**03. Workforce Management**

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable named **workforceManagement**, 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:

**recruitStaff (name, role,** **experience) -** A function that accepts three parameters: **string**, **string**, and **number**.

* If the value of the string **role** is different from "**Developer**", **throw** an error: **`We are not currently hiring for this role.`**
* To be hired, the **employee** must meet the **following requirements**:
  + If the **experience** is **greater** than or **equal** to **4**, **return** the string:

**`{name} has been successfully recruited for the role of {role}.`**

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

**`{name} is not suitable for this role.`**

* There is **no** need for **validation** for the **input**, you will always be given a string, string, and number.
* **computeWages (hoursWorked) -** A function that accepts one parameter: **number**.
* Workers in this company receive **equal** pay per **hour** and this is **BGN 18**.
* 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 1500 bonus.**
* Finally, **return** the employee's salary.
* You need to validate the input, if the **hoursWorked** are not a **number**, or are a **negative** number, **throw** an error: "**Invalid hours**".
* **dismissEmployee (workforce, employeeIndex) -** A function that accepts an array and number.
  + The **workforce** 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 **employeeIndex** specified as a parameter.
* Finally, **return** the changed array of workforce 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 **workforce** parameteris not an array.
    - If the **employeeIndex** 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 **workforceManagement** object:

|  |
| --- |
| workforceManagement.js |
| const workforceManagement = {    recruitStaff(name, role, experience) {      if (role === "Developer") {        if (experience >= 4) {          return `${name} has been successfully recruited for the role of ${role}.`;        } else {          return `${name} is not suitable for this role.`;        }      }      throw new Error(`We are not currently hiring for this role.`);    },    computeWages(hoursWorked) {      let hourlyRate = 18;      let totalPayment = hourlyRate \* hoursWorked;      if (typeof hoursWorked !== "number" || hoursWorked < 0) {        throw new Error("Invalid hours");      } else if (hoursWorked > 160) {        totalPayment += 1500;      }      return totalPayment;    },    dismissEmployee(workforce, employeeIndex) {      let updatedStaff = [];      if (!Array.isArray(workforce) || !Number.isInteger(employeeIndex) || employeeIndex < 0 || employeeIndex >= workforce.length) {        throw new Error("Invalid input");      }      for (let i = 0; i < workforce.length; i++) {        if (i !== employeeIndex) {          updatedStaff.push(workforce[i]);        }      }      return updatedStaff.join(", ");    }  }; |

**Submission**

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