Lesson 02 Demo 15

Implementing Stack and Queue using Deque

Objective: To demonstrate the implementation of both stack and queue functionalities using a deque (double-ended queue) in JavaScript

Tools required: Visual Studio Code and Node.js

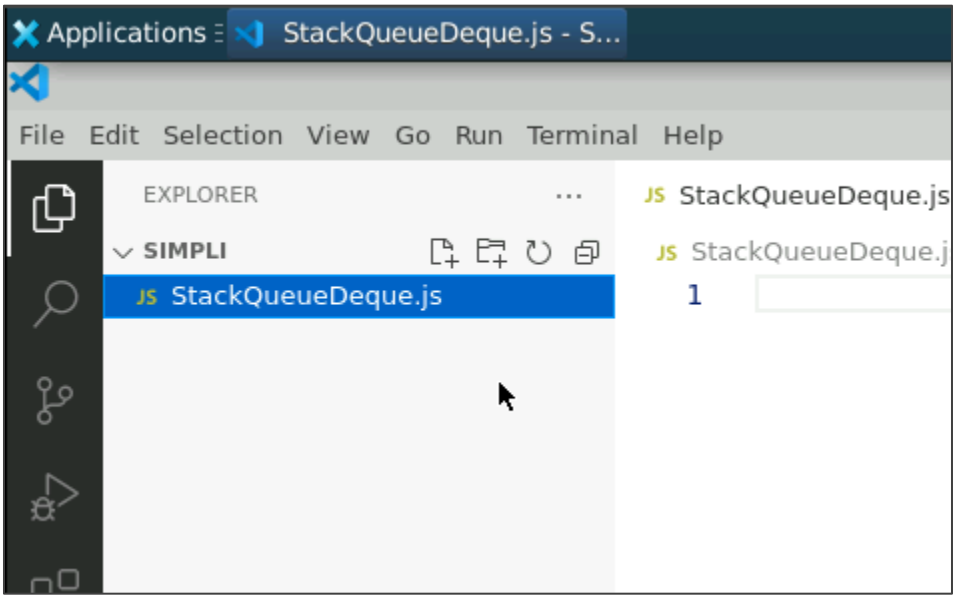
Prerequisites: Basic understanding of stack, queue, deque and JavaScript

Steps to be followed:

1. Create and execute the JS file

Step 1: Create and execute the JS file

1.1 Open the Visual Studio Code editor and create a JavaScript file named **StackQueueDeque.js**



1.2 Write the code given below in the **StackQueueDeque.js** file:

```
// Deque implementation
class Deque {
  constructor() {
    this.items = [];
  }

  // Methods for Stack implementation
  push(item) {
    this.items.push(item);
  }

  pop() {
    if (this.isEmpty()) {
      return undefined;
    }
    return this.items.pop();
  }

  // Methods for Queue implementation
  enqueue(item) {
    this.items.push(item);
  }

  dequeue() {
    if (this.isEmpty()) {
      return undefined;
    }
    return this.items.shift();
  }

  isEmpty() {
    return this.items.length === 0;
  }

  size() {
    return this.items.length;
  }
}
```

```
// Example usage
const stack = new Deque();
stack.push(1);
stack.push(2);
console.log('Stack pop:', stack.pop());

const queue = new Deque();
queue.enqueue(1);
queue.enqueue(2);
console.log('Queue dequeue:', queue.dequeue());
```

```
// Deque implementation
class Deque {
    constructor() {
        this.items = [];
    }

    // Methods for Stack implementation
    push(item) {
        this.items.push(item);
    }

    pop() {
        if (this.isEmpty()) {
            return undefined;
        }
        return this.items.pop();
    }

```

```
    // Methods for Queue implementation
    enqueue(item) {
        this.items.push(item);
    }

    dequeue() {
        if (this.isEmpty()) {
            return undefined;
        }
        return this.items.shift();
    }

    isEmpty() {
        return this.items.length === 0;
    }

    size() {
        return this.items.length;
    }
}

```

```
// Example usage
const stack = new Deque();
stack.push(1);
stack.push(2);
console.log('Stack pop:', stack.pop());

const queue = new Deque();
queue.enqueue(1);
queue.enqueue(2);
console.log('Queue dequeue:', queue.dequeue());

```

1.3 Save the file and execute it in the terminal using the command given below:
node StackQueueDeque.js

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$ ls
StackQueueDeque.js
priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$ node StackQueueDeque.js
Stack pop: 2
Queue dequeue: 1
priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$
```

This example illustrates the implementation of stack and queue using a deque in JavaScript.

By following these steps, you have successfully implemented stack and queue operations using a deque in JavaScript, broadening your understanding of versatile data structures and their applications in programming.