName} -- Personal Number: {personalNum}"`

Examples

Input	Output
['Silas Butler', 'Adnaan Buckley', 'Juan Peterson', 'Brendan Villarreal']	Name: Silas Butler -- Personal Number: 12 Name: Adnaan Buckley -- Personal Number: 14 Name: Juan Peterson -- Personal Number: 13 Name: Brendan Villarreal -- Personal Number: 18

2. Towns

You're tasked to create and print **objects** from a text table.

You will receive the input as an **`array`** of strings, where each string represents a table row, with values on the row separated by pipes **" | "** and spaces.

The table will consist of exactly 3 columns **"Town"**, **"Latitude"** and **"Longitude"**. The **`latitude`** and **`longitude`** columns will always contain **valid numbers**. Check the examples to get a better understanding of your task.

The **output** should be **objects**. Latitude and longitude must be parsed to **numbers and formatted to the second decimal point**!

Examples

Input
['Atlanta | 33.753746 | -84.386330', 'Beijing | 39.913818 | 116.363625'];
Output
{ town: 'Atlanta', latitude: '33.76', longitude: '-84.39' } { town: 'Beijing', latitude: '39.91', longitude: '116.36' }

3. Store Provision

You will receive **two arrays**. The first array represents a current **stock** of the local store. The second array will contain **products** which the store has **ordered** for delivery.

The following information applies to both arrays:

Every **even** index will hold the **name** of the **`product`** and on every **odd** index will hold the **`quantity`** of that **`product`**. The second array could contain products that are **already in** the local store. If that happens **increase** the **quantity** for the given product .You should store them into an **`object`**, and print them in the following format: **`(product -> quantity)`**

All of the arrays values will be **strings.**

Examples

Input	Output
['Chips', '5', 'CocaCola', '9', 'Bananas', '14', 'Pasta', '4', 'Beer', '2'], ['Flour', '44', 'Oil', '12', 'Pasta', '7', 'Tomatoes', '70', 'Bananas', '30']	Chips -> 5 CocaCola -> 9 Bananas -> 44 Pasta -> 11 Beer -> 2 Flour -> 44 Oil -> 12 Tomatoes -> 70

4. Movies

Write a function that stores information about movies inside an array. The movies object info must be **`name`, `director`** and **`date`**. You can receive several types of input:

- **`"addMovie {movie name}"`** – add the movie
- **`"{movie name} directedBy {director}"`** – check if the movie **exists** and then add the director
- **`"{movie name} onDate {date}"`** – check if the movie **exists** and then add the date

At the end print all the movies that have **all the info** (if the movie has **no** director, name or date, **don't** print it) in **`JSON`** **format.**

Examples

Input	Output
['addMovie Fast and Furious', 'addMovie Godfather', 'Inception directedBy Christopher Nolan', 'Godfather directedBy Francis Ford Coppola', 'Godfather onDate 29.07.2018',	{"name":"Fast and Furious","date":"30.07.2018","director":"Rob Cohen"} {"name":"Godfather","director":"Francis Ford Coppola","date":"29.07.2018"}

'Fast and Furious onDate 30.07.2018', 'Batman onDate 01.08.2018', 'Fast and Furious directedBy Rob Cohen']	

5. Inventory

Create a function which creates a **register for heroes**, with their **names**, **level**, and **items** (if they have such).

The **input** comes as **array of strings**. Each element holds data for a hero, in the following format:

"**{heroName} / {heroLevel} / {item1}, {item2}, {item3}...**"

You must store the data about every hero. The **name** is a **string**, the **level** is a **number** and the items are all **strings.**

The **output** is all of the data for all the heroes you've stored **sorted ascending by level** and **the items are sorted alphabetically**. The data must be in the following format for each hero:

Hero: {heroName}

level => {heroLevel}

Items => {item1}, {item2}, {item3}

Examples

Input	Output
["Isacc / 25 / Apple, GravityGun", "Derek / 12 / BarrelVest, DestructionSword", "Hes / 1 / Desolator, Sentinel, Antara"]	Hero: Hes level => 1 items => Antara, Desolator, Sentinel Hero: Derek level => 12 items => BarrelVest, DestructionSword Hero: Isacc level => 25 items => Apple, GravityGun