Async JS Practice Question Set 1

Instructions:

- Specific instructions have been written for each question present in this set.
- Don’t use async-await in this set.
- Please follow ES6 norms for writing your functions.
- You can make use of some methods such as .length, toLowerCase(), toUpperCase() and .join() if needed.
- An example has been provided for fetch call related questions for your understanding.

1. Write a function ‘delayedGreeting’ that consoles a greeting message after a delay of 2 seconds using setTimeout. You can practice this question in any JS editor or your browser console.

```
// Your Code here

delayedGreeting();
// Hello, welcome to my portfolio!
```

COPY

2. Write a function ‘delayedAddition’ that takes in two numbers and consoles their sum after a delay of 4 seconds using setTimeout. You can practice this question in any JS editor or your browser console.

```
// Your code here

delayedAddition(2, 3);
// 5
```

COPY

3. Write a function ‘delayAlert’ that takes in a message ‘Hello, world!’ and a delay time in milliseconds, and displays the message in an alert box after the specified delay time using setTimeout. You can practice this question in any JS editor or your browser console.

```
// Your Code here

delayedAlert('Hello, world!', 2000);
// Should display an alert box with the message, Hello, world!
```

COPY

4. Write a function delayedLoop that takes a number 3 and consoles a message 'Hello' three times after a delay of 1 second each, using a for-loop and setTimeout. You can practice this question in any JS editor or your browser console.

```
// Your Code here

delayedLoop(3);
// should print Hello three times after 1 second each
// Output:
// Hello -- after 1 second
// Hello -- after another 1 second
// Hello -- after another 1 second
```

COPY

5. Make a fake fetch call that takes a message and a boolean value to get data and console the message received from the server. A fake fetch has been provided. You can practice this question in any JS editor or your browser console.

```
const fakeFetch = (msg, shouldSucceed) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (shouldSucceed) {
        resolve(`message from server: ${msg}`);
      }
      reject(`error from server: ${msg}`);
    }, 3000);
  });
};

// Your Code here

// Hi -- after 3 seconds
```

COPY

6. EXAMPLE QUESTION: (Solution has been provided for this question for your understanding)

   Use this URL - *https://example.com/api/itemlist* to make a fake fetch call and handle errors if any. Show a proper message to the user on the DOM, as per the status and message received from the server. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/itemlist") {
        reject({
          status: 404,
        message: "Items list not found."
      });
    } else {
        resolve({
          status: 200,
        data: {
            message: "Success",
          data: [
              { itemName: "Bread", price: 30 },
            { itemName: "Water Bottle", price: 50 },
            { itemName: "Dairy Milk", price: 20 }
          ]
        }
      });
    }
  }, 2000);
});
}

// Your Code here (Solution Given)
const displayOutput = document.querySelector("#output");

fakeFetch("https://example.com/api/itemlist")
  .then((response) => console.log(response))
  .catch((error) => {
    if (error.status === 404) {
      displayOutput.textContent =
        "The data you are looking for, does not exist.";
    }
  });

// Output on the DOM should be:
// The data you are looking for, does not exist.
```

COPY

Explanation:

In the above code solution, we are making a fakeFetch function call with the URL `https://example.com/api/itemlist`.

If the Promise is resolved, the **`then`**method is executed with the successful response as the argument, and the console.log statement outputs the response object to the console.

If the Promise is rejected, the **`catch`**method is executed with the error object as the argument, and the `if`statement checks if the error status is equal to 404. If the error status is 404, the message "The data you are looking for, does not exist." is displayed in the HTML element with ID "output".

7. Use this URL - `https://example.com/api/chat` to make a fake fetch call and handle errors if any. Show a proper message to the user on the DOM, as per the status and message received from the server. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/chat") {
        reject({
          status: 503,
        message: "Service Unavailable"
      });
    } else {
        resolve({
          status: 200,
        data: {
            message: "Success"
        }
      });
    }
  }, 2000);
});
};

// Your Code here

// Output on the DOM should be:
// We are facing high demand at the moment. Please check back later in sometime.
```

COPY

8. Use this URL - *https://example.com/api/itemlist* to make a fake fetch call and list out all the items as an ordered list on the DOM. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/itemlist") {
        resolve({
          status: 200,
          message: "Success",
          data: [
            { itemName: "Bread", price: 30, quantity: 10 },
            { itemName: "Water Bottle", price: 50, quantity: 50 },
            { itemName: "Dairy Milk", price: 20, quantity: 30 }
          ]
        });
      } else {
        reject({
          status: 404,
          message: "Items list not found."
        });
      }
    }, 2000);
  });
};

// Your Code here

// Output on the DOM should be in the format, {itemName} -- INR {price} -- {quantity}:
// 1. Bread -- INR 30 -- 10
// 2. Water Bottle -- INR 50 -- 50
// 3. Dairy Milk -- INR 20 -- 30
```

COPY

9. Use this URL - *https://example.com/api/data* to make a fake fetch call and handle errors if any. Show a proper message to the user on the DOM, as per the status and message received from the server. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/data") {
        reject({
          status: 500,
        message: "Internal Server Error"
      });
    } else {
        resolve({
          status: 200,
        data: {
            message: "Success"
        }
      });
    }
  }, 2000);
});
};
```

```
// Your Code here

// Output on the DOM should be:
// Internal Server Error! The server crashed. Please try again in some time.
```

COPY

10. Use this URL - *`https://example.com/api/profile`* to make a fake fetch call and handle errors if any. Show a proper message to the user on the DOM, as per the status and message received from the server. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/profile") {
        reject({
          status: 401,
          message: "Unauthorized Access"
        });
      } else {
        resolve({
          status: 200,
          data: {
            message: "Success"
          }
        });
      }
    }, 2000);
  });
};

// Your Code here

// Output on the DOM should be:
// Unauthorized Access! Looks like you are not logged in. Please login to see your profil
```

COPY

11. Use this URL - *`https://example.com/api/profile/NC002`* in which we are passing the id of a user to make a fake fetch call and display a welcome message to the user on the DOM. A fakeFetch has been provided. Use HTML, CSS & JS template in REPL or Vanilla template in CodeSandbox for this question.

```
const fakeFetch = (url) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (url === "https://example.com/api/profile/NC002") {
        resolve({
          status: 200,
          data: {
            message: "Success",
            data: { id: "NC002", name: "Roshan", institute: "neoG Camp" }
          }
        });
      } else {
        reject({
          status: 404,
          message: "Resource not found"
        });
      }
    }, 2000);
  });
};

// Your Code here
// Output on the DOM should be: Welcome!, Roshan from neoG Camp
```