JS Advanced Exam-Retake

Problem 3. Unit Testing

Your Task

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

- **`PickVenue (capacity, pricePerGuest, location)`** - A function that accepts **three** parameters: **number**, **number**, and **string**.
    - There is a **need for validation** for the input, in case of submitted **invalid** parameters or empty **string**, **throw** an error **`"Invalid Information!`**
        - If the value of the string **`location`** is different from "**Varna**", **throw** an error:

          `"The location of this venue is not in the correct area!"`
    - To be picked, the **venue** must meet the **following requirement**:
        - If the **`capacity`** of the venue is **greater** or **equal** to **150**, and **pricePerGuest** is **less** or **equal** to **120 return** the string:

          `"This venue meets the requirements, with capacity of ${capacity} guests and ${pricePerGuest}$ cover."`
    - Otherwise, if the above conditions are **not** met, **return** the following message:

      `"This venue does not meet your requirements!"`
- **`otherSpendings (weddingDecoration, photography, discount)`** - A function that accepts three parameters: **array, array,** and **boolean**.
    - Calculate the **total price** you are going to pay depending on the purchased **weddingDecoration** and **photography:**
    - The theater offers **two** options for **weddingDecoration** and **photography**:
        - The two options for **weddingDecoration** are:
            - **`flowers`**, which costs **`$500`**
            - **`Fabric drapes and curtains`**, which costs **`$400`**

        - The two options for **photography** are:
            - **pictures**, which costs **$700**
            - **video**, which costs **$1300**
    - If the **discount** is **true,** a **15%** discount should be applied**.** Then **return** the following message:

      "You spend {totalPrice}$ for wedding decoration and photography with 15% discount!"
    - Else, **return** the following message:

      "You spend {totalPrice}$ for wedding decoration and photography!"
  - You need to validate the input, if the **weddingDecoration, photography** and **discount** are not an **array, array** and **Boolean, throw** an error: "**Invalid Information!**"

- **tableDistribution (guests, tables) -** A function that accepts two parameters: **number, number**.
    - You need to **calculate** how many guests on table you will have**.**
  - If the **peopleOnTable** are **less** than **6**. **return** the following message:

    "There is only {peopleOnTable} people on every table, you can join some tables."
    - Else, **return** the following message:

      "You have {tables} tables with {peopleOnTable} guests on table."
    - You **need to validate** the input, if the **guests** and **tables** are not a **numbers**, or are a **negative** numbers, **throw** an error: **"Invalid Information!"**.

JS Code

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

weddingDay.js

```
const weddingDay = {
    pickVenue(capacity, pricePerGuest, location) {
        if (typeof capacity !== 'number' || typeof pricePerGuest !== 'number' || typeof location !== 'string' || location === '') {
            throw new Error("Invalid Information!")
          };
            if (location == "Varna") {

                if (capacity >= 150 && pricePerGuest <= 120) {
                    return `This venue meets the requirements, with capacity of ${capacity} guests and ${pricePerGuest}$ cover.`;
                } else {
```

```
                    return `This venue does not meet your requirements!`;
                }
            }else {
                throw new Error(`The location of this venue is not in the correct area!`);
            };
    },

    otherSpendings(weddingDecoration, photography, discount) {
        if (!Array.isArray(weddingDecoration) || !Array.isArray(photography) || typeof discount
!== "boolean") {
          throw new Error("Invalid Information!");
        }
        let totalPrice = 0;

        weddingDecoration.forEach((decoration) => {
          if (decoration === "flowers") {
            totalPrice += 500
          } else if (decoration === "Fabric drapes and curtains") {
            totalPrice += 400
          }
        });

        photography.forEach((service) => {
          if (service === "pictures") {
            totalPrice += 700
          } else if (service === "video") {
            totalPrice += 1300
          }
        });
        if (discount) {
          totalPrice = totalPrice * 0.85;
          return `You spend ${totalPrice}$ for wedding decoration and photography with 15%
discount!`
        } else {
          return `You spend ${totalPrice}$ for wedding decoration and photography!`
        }
```

```
    }
,
    tableDistribution(guests, tables) {
        if (typeof guests !== "number" || guests <= 0 ||
            typeof tables !== "number" || tables <= 0) {
            throw new Error("Invalid Information!");
        }
        let peopleOnTable = Math.round(guests / tables);

        if(peopleOnTable < 6) {
            return `There is only ${peopleOnTable} people on every table, you can join some
tables.`
        }else{
            return `You have ${tables} tables with ${peopleOnTable} guests on table.`
        }
    }
}
```

Submission

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