# Lab: Advanced Data Types

## Calorie Object

Write a function that composes an object by given properties. The input comes as an **array of strings**. Every **even index** of the array represents the **name of the food**. Every **odd index** is a **number** that is equal to the **calories in 100 grams of the given product**. Assign each value to its corresponding property, and finally print the object.

The **input** comes as an **array of string** **elements**.

The **output** should be printed on the console.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| ['Yoghurt', '48', 'Rise', '138', 'Apple', '52'] | { Yoghurt: 48, Rise: 138, Apple: 52 } |
| ['Potato', '93', 'Skyr', '63', 'Cucumber', '18', 'Milk', '42'] | { Potato: 93, Skyr: 63, Cucumber: 18, Milk: 42 } |

## Person Info

Write a function that receives **3 parameters**, sets them to an **object**, and **returns** that object.

The input comes as **3 separate strings** in the following order: **firstName**, **lastName**, **age**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Object Properties** |
| "Peter",  "Pan", "20" | firstName: Peter  lastName: Pan  age: 20 |
| "George",  "Smith", "18" | firstName: George  lastName: Smith  age: 18 |

## Inventory

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

The **input** comes as an **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**, a **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**. 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 => Desolator, Sentinel, Antara  Hero: Derek  level => 12  items => BarrelVest, DestructionSword  Hero: Isacc  level => 25  items => Apple, GravityGun |
| [  'Batman / 2 / Banana, Gun',  'Superman / 18 / Sword',  'Poppy / 28 / Sentinel, Antara'  ] | Hero: Batman  level => 2  items => Banana, Gun  Hero: Superman  level => 18  items => Sword  Hero: Poppy  level => 28  items => Sentinel, Antara |

## 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** |
| ['Sofia | 42.696552 | 23.32601',  'Beijing | 39.913818 | 116.363625'] |
| **Output** |
| { town: 'Sofia', latitude: '42.70', longitude: '23.33' }  { town: 'Beijing', latitude: '39.91', longitude: '116.36' } |

|  |
| --- |
| **Input** |
| ['Plovdiv | 136.45 | 812.575'] |
| **Output** |
| { town: 'Plovdiv', latitude: '136.45', longitude: '812.58' } |

## Town Population

You have been tasked to create a registry for different **towns** and their **population**.

### Input

The **input** comes as array of strings. Each element will contain data for a town and its population in the following format: "{townName} <-> {townPopulation}"

If you receive the same town twice, **you should add** the **given population** to the **current one**.

### Output

As **output**, you must print all the towns and their population.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| ['Sofia <-> 1200000',  'Montana <-> 20000',  'New York <-> 10000000',  'Washington <-> 2345000',  'Las Vegas <-> 1000000'] | Sofia : 1200000  Montana : 20000  New York : 10000000  Washington : 2345000  Las Vegas : 1000000 |
| ['Istanbul <-> 100000',  'Honk Kong <-> 2100004',  'Jerusalem <-> 2352344',  'Mexico City <-> 23401925',  'Istanbul <-> 1000'] | Istanbul : 101000  Honk Kong : 2100004  Jerusalem : 2352344  Mexico City : 23401925 |

## Lowest Prices in Cities

You will be given several towns, with products and their price. You need to find **the lowest price** for **every product** and **the town it is sold at** for that price.

### Input

The **input** comes as an array of strings. Each element will hold data about a **town**, **product**, and **its price** at that town. The **town** and **product** will be **strings**, the **price** will be a **number**. The input will come in the following format:

{townName} | {productName} | {productPrice}

### Output

As **output**, you must print **each** **product** with its **lowest price** and **the town** at which the product is **sold at that** **price**. If **two towns share** the **same lowest price**, print the one that was **entered first**.   
The output, for every product, should be in the following format:

{productName} -> {productLowestPrice} ({townName})

The **order of output** in - **order of entrance**. See the examples for more info.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| ['Sample Town | Sample Product | 1000',  'Sample Town | Orange | 2',  'Sample Town | Peach | 1',  'Sofia | Orange | 3',  'Sofia | Peach | 2',  'New York | Sample Product | 1000.1',  'New York | Burger | 10'] | Sample Product -> 1000 (Sample Town)  Orange -> 2 (Sample Town)  Peach -> 1 (Sample Town)  Burger -> 10 (New York) |