**3. Plan Your Trip**

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable named **planYourTrip**, 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 that should have the following functionality:

* **choosingDestination (destination, season, year) -** A function that accepts **three** parameters: **string**, **string**, and **number**.
* If the **year** is different than **2024**, **throw** an error: **"Invalid Year!"**
* If the value of the string **destination** is different from "**Ski Resort**", **throw** an error:

**"This destination is not what you are looking for."**

* To be picked, the **destination** must meet the **following requirement**:
  + If the **season** is **Winter**, **return** the string:

**"Great choice! The ${season} is the perfect time to visit the ${destination}."**

* Otherwise, if the above condition in **not** met, **return** the following message:

**"Consider visiting during the Winter for the best experience at the ${destination}."**

* There is **no** need for **validation** for the **input**, you will always be given two strings, and number.
* **exploreOptions (activities, activityIndex) -** A function that accepts an **array** and **number**. The **activities** array will store the different activities (["**Skiing** ", "**Snowboarding** ", "**Winter Hiking** "…]).
  + You must **remove** an **element** (activity) from the **array** that is located on the **index** specified as a parameter.
  + Finally, **return** the changed array of activities 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 Information!"**:
    - If passed **activities** parameteris **not** an array.
    - If the **activityIndex** is not a number and is outside the limits of the array.
    - If the **activityIndex** is not a integer number.
* **estimateExpenses (distanceInKilometers, fuelCostPerLiter) -** A function that accepts two parameters: **number, number**.
* You need to **calculate** the **cost** of the travel.
  + - **The result must be formatted to the second digit after the decimal point.**
* If the total cost is **less** or **equal** to $500. **return** the following message:

**"The trip is budget-friendly, estimated cost is $${totalCost}."**

* Else, **return** the following message:

**"The estimated cost for the trip is $${totalCost}, plan accordingly."**

* You **need to validate** the input, if the **distanceInKilometers** and **fuelCostPerLiter** are not **numbers**, or are **negative** numbers or **zero**, **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 **planYourTrip** object:

|  |
| --- |
| planYourTrip.js |
| const planYourTrip = {    choosingDestination(destination, season, year) {      if (year != 2024) {        throw new Error(`Invalid Year!`);      } else {        if (destination == "Ski Resort") {          if (season === "Winter") {            return `Great choice! The ${season} is the perfect time to visit the ${destination}.`;          } else {            return `Consider visiting during the Winter for the best experience at the ${destination}.`;          }        } else {          throw new Error(`This destination is not what you are looking for.`);        }      }    },    exploreOptions(activities, activityIndex) {      let result = [];      if (        !Array.isArray(activities) ||        !Number.isInteger(activityIndex) ||        activityIndex < 0 ||        activityIndex >= activities.length      ) {        throw new Error("Invalid Information!");      }      for (let i = 0; i < activities.length; i++) {        if (i !== activityIndex) {          result.push(activities[i]);        }      }      return result.join(", ");    },    estimateExpenses(distanceInKilometers, fuelCostPerLiter) {      let totalCost = (distanceInKilometers \* fuelCostPerLiter).toFixed(2);      if (        typeof distanceInKilometers !== "number" ||        distanceInKilometers <= 0 ||        typeof fuelCostPerLiter !== "number" ||        fuelCostPerLiter <= 0      ) {        throw new Error("Invalid Information!");      } else if (totalCost <= 500) {        return `The trip is budget-friendly, estimated cost is $${totalCost}.`;      } else {        return `The estimated cost for the trip is $${totalCost}, plan accordingly.`;      }    },  }; |

**Submission**

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