Lab: Objects

1. Towns to JSON

You're tasked to create and print a JSON from a text table. You will receive input as an array of strings, where each string represents a table row, with values on the row encompassed by pipes **"|"** and optionally 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.

Input

The **input** comes as an array of strings – the first string contains the table's headings while the remaining strings contains the data of the table.

Output

- The **output** should be an array of objects wrapped in **JSON.stringify()**.
- **Latitude** and **longitude** must be parsed to **numbers and formatted to the second decimal point.**

Examples

Input	Output

`['| Town | Latitude | Longitude |',`
`'| Melbourne |-37.840935 | 144.946457|',`
`'| Beijing | 39.913818 | 116.363625 |']`
	`[{"Town":"Melbourne",`
	`"Latitude":-37.84,`
	`"Longitude":144.95`
	`},`
	`{"Town":"Beijing",`
	`"Latitude":39.91,`
	`"Longitude":116.36`
	`}]`

`['| Town | Latitude | Longitude |',`
`'| Sydney | -33.865143 | 151.209900 |',`
`'| Perth | -31.953512 | 115.857048 |']`
	`[{"Town":"Sydney",`
	`"Latitude":-33.87,`
	`"Longitude":151.21`
	`},`
	`{"Town":"Perth",`
	`"Latitude":-31.95,`
	`"Longitude":115.86`
	`}]`

What to submit?

Function Signature: function main(input)

2. Score to HTML

You will be given a JSON that represents an array of objects. Parse the JSON and create an HTML table using the supplied objects. The table should have 2 columns **"name"** and **"score"** and each object in the array will also have these keys.

Any text elements must also be **escaped** in order to ensure no dangerous code can be passed.

Input

The **input** comes as array with a single string argument (the array of objects as a JSON).

Output

The **output** should be printed on the console - a table with 2 columns - **"name"** and **"score"**, containing the values from the objects as rows.

Input	Output

`['[{"name":"Peter","score":479},`
`   {"name":"George","score":205}]']`
	`<table>`
	`  <tr><th>name</th><th>score</th></tr>`
	`  <tr><td>Peter</td><td>479</td></tr>`
	`  <tr><td>George</td><td>205</td></tr>`
	`</table>`

`['[{"name":"Peter & Kiro",`
`     "score":479`
`   },`
`   {"name":"George, Maria & Viki",`
`     "score":205`
`   }]']`
	`<table>`
	`  <tr><th>name</th><th>score</th></tr>`
	`  <tr><td>Peter &amp;`
	`Kiro</td><td>479</td></tr>`
	`  <tr><td>George, Maria &amp;`
	`Viki</td><td>205</td></tr>`
	`</table>`

What to submit?

Function Signature: function main(input)

3. From JSON to HTML Table

You're tasked to create an HTML table of students and their scores. You will receive a single string representing an **array of objects**, the **table's headings** should be equal to the **objects' keys**, while **each object's values** should be a **new entry** in the table. Any **text values** in an object should be **escaped**, in order to avoid introducing dangerous code into the HTML.

Input

The **input** comes as array with a **single string argument** (the array of objects).

Output

The **output** should be printed on the console – for each **entry row** in the input print the **object representing it**.

Note:

Object's **keys** will always be the **same.**

HTML

You are provided with an HTML file to test your table in the browser.

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>FromJSONToHTMLTable</title>
    <style>
        table,th{
            border: groove;
            border-collapse: collapse;
        }
        td{
            border: 1px solid black;
        }
        td,th{
            padding: 5px;
        }
    </style>
</head>
<body>
    <div id="wrapper">
    </div>
    <script>
        function fromJSONToHTMLTable(input){
            //Write your code here
        }
        window.onload = function(){
            let container = document.getElementById('wrapper');
            container.innerHTML = fromJSONToHTMLTable(['[{"Name":"Tomatoes &
Chips","Price":2.35},{"Name":"J&B Chocolate","Price":0.96}]']);
        };
    </script>
</body>
</html>
```

Examples

Input	Output

`['[{"Name":"Tomatoes & Chips","Price":2.35},{"Name":"J&B Chocolate","Price":0.96}]']`	`<table>`
	`   <tr><th>Name</th><th>Price</th></tr>`
	`   <tr><td>Tomatoes &amp; Chips</td><td>2.35</td></tr>`
	`   <tr><td>J&amp;B Chocolate</td><td>0.96</td></tr>`
	`</table>`

`['[{"Name":"Peter <div>-a","Age":20,"City":"Sydney"},{"Name":"George","Age":18,"City":"Perth"},{"Name":"Angel","Age":18,"City":"Melbourne"}]']`	`<table>`
	`   <tr><th>Name</th><th>Age</th><th>City</th></tr>`
	`   <tr><td>Peter &lt;div&gt;-a</td><td>20</td><td>Sydney</td></tr>`
	`   <tr><td>George</td><td>18</td><td>Perth</td></tr>`
	`   <tr><td>Angel</td><td>18</td><td>Melbourne</td></tr>`
	`</table>`

What to submit?

Submit only the function and rename it to **main.**

Function Signature: function main(input)

4. Sum by Town

You're tasked with calculating the total sum of income for a number of Towns. You will receive an array of strings representing towns and their incomes. Every **even** index will be a **town** and every **odd** index will be an **income** belonging to that town. Create an object that will hold all the **towns as keys** and their **total income** (the sum of their incomes) **as values** to those keys and print it as a JSON.

Input

The **input** comes as an array of strings - each even index is the name of a town and each odd index is an income belonging to that town.

Output

The **output** should be printed on the console - JSON representation of the object containing all towns and their total incomes.

Examples

Input	Output
['Sydney', '20', 'Melbourne', '3', 'Sydney', '5', 'Melbourne', '4']	{"Sydney":25 ,"Melbourne" :7}
['Sydney', '20', 'Melbourne', '3', 'sydney', '5', 'melbourne', '4']	{"Sydney":20 ,"Melbourne" :3,"sydney": 5,"melbourne ":4}

What to submit?

Function Signature: function main(input)

5. Count Words in a Text

You are tasked to count the number of words in a text using an object as an associative array. Any combination of letters, digits and _ (underscore) should be counted as a word. The words should be stored in the object as properties - the **key** being the **word** and the **value** being the **amount of times the word is contained in the text**.

Input

The **input** comes as an array of strings containing one entry - the text whose words should be counted. The text may consist of more than one sentence.

Output

The **output** should be printed on the console - the JSON representation of the object containing the words.

Examples

Input	Output

`['Far too slow, you're far too slow.']`	`{"Far":1,`
	` "too":2,`
	` "slow":2,`
	` "you":1,`
	` "re":1,`
	` "far":1}`

`['JS devs use Node.js for server-side JS.-- JS for devs']`	`{"JS":3,`
	` "devs":2,`
	` "use":1,`
	` "Node":1,`
	` "js":1,`
	` "for":2,`
	` "server":1,`
	` "side":1}`

What to submit?

Function Signature: function main(input)

6. Populations in Towns

You have been tasked to create a register 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
[Sydney<-> 1200000', 'Montana <-> 20000', 'New York <-> 10000000', 'Washington <-> 2345000', 'Las Vegas <-> 1000000']	Sydney: 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

What to submit?

Function Signature: function main(input)

7. City Markets

You have been tasked to follow the sales of products in the different towns. For every town you need to keep track of all the products sold, and for every product, the amount of total income.

The **town** and **product** are both **strings**. The **amount of sales** and **price for one unit** will be **numbers**. Store all towns together with its products and the **total income** of each products. The total income is calculated with the following formula - **amount of sales * price for one unit**. If you receive as input a town you already have, you should just **add** the **new product** to it.

Input

The **input** comes as array of strings. Each element will represent data about a product and its sales. The format of input is:

`{town} -> {product} -> {amountOfSales} : {priceForOneUnit}`

Output

As **output,** you must print every town, its products and the total income of each product in the following format:

```
“Town – {townName}
 $$${product1Name} : {productTotalIncome}
 $$${product2Name} : {productTotalIncome}
 ...”
```

The **order of output** should be in **order of appearance**.

Examples

Input	Output
['Sydney '-> Laptops HP -> 200 : 2000', 'Sydney '-> Raspberry -> 200000 : 1500', 'Sydney'-> Audi Q7 -> 200 : 100000', 'Montana -> Portokals -> 200000 : 1', 'Montana -> Qgodas -> 20000 : 0.2', 'Montana -> Chereshas -> 1000 : 0.3']	Town - Sydney $$$Laptops HP : 400000 $$$Raspberry : 300000000 $$$Audi Q7 : 20000000 Town - Montana $$$Portokals : 200000 $$$Qgodas : 4000 $$$Chereshas : 300

What to submit?

Function Signature: function main(input)

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

`{townName} | {productName} | {productPrice}`

If you receive the same **town** and **product more than once,** you should **update** the **old value** with the **new one**.

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** should be in **order of appearance**. See the examples for more info.

Examples

Input	Output
['Sample Town | Sample Product | 1000', 'Sample Town | Orange | 2', 'Sample Town | Peach | 1', 'Sydney | Orange | 3', 'Sydney | 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)

What to submit?

Function Signature: function main(input)

9. Extract Unique Words

Write a JS function that **extracts** all **UNIQUE** words from a **valid text** and **stores them**. Ensure that there are **NO duplicates** in the stored words. Once you find a word, there is no need for you to store it again if you meet it again in the text. You also need to make all characters from the words you've stored into **lowercase**.

The **input** comes as array of strings. Each element will represent a sentence.

The **output** is all of the unique words you've found, each with each, **separated** by a **coma and a space**, printed in order of appearance.

Examples

Input	Output
['Lorem ipsum dolor sit amet, consectetur adipiscing elit.', 'Pellentesque quis hendrerit dui.', 'Quisque fringilla est urna, vitae efficitur urna vestibulum fringilla.', 'Vestibulum dolor diam, dignissim quis varius non, fermentum non felis.', 'Vestibulum ultrices ex massa, sit amet faucibus nunc aliquam ut.', 'Morbi in ipsum varius, pharetra diam vel, mattis arcu.', 'Integer ac turpis commodo, varius nulla sed, elementum lectus.', 'Vivamus turpis dui, malesuada ac turpis dapibus, congue egestas metus.']	lorem, ipsum, dolor, sit, amet, consectetur, adipiscing, elit, pellentesque, quis, hendrerit, dui, quisque, fringilla, est, urna, vitae, efficitur, vestibulum, diam, dignissim, varius, non, fermentum, felis, ultrices, ex, massa, faucibus, nunc, aliquam, ut, morbi, in, pharetra, vel, mattis, arcu, integer, ac, turpis, commodo, nulla, sed, elementum, lectus, vivamus, malesuada, dapibus, congue, egestas, metus
['Interdum et malesuada fames ac ante ipsum primis in faucibus.', 'Vestibulum volutpat lacinia blandit.', 'Pellentesque dignissim odio in hendrerit lacinia.', 'Vivamus placerat porttitor purus nec hendrerit.', 'Aliquam erat volutpat. Donec ac augue ligula.', 'Praesent venenatis sapien vitae libero ornare, nec pulvinar velit finibus.', 'Proin dui neque, rutrum vel dolor ut, placerat blandit sapien.', 'Pellentesque at est arcu.', 'Nullam eget orci laoreet, feugiat nisi vitae, egestas libero.', 'Pellentesque pulvinar aliquet felis.', 'Interdum et malesuada fames ac ante ipsum primis in faucibus.', 'Etiam sit amet nisl ex.', 'Sed lacinia pretium metus quis fermentum.', 'Praesent a ante suscipit, efficitur risus cursus, scelerisque risus.']	interdum, et, malesuada, fames, ac, ante, ipsum, primis, in, faucibus, vestibulum, volutpat, lacinia, blandit, pellentesque, dignissim, odio, hendrerit, vivamus, placerat, porttitor, purus, nec, aliquam, erat, donec, augue, ligula, praesent, venenatis, sapien, vitae, libero, ornare, pulvinar, velit, finibus, proin, dui, neque, rutrum, vel, dolor, ut, at, est, arcu, nullam, eget, orci, laoreet, feugiat, nisi, egestas, aliquet, felis, etiam, sit, amet, nisl, ex, sed, pretium, metus, quis, fermentum, a, suscipit, efficitur, risus, cursus, scelerisque

What to submit?

Function Signature: function main(input)