.

Lesson 02 Demo 10

Implementing CRUD Operations on a Stack

Objective: To implement CRUD operations on a stack, showcasing how to push, pop, peek, display, and clear elements within a stack structure, enhancing your ability to manipulate data structures and apply fundamental programming concepts in practical scenarios

Tools required: Visual Studio Code

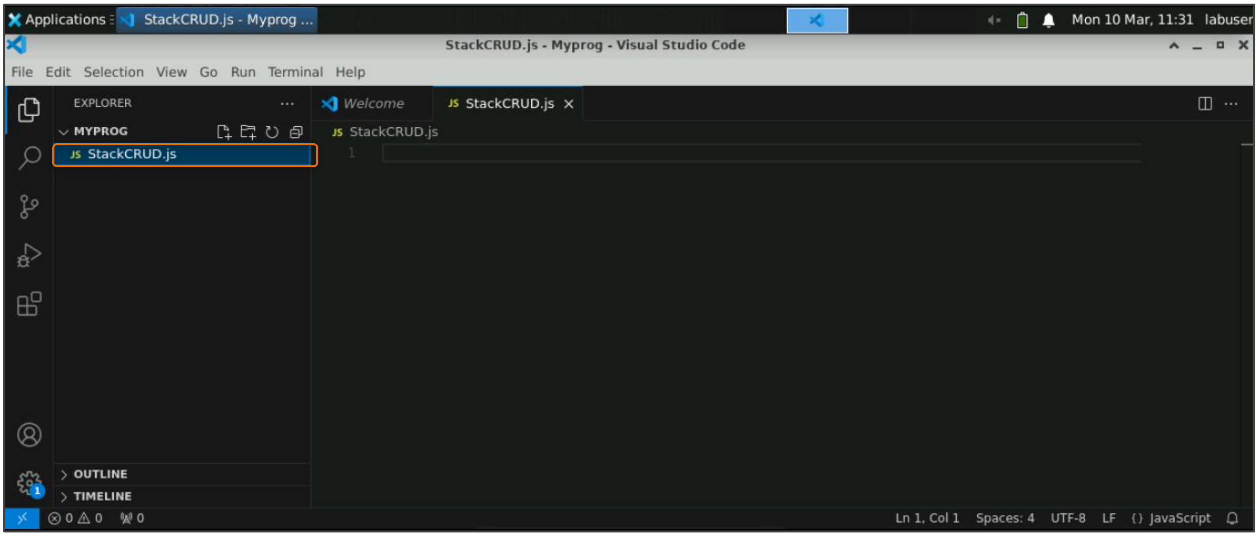
Prerequisites: A basic understanding of data structures and JavaScript

Steps to be followed:

1. Create a JavaScript file and execute it

Step 1: Create a JavaScript file and execute it

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

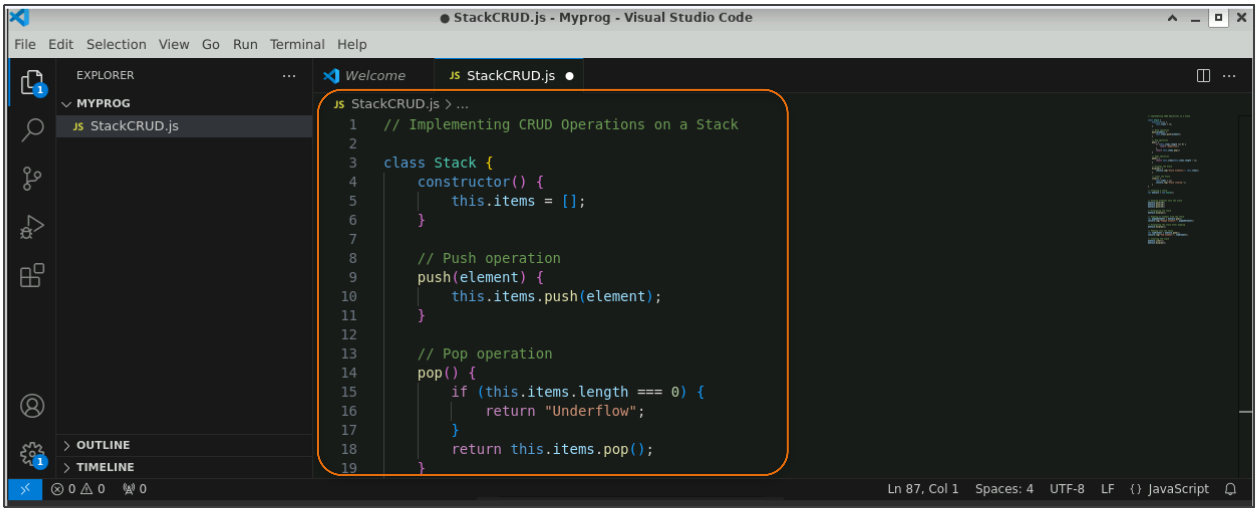

Applications
StackCRUD.js - Myprog ...
Mon 10 Mar, 11:31 labuser
StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
Welcome
StackCRUD.js
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

.

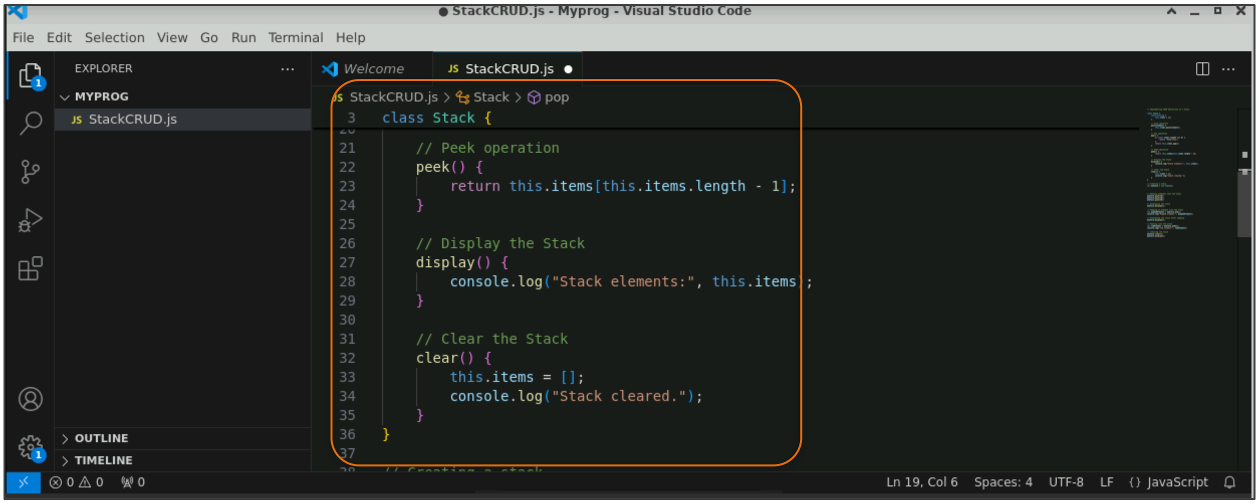
1.2 Add the following code to the file:

```
// Implementing CRUD Operations on a Stack

class Stack {
  constructor() {
    this.items = [];
  }

  // Push operation
  push(element) {
    this.items.push(element);
  }

  // Pop operation
  pop() {
    if (this.items.length === 0) {
      return "Underflow";
    }
    return this.items.pop();
  }

  // Peek operation
  peek() {
    return this.items[this.items.length - 1];
  }

  // Display the Stack
  display() {
    console.log("Stack elements:", this.items);
  }

  // Clear the Stack
  clear() {
    this.items = [];
    console.log("Stack cleared.");
  }
}

// Creating a stack
let myStack = new Stack();
```

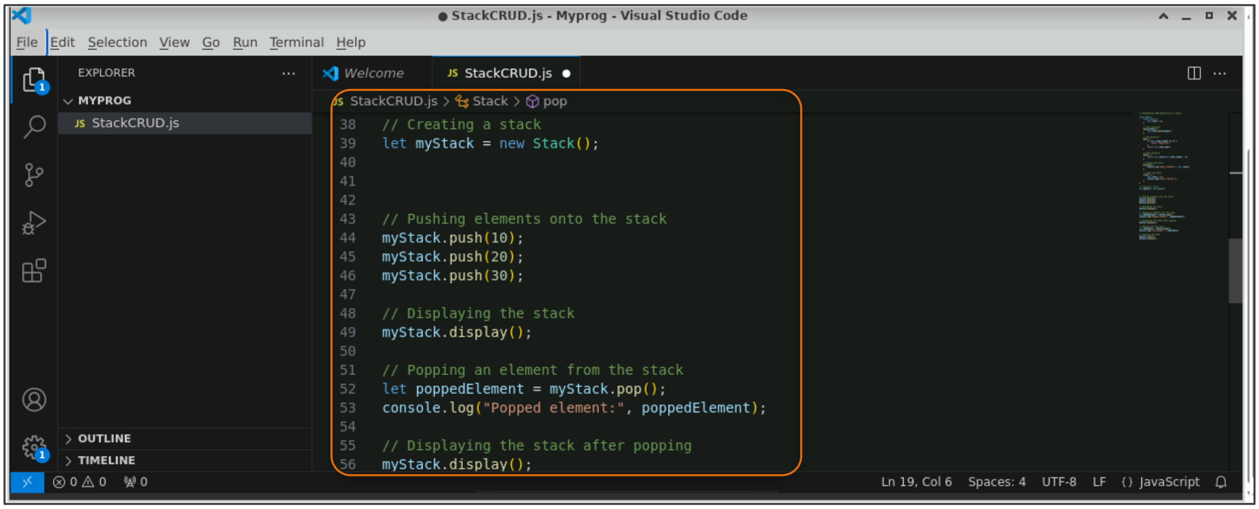
```
// Pushing elements onto the stack
myStack.push(10);
myStack.push(20);
myStack.push(30);

// Displaying the stack
myStack.display();

// Popping an element from the stack
let poppedElement = myStack.pop();
console.log("Popped element:", poppedElement);

// Displaying the stack after popping
myStack.display();

// Peeking into the stack
let topElement = myStack.peek();
console.log("Top element:", topElement);

// Clearing the stack
myStack.clear();
myStack.display();
```


● StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
OUTLINE
TIMELINE
Welcome
StackCRUD.js
StackCRUD.js > ...
Ln 87, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

```
// Implementing CRUD Operations on a Stack

class Stack {
    constructor() {
        this.items = [];
    }

    // Push operation
    push(element) {
        this.items.push(element);
    }

    // Pop operation
    pop() {
        if (this.items.length === 0) {
            return "Underflow";
        }
        return this.items.pop();
    }
```

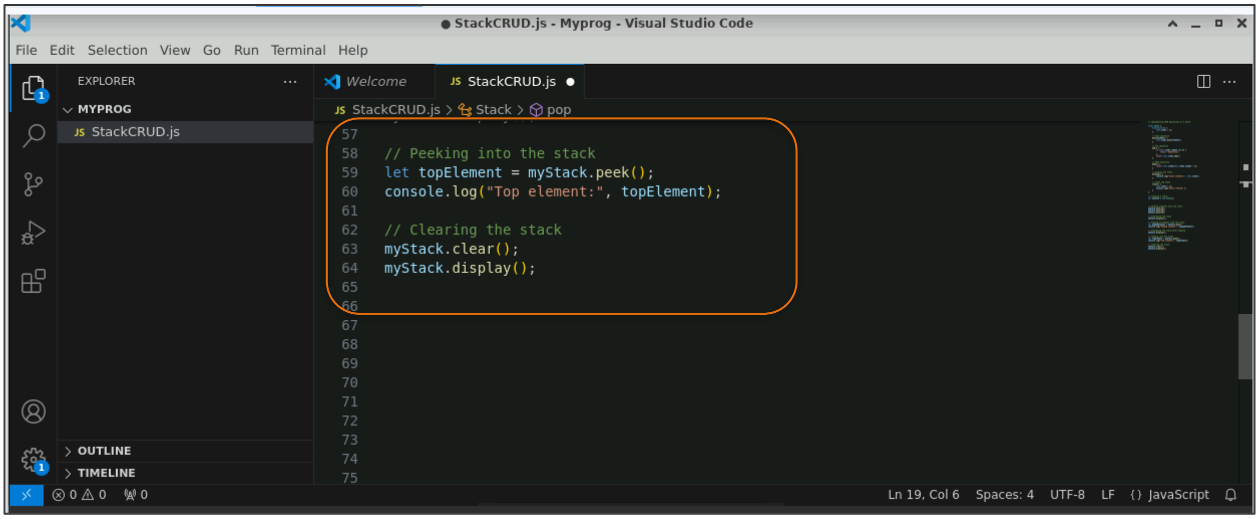

● StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
OUTLINE
TIMELINE
Welcome
StackCRUD.js
StackCRUD.js > Stack > pop
Ln 19, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

```
class Stack {
    // Peek operation
    peek() {
        return this.items[this.items.length - 1];
    }

    // Display the Stack
    display() {
        console.log("Stack elements:", this.items);
    }

    // Clear the Stack
    clear() {
        this.items = [];
        console.log("Stack cleared.");
    }
}

// Creating a stack
```


● StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
OUTLINE
TIMELINE
Welcome
StackCRUD.js
StackCRUD.js > Stack > pop
Ln 19, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

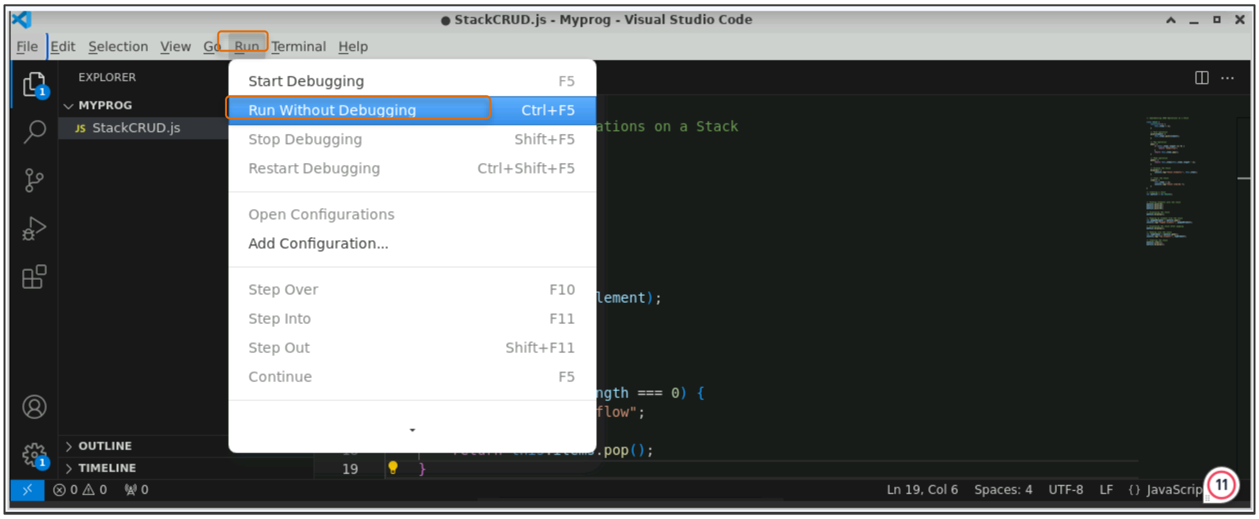
```
// Creating a stack
let myStack = new Stack();



// Pushing elements onto the stack
myStack.push(10);
myStack.push(20);
myStack.push(30);

// Displaying the stack
myStack.display();

// Popping an element from the stack
let poppedElement = myStack.pop();
console.log("Popped element:", poppedElement);

// Displaying the stack after popping
myStack.display();
```

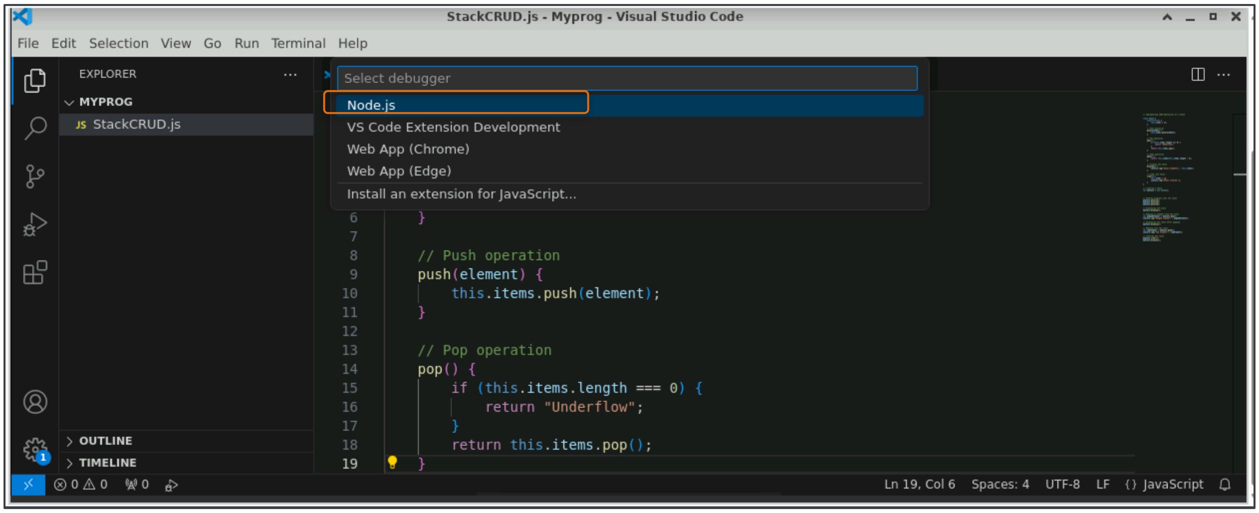
.


● StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
Welcome
StackCRUD.js ●
StackCRUD.js > Stack > pop
OUTLINE
TIMELINE
⊗ 0 ⚠ 0 0
Ln 19, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

```
// Peeking into the stack
let topElement = myStack.peek();
console.log("Top element:", topElement);

// Clearing the stack
myStack.clear();
myStack.display();
```

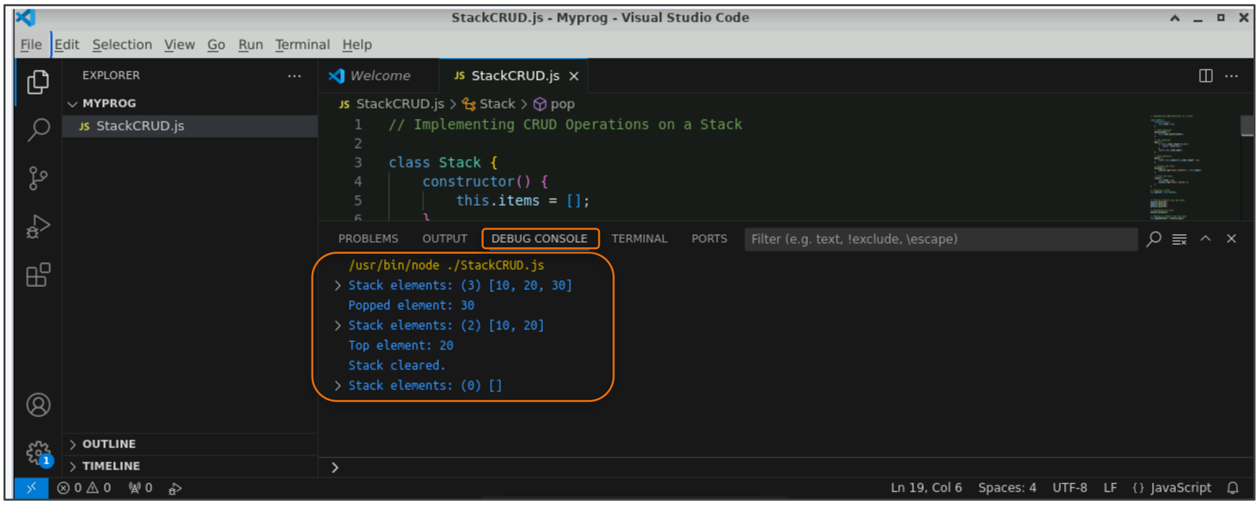
1.3 Click **Run** and then **Run Without Debugging**. Select **Node.js** to check the output in the DEBUG CONSOLE.




StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
Select debugger
Node.js
VS Code Extension Development
Web App (Chrome)
Web App (Edge)
Install an extension for JavaScript...
OUTLINE
TIMELINE
Ln 19, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

```
    }

    // Push operation
    push(element) {
        this.items.push(element);
    }

    // Pop operation
    pop() {
        if (this.items.length === 0) {
            return "Underflow";
        }
        return this.items.pop();
    }
```

1.4 View the output in the **DEBUG CONSOLE** as shown below:


StackCRUD.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
StackCRUD.js
Welcome
StackCRUD.js
StackCRUD.js > Stack > pop
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Filter (e.g. text, !exclude, \escape)
OUTLINE
TIMELINE
Ln 19, Col 6 Spaces: 4 UTF-8 LF {} JavaScript

```
// Implementing CRUD Operations on a Stack

class Stack {
    constructor() {
        this.items = [];
```

```
/usr/bin/node ./StackCRUD.js
> Stack elements: (3) [10, 20, 30]
  Popped element: 30
> Stack elements: (2) [10, 20]
  Top element: 20
  Stack cleared.
> Stack elements: (0) []
```

Note: This example illustrates CRUD operations on a stack, including pushing, popping, peeking, displaying, and clearing elements.

By following these steps, you have successfully implemented various CRUD operations a stack in JavaScript, enhancing your ability to manipulate data structures and apply fundamental programming concepts in practical scenarios.