Bài tập bài 11: Javascript nâng cao (Tiết 1)

Câu 01: Ví dụ về hoisting biến

- Đề bài: Dự đoán kết quả console của đoạn code sau.
- `var x = 5;`
- `console.log(x + y);`
- `var y = 10;`
- Đáp án:
- `var x = 5;`
- `console.log(x + y); // NaN`
- `var y = 10;`
    - Trong ví dụ này, biến x được gán giá trị trước khi sử dụng, nhưng biến y vẫn chưa được khai báo. Kết quả là y sẽ có giá trị undefined, và khi tính toán x + y, kết quả sẽ là NaN (Not a Number).

Câu 02: Ví dụ về hoisting hàm

- Đề bài: Dự đoán kết quả console của đoạn code sau.
- `foo();`
- `function foo() {`
- `  console.log("Hello");`
- `}`
- Đáp án:
- `foo(); // "Hello"`
- `function foo() {`
- `  console.log("Hello");`
- `}`
    - Trong ví dụ này, hàm foo được sử dụng trước khi nó được khai báo. JavaScript tự động di chuyển phần khai báo của hàm foo lên đầu phạm vi hiện tại.

Câu 03: Ví dụ về hoisting với khối mã (block scope)

- Đề bài: Dự đoán kết quả console của đoạn code sau.
- `function foo() {`
- `  if (true) {`
- `    var x = 5;`
- `  }`
- `  console.log(x);`
- `}`
- `foo();`
- Đáp án:
- `function foo() {`
- `  if (true) {`
- `    var x = 5;`
- `  }`
- `  console.log(x); // 5`
- `}`
- `foo();`
    - Trong ví dụ này, biến x được khai báo trong một khối mã (if statement), nhưng vẫn có thể truy cập và sử dụng ngoài khối mã đó.

Câu 04: Ví dụ về hoisting với let và const (block scope)

- Đề bài: Dự đoán kết quả console của đoạn code sau.
- `function foo() {`

```
  if (true) {
    let x = 5;
    const y = 10;
  }
  console.log(x);
  console.log(y);
}
foo();
```

Đáp án:

```
function foo() {
  if (true) {
    let x = 5;
    const y = 10;
  }
  console.log(x); // ReferenceError: x is not defined
  console.log(y); // ReferenceError: y is not defined
}
foo();
```

- Trong ví dụ này, biến x và y được khai báo bằng let và const, nên chúng chỉ có hiệu lực trong phạm vi khối mã mà chúng được khai báo.

Câu 05: Ví dụ về hoisting với hàm trong một khối mã

Đề bài: Dự đoán kết quả console của đoạn code sau.

```
function foo() {
  if (true) {
    function bar() {
      console.log("Hello");
    }
    bar();
  }
  bar();
}
foo();
```

Đáp án:

```
function foo() {
  if (true) {
    function bar() {
      console.log("Hello");
    }
    bar(); // "Hello"
  }
  bar(); // "Hello"
}
foo();
```

- Trong ví dụ này, hàm **bar** được khai báo trong khối mã **if**, nhưng vẫn có thể truy cập và sử dụng ngoài khối mã đó.

Câu 06: Ví dụ về hoisting với hàm trong một khối mã (let)

Đề bài: Dự đoán kết quả console của đoạn code sau.

```
function foo() {
  if (true) {
    let bar = function () {
      console.log("Hello");
    };
    bar();
  }
  bar();
}
```

```
foo();
```
- Đáp án:

```
function foo() {
  if (true) {
    let bar = function () {
      console.log("Hello");
    };
    bar(); // "Hello"
  }
  bar(); // ReferenceError: bar is not defined
}
foo();
```

  - Trong ví dụ này, hàm **bar** được khai báo bằng từ khóa **let** trong khối mã **if**, nên nó chỉ có hiệu lực trong phạm vi khối mã đó.

Câu 07: Ví dụ về hoisting trong vòng lặp

- Đề bài: Dự đoán kết quả console của đoạn code sau.

```
// Dùng từ khóa var
for (var i = 0; i < 5; i++) {
  console.log(i);
}
console.log("i =", i);

// Dùng từ khóa let
for (let j = 0; j < 5; j++) {
  console.log(j);
}
console.log("j =", j);
```

- Đáp án:

```
// Dùng từ khóa var
for (var i = 0; i < 5; i++) {
  console.log(i); // 0, 1, 2, 3, 4
}
console.log("i =", i); // Cuối cùng: 5

// Dùng từ khóa let
for (let j = 0; j < 5; j++) {
  console.log(j); // 0, 1, 2, 3, 4
}
console.log("j =", j); // ReferenceError: j is not defined
```

  - Trong ví dụ này, biến **i** được khai báo bằng từ khóa **var** trong vòng lặp, nhưng vẫn có thể truy cập và sử dụng ngoài vòng lặp. Còn biến **j** được khai báo bằng từ khóa **let** trong vòng lặp, nên chỉ có thể truy cập và sử dụng bên trong vòng lặp.

Câu 08: Viết hàm tính toán

- Đề bài:
  - Tạo 1 file có tên math.js. Viết các hàm tổng, hiệu, tích, thương và export các hàm đó.
  - Tạo 1 file có tên main.js. Import các hàm tính toán bên trên và tính các phép tính sau:
    - tong(2, 3) —> 5
    - hieu(5, 2) —> 3
    - tich(2, 3) —> 6
    - thuong(6, 2) —> 3
- Đáp án:

- File math.js
- ```
  export function tong(a, b) {
    return a + b;
  }

  export function hieu(a, b) {
    return a - b;
  }

  export function tich(a, b) {
    return a * b;
  }

  export function thuong(a, b) {
    return a / b;
  }
  ```
- File main.js
- ```
  import { tong, hieu, tich, thuong } from "./test.js";

  console.log(tong(2, 3)); // 5
  console.log(hieu(5, 2)); // 3
  console.log(tich(2, 3)); // 6
  console.log(thuong(6, 2)); // 3
  ```

Câu 09: Trích xuất các thuộc tính từ JSON

- Đề bài:
  - Cho một chuỗi JSON như sau:
  - ```
    const dataJSON = `{
      "name": "Lê Văn A",
      "age": 20,
      "email": "levana@gmail.com",
      "address": {
        "street": "Số 123, đường ABC",
        "city": "Hà Nội",
        "country": "Việt Nam"
      }
    }`;
    ```
  - Hãy trích xuất các thuộc tính "name", "email" và "city" và in ra màn hình console.
- Đáp án:

```
const dataJSON = `{
  "name": "Lê Văn A",
  "age": 20,
  "email": "levana@gmail.com",
  "address": {
    "street": "Số 123, đường ABC",
    "city": "Hà Nội",
    "country": "Việt Nam"
  }
}`;

const dataJS = JSON.parse(dataJSON);

console.log(dataJS.name); // Lê Văn A
console.log(dataJS.email); // levana@gmail.com
console.log(dataJS.address.city); // Hà Nội
```

Câu 10: Xử lý mảng đối tượng JSON
```

- Đề bài:
  - Cho một mảng đối tượng JSON như sau:
  - ```
    const dataJSON = `
      [
        {
          "name": "Lê Văn A",
          "age": 30,
          "skills": ["JavaScript", "HTML", "CSS"]
        },
        {
          "name": "Nguyễn Thị B",
          "age": 25,
          "skills": ["Python", "Java", "C++"]
        },
        {
          "name": "Đỗ Văn C",
          "age": 35,
          "skills": ["Ruby", "PHP", "SQL"]
        }
      ]
    `;
    ```
  - Hãy tạo một mảng mới chứa tên của tất cả các người trong mảng ban đầu: **["Lê Văn A", "Nguyễn Thị B", "Đỗ Văn C"]**.
- Đáp án:

```
const dataJSON = `
  [
    {
      "name": "Lê Văn A",
      "age": 30,
      "skills": ["JavaScript", "HTML", "CSS"]
    },
    {
      "name": "Nguyễn Thị B",
      "age": 25,
      "skills": ["Python", "Java", "C++"]
    },
    {
      "name": "Đỗ Văn C",
      "age": 35,
      "skills": ["Ruby", "PHP", "SQL"]
    }
  ]
`;

const dataJS = JSON.parse(dataJSON);

const namesArray = dataJS.map((obj) => obj.name);
console.log(namesArray); // ["Lê Văn A", "Nguyễn Thị B", "Đỗ Văn C"]
```

Câu 11: Tính tổng giá trị đơn hàng

- Đề bài:
  - Cho 2 chuỗi JSON chứa thông tin đơn hàng (ordersJSON) và thông tin sản phẩm (productsJSON):
  - ```
    const ordersJSON = `
      [
        {
          "id": 1,
          "items": [
            {
              "productId": 1,
    ```

```
            "quantity": 2
          },
          {
            "productId": 2,
            "quantity": 1
          }
        ]
      },
      {
        "id": 2,
        "items": [
          {
            "productId": 3,
            "quantity": 3
          }
        ]
      }
    ]
  `;

  const productsJSON = `
    [
      {
        "id": 1,
        "name": "iPhone 12",
        "price": 1200
      },
      {
        "id": 2,
        "name": "Samsung Galaxy S21",
        "price": 1000
      },
      {
        "id": 3,
        "name": "Google Pixel 5",
        "price": 900
      }
    ]
  `;
```

- Viết một hàm có tên **calculateOrderTotal()** nhận vào một **id đơn hàng** và tính tổng giá trị của đơn hàng đó, sử dụng thông tin từ mảng đối tượng JSON "ordersJSON" và "productsJSON".
- Ví dụ:
  - calculateOrderTotal(1) —> 3400
  - calculateOrderTotal(2) —> 2700
  - calculateOrderTotal(3) —> Không tìm thấy đơn hàng.

- Đáp án:

```
const ordersJSON = `
  [
    {
      "id": 1,
      "items": [
        {
          "productId": 1,
          "quantity": 2
        },
        {
          "productId": 2,
          "quantity": 1
        }
      ]
    },
```

```
      {
        "id": 2,
        "items": [
          {
            "productId": 3,
            "quantity": 3
          }
        ]
      }
    ]
  `;

  const productsJSON = `
    [
      {
        "id": 1,
        "name": "iPhone 12",
        "price": 1200
      },
      {
        "id": 2,
        "name": "Samsung Galaxy S21",
        "price": 1000
      },
      {
        "id": 3,
        "name": "Google Pixel 5",
        "price": 900
      }
    ]
  `;

  const ordersJS = JSON.parse(ordersJSON);
  const productsJS = JSON.parse(productsJSON);

  function calculateOrderTotal(orderId) {
    const order = ordersJS.find((item) => item.id === orderId);
    if (order) {
      let total = 0;
      for (const item of order.items) {
        const product = productsJS.find((p) => p.id ===
item.productId);
        if (product) {
          total += product.price * item.quantity;
        }
      }
      return total;
    } else {
      return "Không tìm thấy đơn hàng.";
    }
  }

  console.log(calculateOrderTotal(1)); // 3400 (Vì: 2 * 1200 + 1 *
1000)
  console.log(calculateOrderTotal(2)); // 2700 (Vì: 3 * 900)
  console.log(calculateOrderTotal(3)); // Không tìm thấy đơn hàng.
```

Câu 12: Xóa sản phẩm từ JSON

- Đề bài:
  - Cho một mảng đối tượng JSON chứa thông tin các sản phẩm:
  - let productsJSON = `
  -   [

```
    {
      "id": 1,
      "name": "iPhone 12",
      "price": 1200
    },
    {
      "id": 2,
      "name": "Samsung Galaxy S21",
      "price": 1000
    },
    {
      "id": 3,
      "name": "Google Pixel 5",
      "price": 900
    }
  ]
`;
```

- Viết một hàm có tên **deleteProduct()** nhận vào **id của một sản phẩm** và xóa sản phẩm đó khỏi mảng JSON "productsJSON" và cập nhật lại mảng.
- Ví dụ:
  - deleteProduct(2) —> [{"id":1,"name":"iPhone 12","price":1200},{"id":3,"name":"Google Pixel 5","price":900}]

- Đáp án:

```
let productsJSON = `
  [
    {
      "id": 1,
      "name": "iPhone 12",
      "price": 1200
    },
    {
      "id": 2,
      "name": "Samsung Galaxy S21",
      "price": 1000
    },
    {
      "id": 3,
      "name": "Google Pixel 5",
      "price": 900
    }
  ]
`;

let productsJS = JSON.parse(productsJSON);

function deleteProduct(productId) {
  productsJS = productsJS.filter(item => item.id !== productId);
  productsJSON = JSON.stringify(productsJS);
}

deleteProduct(2);
console.log(productsJSON);
// [{"id":1,"name":"iPhone 12","price":1200},{"id":3,"name":"Google Pixel 5","price":900}]
```

Câu 13: Tính tổng số lượng sản phẩm từ JSON

- Đề bài:
  - Cho một mảng đối tượng JSON chứa thông tin các sản phẩm:
  - const productsJSON = `
  - [

```
    {
      "id": 1,
      "name": "iPhone 12",
      "quantity": 10
    },
    {
      "id": 2,
      "name": "Samsung Galaxy S21",
      "quantity": 5
    },
    {
      "id": 3,
      "name": "Google Pixel 5",
      "quantity": 8
    }
  ]
`;
```

- Viết một hàm **calculateTotalQuantity()** để tính tổng số lượng sản phẩm từ mảng đối tượng JSON.
- Ví dụ:
  - calculateTotalQuantity() —> 23

- Đáp án:

```
const productsJSON = `
  [
    {
      "id": 1,
      "name": "iPhone 12",
      "quantity": 10
    },
    {
      "id": 2,
      "name": "Samsung Galaxy S21",
      "quantity": 5
    },
    {
      "id": 3,
      "name": "Google Pixel 5",
      "quantity": 8
    }
  ]
`;


let productsJS = JSON.parse(productsJSON);

function calculateTotalQuantity() {
  const totalQuantity = productsJS.reduce((total, item) => {
    return total + item.quantity;
  }, 0);
  return totalQuantity;
}

console.log(calculateTotalQuantity()); // 23
```