# Lab: Associative Arrays

Problems for exercise and homework for the ["JS Fundamentals" Course @ SoftUni.](https://softuni.bg/trainings/3449/programming-fundamentals-with-javascript-september-2021)   
Submit your solutions in the SoftUni judge system at: <https://judge.softuni.org/Contests/1231>

## Phone Book

Write a function that stores information about a **person’s name** and **phone number**. The input comes as an **array of strings**. Each string contains the name and the number. If you receive the same name **twice** just **replace** the number. In the end, print the result **without sorting it**. Try using an **associative array.**

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| **['Tim 0834212554',**  **'Peter 0877547887',**  **'Bill 0896543112',**  **'Tim 0876566344']** | **Tim -> 0876566344**  **Peter -> 0877547887**  **Bill -> 0896543112** |
| **['George 0552554',**  **'Peter 087587',**  **'George 0453112',**  **'Bill 0845344']** | **George -> 0453112**  **Peter -> 087587**  **Bill -> 0845344** |

## Meetings

Write a function that manages meeting appointments. The input comes as an **array of strings**. Each string contains a **weekday** and person’s **name**. For each **successful** meeting, **print a message**. If you receive the **same weekday** twice, the meeting cannot be scheduled so print a **conflicting message**. In the end, print a list of all **successful** meetings. See example for message format and details.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| **['Monday Peter',**  **'Wednesday Bill',**  **'Monday Tim',**  **'Friday Tim']** | **Scheduled for Monday**  **Scheduled for Wednesday**  **Conflict on Monday!**  **Scheduled for Friday**  **Monday -> Peter**  **Wednesday -> Bill**  **Friday -> Tim** |
| **['Friday Bob',**  **'Saturday Ted',**  **'Monday Bill',**  **'Monday John',**  **'Wednesday George']** | **Scheduled for Friday**  **Scheduled for Saturday**  **Scheduled for Monday**  **Conflict on Monday!**  **Scheduled for Wednesday**  **Friday -> Bob**  **Saturday -> Ted**  **Monday -> Bill**  **Wednesday -> George** |

## Address Book

Write a function that stores information about a person’s **name** and his **address**. The input comes as an **array of strings**. Each string contains the **name** and the **address** separated by a **colon**. If you receive the same name **twice** just **replace** the address. In the end, print the full list, **sorted alphabetically** by the person’s name.

|  |  |
| --- | --- |
| **Input** | **Output** |
| **['Tim:Doe Crossing',**  **'Bill:Nelson Place',**  **'Peter:Carlyle Ave',**  **'Bill:Ornery Rd']** | **Bill -> Ornery Rd**  **Peter -> Carlyle Ave**  **Tim -> Doe Crossing** |
| **['Bob:Huxley Rd',**  **'John:Milwaukee Crossing',**  **'Peter:Fordem Ave',**  **'Bob:Redwing Ave',**  **'George:Mesta Crossing',**  **'Ted:Gateway Way',**  **'Bill:Gateway Way',**  **'John:Grover Rd',**  **'Peter:Huxley Rd',**  **'Jeff:Gateway Way',**  **'Jeff:Huxley Rd']** | **Bill -> Gateway Way**  **Bob -> Redwing Ave**  **George -> Mesta Crossing**  **Jeff -> Huxley Rd**  **John -> Grover Rd**  **Peter -> Huxley Rd**  **Ted -> Gateway Way** |

## Storage

Write a function that takes a certain number of **items** and their **quantity**. If the same item appears **more than once**, **add the new amount** to the **existing one**. In the end, print all the items and their amount without sorting them. The input comes as an **array of strings**. Try using a Map().

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| **['tomatoes 10',**  **'coffee 5',**  **'olives 100',**  **'coffee 40']** | **tomatoes -> 10**  **coffee -> 45**  **olives -> 100** |
| **['apple 50',**  **'apple 61',**  **'coffee 115',**  **'coffee 40']** | **apple -> 111**  **coffee -> 155** |

### Hints

Create the solve() function and create a new Map():



Loop through the array, split into tokens, and create variables for each one:



* This time for the quantity we need a number because if we see the same product again, we must add the new quantity

Now let us make the checks for the keys on the map:



* First, we check if the map does ***NOT*** have the product we are currently at and **if so**, we **set it to the given quantity**
* Otherwise, we get the **existing quantity**, we **add the new quantity,** and **set** the product’s quantity **to the new** one

Now we just have to print the result:



* Each key-value pair is and an **array of 2 elements** (the **key** and the **value**), so we use a **for-of** loop and print the key and the value

## School Grades

Write a function that stores **students** and their **grades** throughout the year. If a student appears more than once, **add** the new **grades** to **existing ones**. Finally, **print** the students and their **average grades**, sorted **alphabetically** by **student name.** The input comes as an **array of strings**.

**Note:** The **average grades** must be fixed to the second decimal place.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| **['Lilly 4 6 6 5',**  **'Tim 5 6',**  **'Tammy 2 4 3',**  **'Tim 6 6']** | **Lilly: 5.25**  **Tammy: 3.00**  **Tim: 5.75** |
| **['Steven 3 5 6 4',**  **'George 4 6',**  **'Tammy 2 5 3',**  **'Steven 6 3']** | **George: 5.00**  **Steven: 4.50**  **Tammy: 3.33** |

## Word Occurrences

Write a function that **counts** the times each **word occurs** in a text. Print the words **sorted by count** in **descending** order. The input comes as an **array of strings**.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| **["Here", "is", "the", "first", "sentence", "Here", "is", "another", "sentence", "And", "finally", "the", "third", "sentence"]** | **sentence -> 3 times**  **Here -> 2 times**  **is -> 2 times**  **the -> 2 times**  **first -> 1 times**  **another -> 1 times**  **And -> 1 times**  **finally -> 1 times**  **third -> 1 times** |
| **["dog", "bye", "city", "dog", "dad", "boys", "ginger"]** | **dog -> 2 times**  **bye -> 1 times**  **city -> 1 times**  **dad -> 1 times**  **boys -> 1 times**  **ginger -> 1 times** |

### Hint

* Create a map
* Loop through the elements of the array of words
* Update the map
* Sort the map by value in descending:



* Finally, print the result in the format as the example above