**JS Advanced Retake Exam**

**Problem 3. Unit Testing**

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable named **companyAdministration**, which represents an object. You may use the following code as a template:

|  |
| --- |
| describe(**"*Tests* …"**, **function**() {  describe(**"*TODO* …"**, **function**() {  ***it***(**"*TODO …*"**, **function**() {  *//* ***TODO:*** …  });  });  *//* ***TODO:*** …  }); |

The object should have the following functionality:

**hiringEmployee (name, position,** **yearsExperience) -** A function that accepts three parameters: **string**, **string**, and **number**.

* If the value of the string **position** is different from "**Programmer**", **throw** an error: **`We are not looking for workers for this position.`**
* To be hired, the **employee** must meet the **following requirements**:
  + If the **yearsЕxperience** are **greater** than or equal to **3**, **return** the string:

**`{name} was successfully hired for the position {position}.`**

* Otherwise, if the above conditions are not met, **return** the following message:

**`{name} is not approved for this position.`**

* There is **no** need for **validation** for the **input**, you will always be given a string, string, and number.
* **calculateSalary (hours) -** A function that accepts one parameter: **number**.
* Workers in this company receive **equal** pay per **hour** and this is **BGN 15**.
* You need to **calculate** the salary by **multiplying** the pay **for one hour** by the number of **hours**.
* **Also**, if the employee has been working for **more than 160 hours**, he must receive an additional **BGN 1000 bonus.**
* Finally, **return** the employee's salary.
* You need to validate the input, if the **hours** are not a **number**, or are a **negative** number, **throw** an error: "**Invalid hours**".
* **firedEmployee** **(employees, index) -** A function that accepts an array and number.
  + The **employees** array will store the names of its employees (["**Petar**", "**Ivan**", "**George**"…]).
  + You must **remove** an **element** (employee) from the **array** that is located on the **index** specified as a parameter.
* Finally, **return** the changed array of employees as a string, **joined** by a **comma** and a **space**.
  + There is a need for validation for the input, an **array** and index may not always be valid. In case of submitted **invalid** parameters, **throw** an error "**Invalid input**":
    - If passed **employees** parameteris not an array.
    - If the **index** is not a number and is outside the limits of the array.

**JS Code**

To ease you in the process, you are provided with an implementation that meets all of the specification requirements for the **companyAdministration** object:

|  |
| --- |
| companyAdministration.js |
| const companyAdministration = {      hiringEmployee(name, position, yearsExperience) {          if (position == "Programmer") {              if (yearsExperience >= 3) {                  return `${name} was successfully hired for the position ${position}.`;              } else {                  return `${name} is not approved for this position.`;              }          }          throw new Error(`We are not looking for workers for this position.`);      },      calculateSalary(hours) {          let payPerHour = 15;          let totalAmount = payPerHour \* hours;          if (typeof hours !== "number" || hours < 0) {              throw new Error("Invalid hours");          } else if (hours > 160) {              totalAmount += 1000;          }          return totalAmount;      },      firedEmployee(employees, index) {          let result = [];          if (!Array.isArray(employees) || !Number.isInteger(index) || index < 0 || index >= employees.length) {              throw new Error("Invalid input");          }          for (let i = 0; i < employees.length; i++) {              if (i !== index) {                  result.push(employees[i]);              }          }          return result.join(", ");      }  } |

**Submission**

Submit your tests inside a **describe()** statement, as shown above.