LAB ASSIGNMENT NO – 8

Aim: Assignment on REPL (Commandline).

Theory:

Node.js REPL (Read-Eval-Print Loop) is an interactive command-line interface that allows users to execute JavaScript code and see immediate results. It's a tool for quickly testing and experimenting with JavaScript code snippets or running small scripts without the need to create a full-fledged file or application.

Here's a brief explanation:

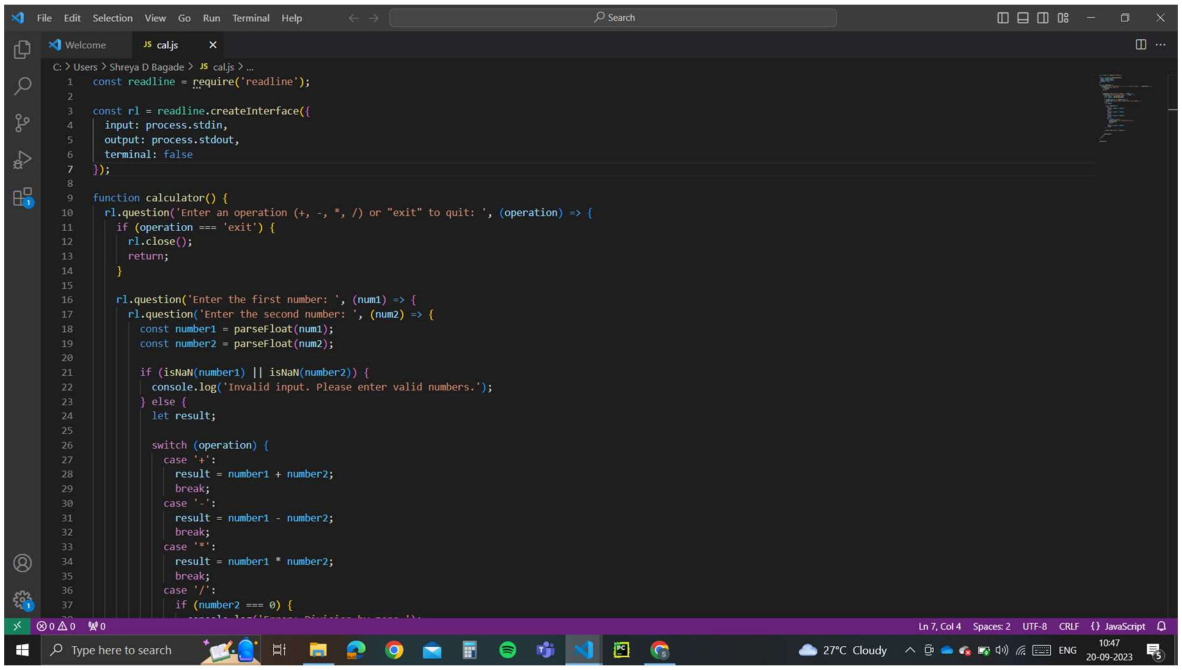
1. Read: The REPL reads JavaScript code entered by the user or from a file.
2. Eval: It evaluates the JavaScript code and executes it in a global context, which means variables and functions declared are available globally within the REPL session.
3. Print: The result of the evaluated expression is printed to the console.
4. Loop: The process repeats, allowing users to continuously input and execute code, providing an interactive development environment.

Developers use the Node.js REPL to test code, experiment with APIs, debug, and learn about JavaScript in an interactive and real-time manner, which enhances their understanding and efficiency in working with Node.js applications.

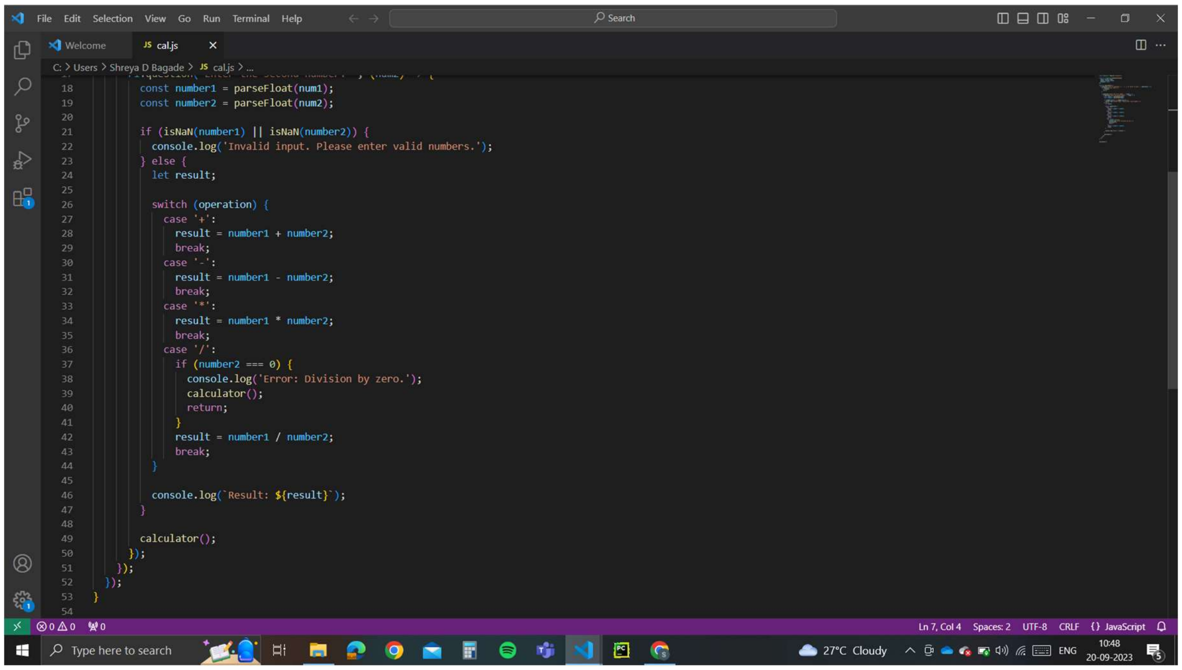
Conclusion: This experiment that is Node.js REPL serves as a versatile interactive tool for rapid prototyping and experimenting with JavaScript code. It allows developers to quickly test and validate ideas without the need for file-based setup. The immediate evaluation and feedback loop promote a highly efficient and iterative development process. Moreover, it aids in learning and exploring JavaScript features, making it an indispensable asset for developers utilizing Node.js.

Shreya Bagade/ SEM V/ TE.IT/ T11/ Roll No-09/ Assignment no-8

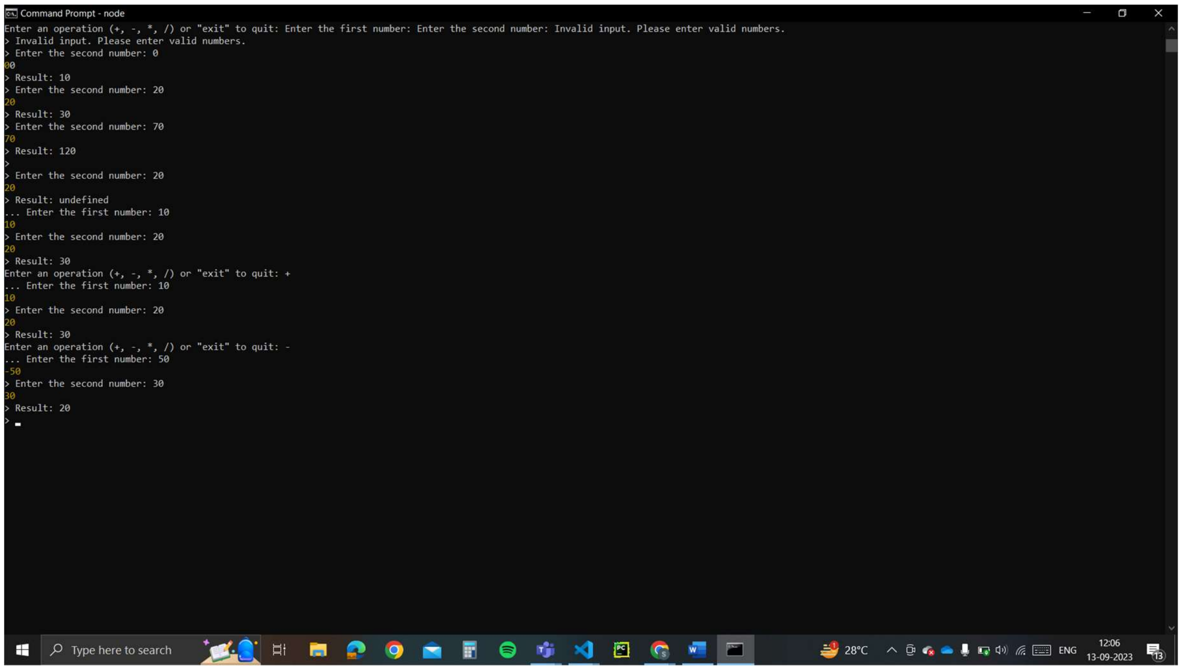
Code:



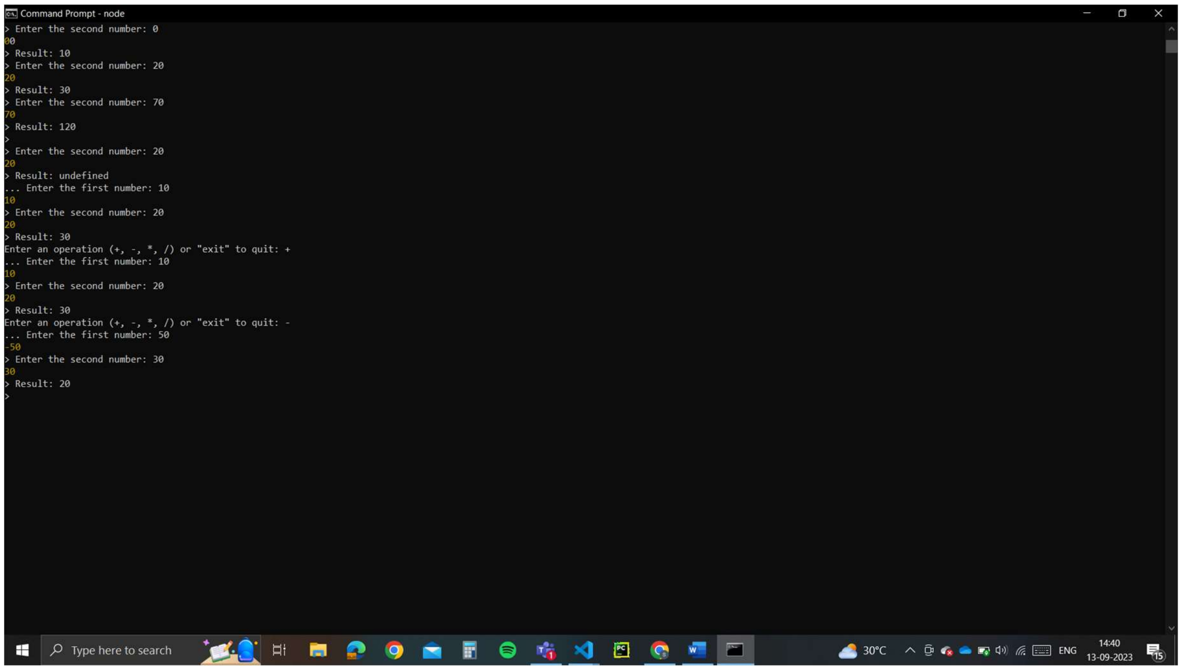
```
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
  terminal: false
});

function calculator() {
  rl.question('Enter an operation (+, -, *, /) or "exit" to quit: ', (operation) => {
    if (operation === 'exit') {
      rl.close();
      return;
    }

    rl.question('Enter the first number: ', (num1) => {
      rl.question('Enter the second number: ', (num2) => {
        const number1 = parseFloat(num1);
        const number2 = parseFloat(num2);

        if (isNaN(number1) || isNaN(number2)) {
          console.log('Invalid input. Please enter valid numbers.');
        } else {
          let result;

          switch (operation) {
            case '+':
              result = number1 + number2;
              break;
            case '-':
              result = number1 - number2;
              break;
            case '*':
              result = number1 * number2;
              break;
            case '/':
              if (number2 === 0) {
```



```
        const number1 = parseFloat(num1);
        const number2 = parseFloat(num2);

        if (isNaN(number1) || isNaN(number2)) {
          console.log('Invalid input. Please enter valid numbers.');
        } else {
          let result;

          switch (operation) {
            case '+':
              result = number1 + number2;
              break;
            case '-':
              result = number1 - number2;
              break;
            case '*':
              result = number1 * number2;
              break;
            case '/':
              if (number2 === 0) {
                console.log('Error: Division by zero.');
                calculator();
                return;
              }
              result = number1 / number2;
              break;
          }

          console.log(`Result: ${result}`);
        }

        calculator();
      });
    });
  });
}
```



```
Enter an operation (+, -, *, /) or "exit" to quit: Enter the first number: Enter the second number: Invalid input. Please enter valid numbers.
> Invalid input. Please enter valid numbers.
> Enter the second number: 0
00
> Result: 10
> Enter the second number: 20
20
> Result: 30
> Enter the second number: 70
70
> Result: 120
>
> Enter the second number: 20
20
> Result: undefined
... Enter the first number: 10
10
> Enter the second number: 20
20
> Result: 30
Enter an operation (+, -, *, /) or "exit" to quit: +
... Enter the first number: 10
10
> Enter the second number: 20
20
> Result: 30
Enter an operation (+, -, *, /) or "exit" to quit: -
... Enter the first number: 50
-50
> Enter the second number: 30
30
> Result: 20
>
```



```
> Enter the second number: 0
00
> Result: 10
> Enter the second number: 20
20
> Result: 30
> Enter the second number: 70
70
> Result: 120
>
> Enter the second number: 20
20
> Result: undefined
... Enter the first number: 10
10
> Enter the second number: 20
20
> Result: 30
Enter an operation (+, -, *, /) or "exit" to quit: +
... Enter the first number: 10
10
> Enter the second number: 20
20
> Result: 30
Enter an operation (+, -, *, /) or "exit" to quit: -
... Enter the first number: 50
-50
> Enter the second number: 30
30
> Result: 20
>
```

Lab Outcome : LO6- Construct back end applications using Node.js/Express.