Exercises: This

1. Company

```
class Company {
    // TODO: implement this class...
}
```

Your Task

Write a Company Class, Which Supports the Described Functionality Below.

Functionality

Constructor()

Should have this **1** property:

- **departments** - empty array

AddEmployee({username}, {Salary}, {Position}, {Department})

This function should add a new employee to the department with the given name.

- If one of the passed parameters is empty string (""), undefined or null, this function should throw an error with the following message:

  "Invalid input!"

- If salary is less than 0, this function should throw an error with the following message:

  " Invalid input!"

- If the new employee is hired successfully, you should add him into the departments array and return the following message:

  " New employee is hired. Name: {name}. Position: {position}"

BestDepartment()

This **function** should print the department with the highest average salary and its employees sorted by their salary by descending and by name in the following format:

" Best Department is: {best department's name}
Average salary: {best department's average salary}
{employee1} {salary} {position}
{employee2} {salary} {position}
{employee3} {salary} {position}
. . ."

Submission

Submit only your **Company class.**

Examples

This is an example how the code is **intended to be used**:

© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.

Sample code usage

`let c = new Company();`
`c.addEmployee("Stanimir", 2000, "engineer", "Construction");`
`c.addEmployee("Pesho", 1500, "electrical engineer", "Construction");`
`c.addEmployee("Slavi", 500, "dyer", "Construction");`
`c.addEmployee("Stan", 2000, "architect", "Construction");`
`c.addEmployee("Stanimir", 1200, "digital marketing manager", "Marketing");`
`c.addEmployee("Pesho", 1000, "graphical designer", "Marketing");`
`c.addEmployee("Gosho", 1350, "HR", "Human resources");`
`console.log(c.bestDepartment());`

Corresponding output
Best Department is: Construction **Average salary: 1500.00** **Stan 2000 architect** **Stanimir 2000 engineer** **Pesho 1500 electrical engineer** **Slavi 500 dyer**

2. Fibonacci

Write a JS function that when called, returns the next Fibonacci number, starting at 0, 1. Use a **closure** to keep the current number.

Input

There will be no input.

Output

The **output** must be a Fibonacci number and must be **returned** from the function.

Examples

Sample exectuion

`let fib = getFibonator();`
`console.log(fib()); // 1`
`console.log(fib()); // 1`
`console.log(fib()); // 2`
`console.log(fib()); // 3`
`console.log(fib()); // 5`
`console.log(fib()); // 8`
`console.log(fib()); // 13`

© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.

3. HEX

```
class Hex {
    // TODO: implement this class...
}
```

Your Task

Write a Hex Class, Which Supports the Described Functionality Below.

Functionality

Constructor({value})

Should have this **1** property:

- **value** - number

ValueOf()

This Function Should Return the Value Property of the Hex Class.

ToString()

This **function** will show its hexidecimal value starting with "0x"

Plus({number})

This function should add a number or Hex object and return a new Hex object.

Minus({number})

This function should subtract a number or Hex object and return a new Hex object.

Parse({string})

Create a `parse` class method that can **parse** Hexidecimal numbers and convert them to standard decimal numbers.

Submission

Submit only your **Hex class.**

Examples

This is an example how the code is **intended to be used**:

Sample exectuion

`let FF = new Hex(255);`
`console.log(FF.toString());`
`FF.valueOf() + 1 == 256;`
`let a = new Hex(10);`
`let b = new Hex(5);`
`console.log(a.plus(b).toString());`

© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.

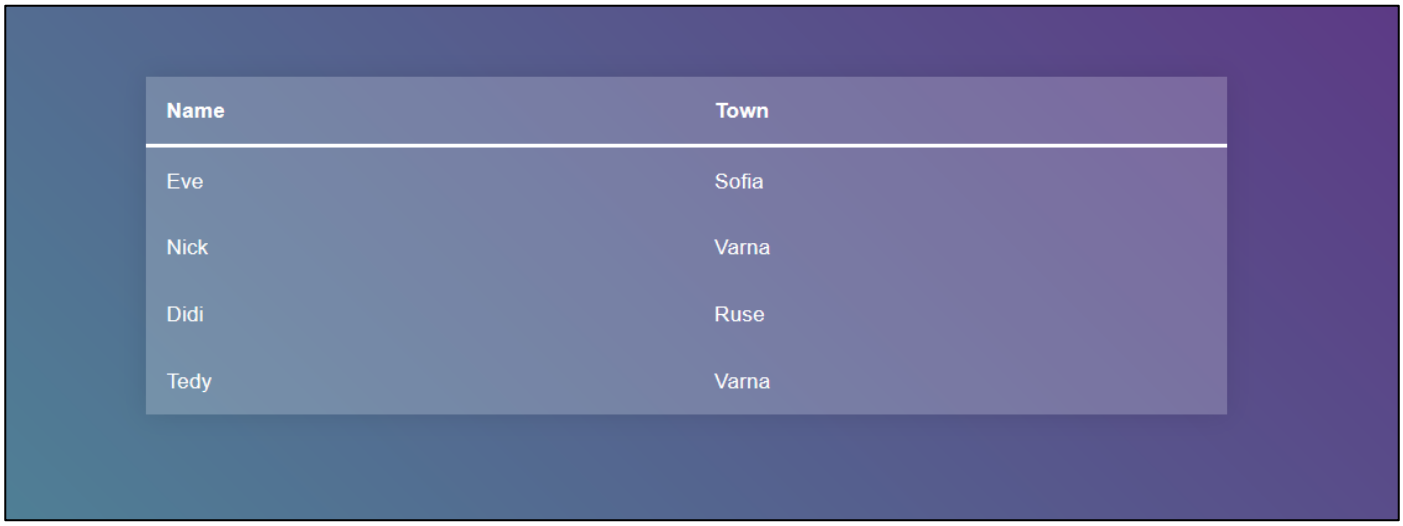
```
console.log(a.plus(b).toString()==='0xF');
```

```
0xFF
0xF
True
```

4. Table

Use the Given Skeleton to Solve This Problem.

Note: You Have NO Permission to Change Directly the Given HTML (Index.html File).



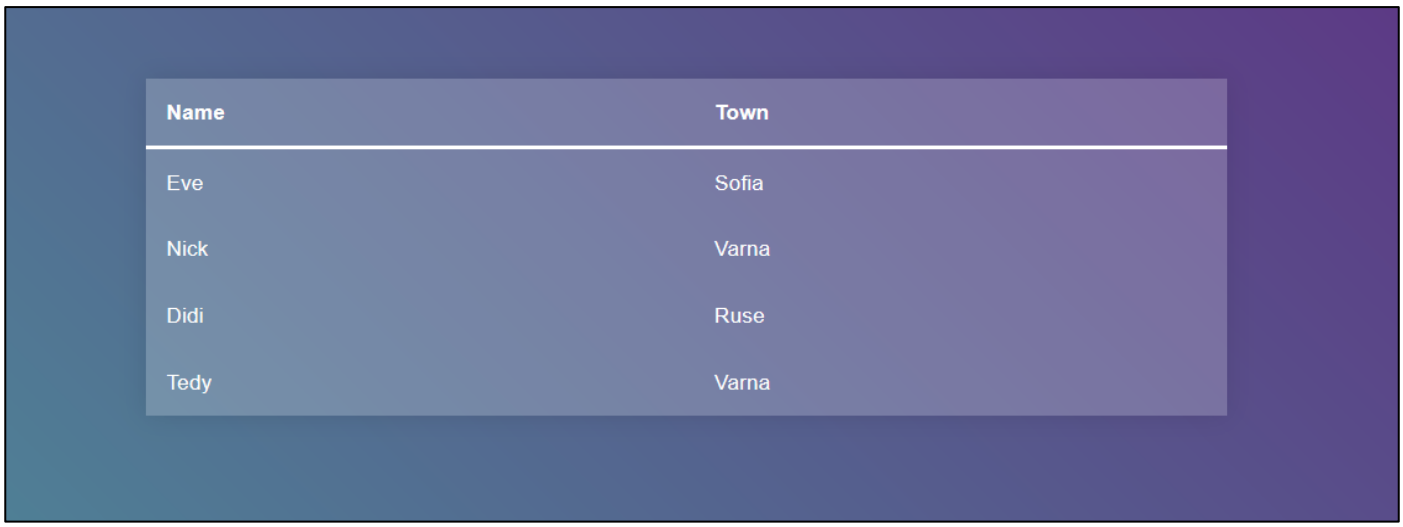
Your Task

Write the missing JavaScript code to make the **Table** application work as expected.

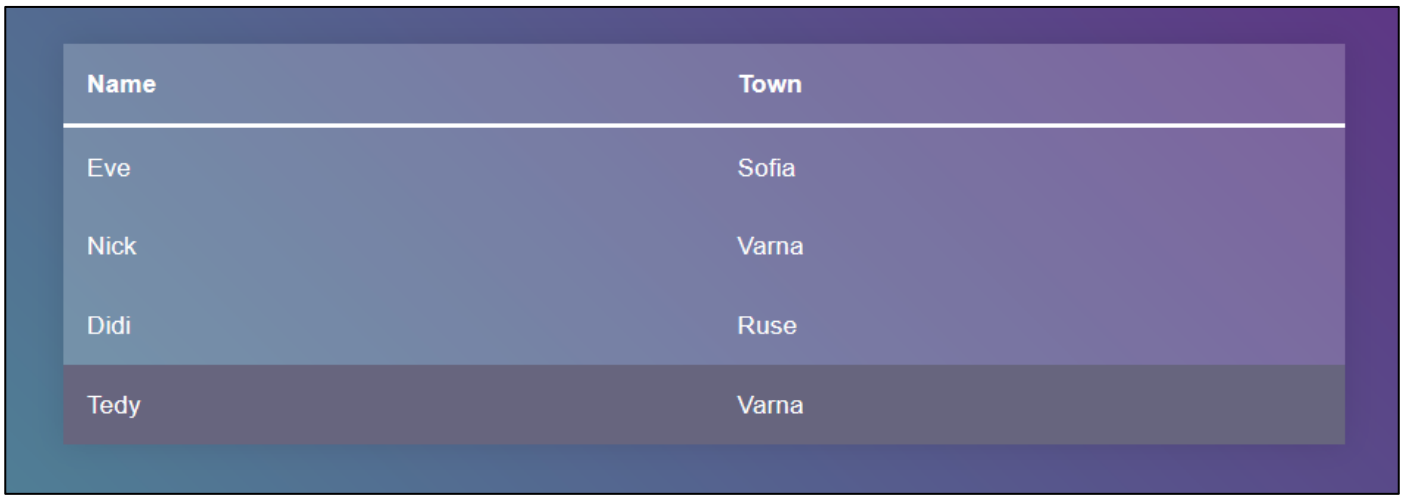
When you **click** on an item from the table you should change its **background color** to "#413f5e".

If the item you've clicked **already has** a **style** property you should **remove** it.

© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.

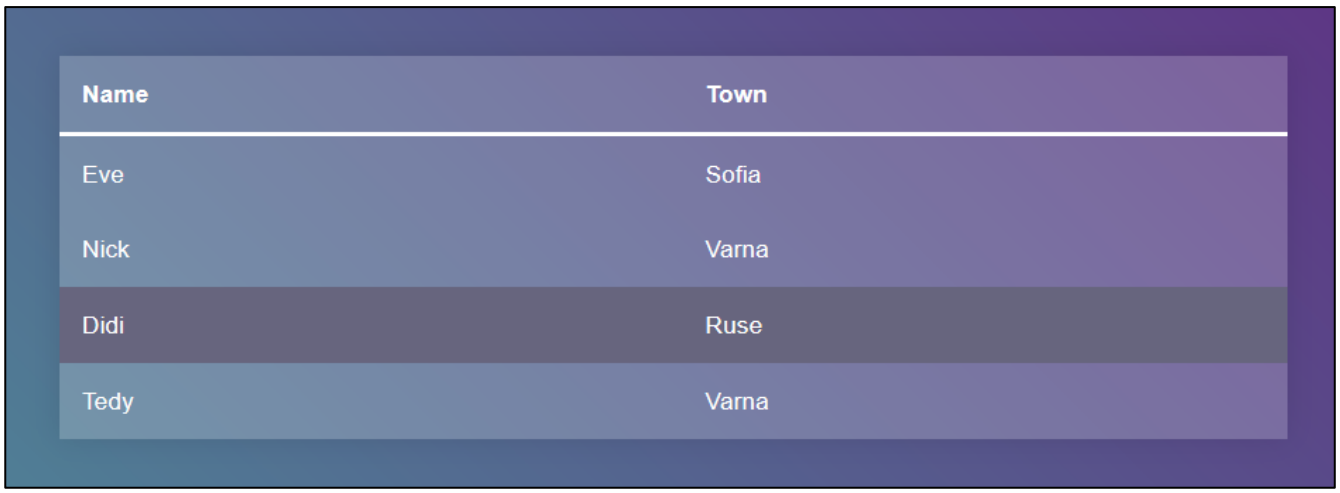


If one of the elements is **clicked** and you click **another** the first element's style property should be **removed** and you should **change** the **background color** of the **newly clicked** item.




```
<table class="minimalistBlack">
  <thead> ... </thead>
  <tbody> event
    <tr> ... </tr>
    <tr> ... </tr>
    <tr> ... </tr>
    <tr style="background-color: rgb(65, 63, 94);"> ... </tr>
  </tbody>
</table>
```


© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.




```
<table class="minimalistBlack">
  <thead> ··· </thead>
  <tbody> event
    <tr> ··· </tr>
    <tr> ··· </tr>
    <tr style="background-color: rgb(65, 63, 94);"> ··· </tr>
    <tr> ··· </tr>
  </tbody>
</table>
```

Note: You **shouldn't** change the head of the table, even if it is clicked.

© Kingsland – https://kingslanduniversity.com. Unauthorized copy, reproduction or use is not permitted.