Hands-on Activity 2.1 | Operators, User Ineraction, and Control Flow	
Course Code: CPE 026	**Program:** Computer Engineering
Course Title: Emerging Technologies 3 in CpE	**Date Performed: 08/22/2024**
Section: BSCPE41S8	**Date Submitted: 08/28/2024**
Name: Christian Ed B. Efa	**Instructor:** Engr. Roman Richard

1. Discussion

Discuss here the relevant concepts of the activity in your own words.

Module Section 2.0

- **tackles about Variables, Naming the variables, Declaring variables, Initializing variables, Declarations and strict mode, Changing variable values, Constants, Scope, A brief word about functions, The var keyword continued, Variable shadowing, Hoisting. That is Using variables, in other words, declaring, initializing, changing, or reading their values is an elementary part of practically every programming language.**

Module 2.1 & 2.2 Section 2-3

- **Discussion of simple types, The number BigInt, or Boolean types are not by chance called primitive, Learning what autoboxing on how a primitive of an object is related and how we can use methods related to string object.**

Module 2.3 Section 4

- **Comments give the opportunity to increase clarity of the code by adding information that will help others to understand selected parts of it. Comments are also very useful for turning selected pieces of code on and off, which we use most often when testing alternative versions or when searching for errors in it.**

2. Materials and Equipment

What materials did you use? Explain in detail.

- **Netacad**

Logging in the netacad and opening the course JSE (JavaScript Essentials) and using the Edube Interactive as my learning module and also my IDE .

3. Procedure

What are the procedures that you performed?

```
1  var height;
2  console.log(height);
3  console.log(weight);
```

- The first line is the variable declaration (we can see the var keyword). This declaration means that the word height will be treated as the name of the container for certain values.

```
let height;
console.log(height);
```

- This variable are using let instead of var

```
var height;
var height;
console.log(height); // -> undefined
```

- basic differences in the use of var and let is that let prevents us from declaring another variable with the same name (an error is generated). Using var allows you to re-declare a variable, which can potentially lead to errors in the program execution.

```
let height;
let height; // -> Uncau
console.log(height);
```

- use let to declare variables, if only because you don't want to accidentally declare a variable again.

2.1.1.19 Variables - Tasks

```
let rosePrice = 8;
let lilyPrice = 10;
let tulipPrice = 2;

let numberOfRoses = 70;
let numberOfLilies = 50;
let numberOfTulips = 120;

let rosesValue = rosePrice * numberOfRoses;
let liliesValue = lilyPrice * numberOfLilies;
let tulipsValue = tulipPrice * numberOfTulips;

let total = rosesValue + liliesValue + tulipsValue;
console.log("Rose - unit price:", rosePrice, ", quantity:", numberOfRoses, ", value:", rosesValue);
console.log("Lily - unit price:", lilyPrice, ", quantity:", numberOfLilies, ", value:", liliesValue);
console.log("Tulip - unit price:", tulipPrice, ", quantity:", numberOfTulips, ", value:", tulipsValue);
console.log("Total: ", total);
```

```
const rosePrice = 8;
const lilyPrice = 10;
const tulipPrice = 2;

let numberOfRoses = 70;
let numberOfLilies = 50;
let numberOfTulips = 120;

let rosesValue = rosePrice * numberOfRoses;
let liliesValue = lilyPrice * numberOfLilies;
let tulipsValue = tulipPrice * numberOfTulips;

let total = rosesValue + liliesValue + tulipsValue;

console.log("Rose - unit price:", rosePrice, ", quantity:", numberOfRoses, ", value:", rosesValue);
console.log("Lily - unit price:", lilyPrice, ", quantity:", numberOfLilies, ", value:", liliesValue);
console.log("Tulip - unit price:", tulipPrice, ", quantity:", numberOfTulips, ", value:", tulipsValue);
console.log("Total: ", total);

numberOfRoses = numberOfRoses - 20;
numberOfLilies = numberOfLilies - 30;

rosesValue = rosePrice * numberOfRoses;
liliesValue = lilyPrice * numberOfLilies;
tulipsValue = tulipPrice * numberOfTulips;

total = rosesValue + liliesValue + tulipsValue;

console.log("Rose - unit price:", rosePrice, ", quantity:", numberOfRoses, ", value:", rosesValue);
console.log("Lily - unit price:", lilyPrice, ", quantity:", numberOfLilies, ", value:", liliesValue);
console.log("Tulip - unit price:", tulipPrice, ", quantity:", numberOfTulips, ", value:", tulipsValue);
console.log("Total: ", total);
```

- This task shows us how to use variables whereis this program calculate and show the quantity and price of each flowers

2.2.1.13 Data types

```
let someResource;
console.log(someResource); // -> undefined
console.log(typeof someResource); // -> undefined

someResource = null;
console.log(someResource); // -> null
console.log(typeof someResource); // -> object
```

- the undefined value is assigned to uninitialized variables automatically. One important caveat for null is that when checked with the typeof operator, it will return "object".

2.2.1.14 Data types

```
const str = String();
const num = Number();
const bool = Boolean();

console.log(str); // ->
console.log(num); // -> 0
console.log(bool); // -> false

const big1 = BigInt(42);
console.log(big1); // -> 42n

const big2 = BigInt();   // -> Uncaught TypeError: Cannot convert undefined to a BigInt
```

- Using literals is not the only way to create variables of the given primitive types. The second option is to make them using constructor functions.
- the function String will by default create and return an empty string – primitive "";
- the function Number will by default create and return the value 0;
- the function Boolean will by default create and return the value of false.

2.2.1.15 Data types

```
Const num = 42;

const strFromNum1 = String(num);
const strFromNum2 = String(8);
const strFromBool = String(true);
const numFromStr = Number("312");
const boolFromNumber = Boolean(0);
```

2.2.1.16 Data types - Type conversions

```
let str = "text";
let strStr = String(str);
console.log(`${typeof str} : ${str}`); // -> string : text
console.log(`${typeof strStr} : ${strStr}`); // -> string : text

let nr = 42;
let strNr = String(nr);
console.log(`${typeof nr} : ${nr}`); // -> number : 42
console.log(`${typeof strNr} : ${strNr}`); // -> string : 42

let bl = true;
let strBl = String(bl);
console.log(`${typeof bl} : ${bl}`); // -> boolean : true
console.log(`${typeof strBl} : ${strBl}`); // -> string : true

let bnr = 123n;
let strBnr = String(bnr);
console.log(`${typeof bnr} : ${bnr}`); // -> bigint : 123
console.log(`${typeof strBnr} : ${strBnr}`); // -> string : 123

let un = undefined;
let strUn = String(un);
console.log(`${typeof un} : ${un}`); // -> undefined : undefined
console.log(`${typeof strUn} : ${strUn}`); // -> string : undefined

let n = null;
let strN = String(n);
console.log(`${typeof n} : ${n}`); // -> object : null
console.log(`${typeof strN} : ${strN}`); // -> string : null
```

- to directly change the value to a string, can be done for all primitive types. we used the discussed technique of character string interpolation.

2.3.1.19 Objects and arrays - Tasks

```
let ticket = {
    from: "TIP QC",
    to: "TIP MANILA",
    price: 5000
};
console.log(`Ticket from: ${ticket.from}`);
console.log(`Ticket to: ${ticket.to}`);
console.log(`Ticket price: ${ticket.price}`);
```

- Using an object ticket that has from,to, and price. Inside the object and displaying its value.

```
let person = {};
person.name = "Mary";
person.surname = "Stuart";
console.log(`${person.name} ${person.surname}`);
```

- Same as the ticket object but using a person name and displaying it.

```
let books = [
    {
        title: "Speaking JavaScript",
        author: "Axel Rauschmayer",
        pages: 460
    },
    {
        title: "Programming JavaScript Applications",
        author: "Eric Elliott",
        pages: 254
    },
    {
        title: "Understanding ECMAScript 6",
        author: "Nicholas C. Zakas",
        pages: 352
    }
];

books.forEach(book => {
    console.log(`Title: ${book.title}, Author: ${book.author}, Pages: ${book.pages}`);
});
```

- This task was Creating an array of three objects representing the books. Each object must have the following properties: title, author, pages. And trying to display it all.

```
let books = [{
        title: "Speaking JavaScript",
        author: "Axel Rauschmayer",
        pages: 460
    },
    {
        title: "Programming JavaScript Applications",
        author: "Eric Elliot",
        pages: 254
    },
    {
        title: "Understanding ECMAScript 6",
        author: "Nicholas C. Zakas",
        pages: 352
    }
];
let newBook = {
    title: "Learning JavaScript Design Patterns",
    author: "Addy Osmani",
    pages: 254
};
books.push(newBook);
console.log(books.length);
console.log(books[0].title);
console.log(books[1].title);
console.log(books[2].title);
console.log(books[3].title);
```

- Adding a newbook with the previous array.

```

let selectedBooks = books.slice(-2);
```

- slice command to copy the last two books to the new array.

```
30  books.shift();
31  console.log(books.length);
32  console.log(books[0].title);
33  console.log(books[1].title);
34  console.log(books[2].title);
```

- Displaying the length of the array and all the names of the books from the collection in turn.

```
29  let selectedBooks = books.slice(-2);
30  books.shift();
31  console.log(books.length);
32  console.log(books[0].title);
33  console.log(books[1].title);
34  console.log(books[2].title);
35
36  let sum = books[0].pages + books[1].pages + books[2].pages;
37  console.log(`pages: ${sum}`);
```

- Display the sum of the pages of all the books from the collection.

2.4.1.6 Comments - Tasks

```
"use strict";

const prefix = "username_";

let userName = "Jack";
const userName = "Adam";

let prefixedUserName;
const prefixedUserName;

userName = "John";
prefixedUserName = prefix + userName;

console.log(prefixedUserName + prefixedUserName2);
console.log(prefixedUserName2);
```

```
1   "use strict";
2
3   const prefix = "username_";
4
5   let userName = "Jack";
6
7
8   let prefixedUserName;
9
10
11  userName = "John";
12  prefixedUserName = prefix + userName;
13
14  console.log(prefixedUserName);
```

- Debugging this by removing the non existing names such as const and prefixed usernames.

3.1.1.12 Operators - Tasks

```
console.log(2 * 3 + 1);  // expected 7
console.log(2 ** 4);  // expected 16
console.log(5 * 1); // expected 5
console.log((8 * 5) + (2 - 2)-1);  // expected 39
```

- This task uses arithmetic operator to solve the problem and give the expected answers.

```
console.log(4 * 5 == 20);
console.log(6 * 5 == "30");
console.log(-17 != 0);
console.log(25 != 1);
console.log(2 + 2 * 2 !== 4);
```

- Using == equal and != not equal to display if the statement is true.

```
console.log(true !== false);
console.log(false == false);
console.log(false == false == true);
console.log(true == false == false && true);
```

- Same as this task using the logical operators.

Precedence

Practically in all the examples where we presented the operation of successive operators, we followed instructions in which one operator was used. In reality, usually multiple operators are used simultaneously. At this point, a quite important question arises: in what order will the interpreter perform them? This will of course affect the final result of the operators, so it is worth taking this into account when writing the instructions.

```
let a = 10;
let b = a + 2 * 3;
let c = a + 2  < 20 - 15;
console.log(a); // -> 10
console.log(b); // -> 16
console.log(c); // -> false
```

- Precedence determines the order in which operations are performed in expressions.

3.3.1.7 Interaction with the user - Tasks

```
let width = prompt("Volume of the box, enter width", 0);
let height = prompt("Volume of the box, enter height", 0);
let length = prompt("Volume of the box, enter length", 0);
let volume = width * height * length;
alert(`Calculated box volume is ${volume}`);
```

- Using prompt to input the width that is 20, height = 10 and the length of 50. To get or get the volume of 10000

3.3.1.8 LAB: Interaction

```
}, {
name: "Helen Richards",
phone: "0800 1111",
email: "libero@convallis.edu"
}];

// write your code here
let name = prompt("Enter the name");
let phone = prompt("Enter the phone");
let email = prompt("Enter the email");

let newContact = {
    name: name,
    phone: phone,
    email: email

};
contacts.push(newContact);

let last = contacts.length - 1;

console.log('${contacts[0].name} / ${contacts[0].phone} / ${contacts[0].email}');
console.log('${contacts[last].name} / ${contacts[last].phone} / ${contacts[last].email}');
```

- Adding a new contact and using the push to add a new inserted contact.

4.1.1.11 Conditional - Tasks

« 4.1.1.11 Conditional - Tasks »

```
let number = prompt("Enter a random number");
if(number > 90 && number < 110) {
    alert("Bingo!");
} else {
    alert("Miss");
}
```

- This task will display using if else statement depending on the inputted number if below 90 = false/miss else above 90 = bingo!

```
let number = prompt("Enter a random number");
let message = (number > 90 && number < 110) ? "Bingo!": "Miss";
alert(message);
```

- Same concept as if else statement, however using ternary conditional operator.

```
let firstNumber = Number(prompt("Enter first number"));
let secondNumber = Number(prompt("Enter second number"));
let operand = prompt("Enter operand (+, -, * or /)");
let result;

if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
    switch (operand) {
        case "+": result = firstNumber + secondNumber; break;
        case "-": result = firstNumber - secondNumber; break;
        case "*": result = firstNumber * secondNumber; break;
        case "/": result = firstNumber / secondNumber; break;
        default: result = "Error: unknown operand";
    }
} else {
    result = "Error: at least one of the entered values is not a number";
}
alert(result);
```

- This code is showing an simple calculator that add,subtract,multiply, and divide the number inputted by the user, if the number or operator is not recognize it will display an error alert.

4.2.1.15 Loops - Tasks

```
for (i=100; i>=0; i-=10) {
    console.log(i);
}
```

- Using for loops to write numbers from 100 by 10's

```
let upperLimit = Number(prompt("Enter upper limit"));
let lowerLimit = Number(prompt("Enter lower limit"));

if (!Number.isNaN(upperLimit) && !Number.isNaN(lowerLimit) && upperLimit > lowerLimit) {
    for (i = upperLimit; i >= lowerLimit; i -= 10) {
        console.log(i);
    }
}
```

- On this task we are using prompt command, the initial value is greater than the final value

```
let numbers = [21, 45, 100, 12, 11, 78, 61, 4, 39, 22];

for (number of numbers) {
    console.log(number);
}

for (number of numbers) {
    if (number % 2 === 0) {
        console.log(number);
    }
}

for (number of numbers) {
    if (number > 10 && number < 60) {
        console.log(number);
    }
}
```

- Using for loop to iterate through the array and print different subsets of values based on specific conditions.

```
let movies = [];
while (true) {
    let title = prompt("Enter movie title");
    let rating = prompt("Enter movie rating (imdb)");

    if (title === null || rating === null) {
        break
    } else {
        movies.push({
            title: title,
            rating: Number(rating)
        });
    }
}

console.log("All with ratings under 7:");
for (movie of movies) {
    if (movie.rating < 7) {
        console.log('${movie.title} (${movie.rating})');
    }
}

console.log("All with ratings over 7:");
for (movie of movies) {
    if (movie.rating >= 7) {
        console.log('${movie.title} (${movie.rating})');
        break;
    }
}
```

- This loop is asking for movie title and rating, it will not stop until i pressed cancel.the program should first print out to the console all movies that have a rating of less than 7, then those whose rating is greater than or equal to 7. Write the name of the movie and its rating next to each other.

```
let vessel = {
    LATITUDE: 40.07288,
    LONGITUDE: 154.48535,
    COURSE: 285.6,
    SPEED: 14.0,
    IMO: 9176717,
    NAME: "MARENO"
}

for (let key in vessel) {
    console.log('${key} -> ${vessel[key]}');
}
```

- object vessel that holds various properties representing information about a vessel, such as its latitude, longitude, speed, and name. The code then uses a for...in loop to iterate through the keys of the object and print each key-value pair to the console.

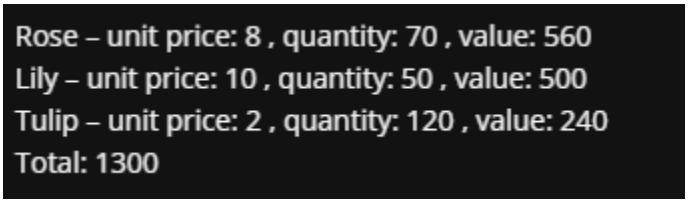
```
while (true) {
    let firstNumber = prompt("Enter first number");
    let secondNumber = prompt("Enter second number");
    let operand = prompt("Enter operand (+, -, * or /)");
    let result;

    if (firstNumber === "Q" || secondNumber === "Q" || operand === "Q") {
        break;
    }

    firstNumber = Number(firstNumber);
    secondNumber = Number(secondNumber);

    if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
        switch (operand) {
            case "+":
                result = firstNumber + secondNumber;
                break;
            case "-":
                result = firstNumber - secondNumber;
                break;
            case "*":
                result = firstNumber * secondNumber;
                break;
            case "/":
                result = firstNumber / secondNumber;
                break;
            default:
                result = "Error: unknown operand";
        }
    } else {
        result = "Error: at least one of the entered values is not a number";
    }
    alert(result);
}
```

- an interactive calculator program that continuously prompts the user for two numbers and an arithmetic operator, performs the specified operation, and displays the result. The loop continues until the user enters "Q" to quit.

4. Output

Screenshot of your outputs based on the procedures.

-2.1.1.19 Variables - Tasks



```
Rose – unit price: 8 , quantity: 50 , value: 400
Lily – unit price: 10 , quantity: 20 , value: 200
Tulip – unit price: 2 , quantity: 120 , value: 240
Total: 840
```

2.2.1.13 Data types

```
Console >_
undefined
undefined
object
```

2.2.1.14 Data types

```
Console >_
0
false
42
Uncaught TypeError: Cannot convert undefined to a BigInt
```

2.2.1.16 Data types - Type conversions

```
Console >_
string : text
string : text
number : 42
string : 42
boolean : true
string : true
bigint : 123
string : 123
undefined : undefined
string : undefined
object : null
string : null
```

2.3.1.19 Objects and arrays - Tasks

```
Console >_
Ticket from: TIP QC
Ticket to: TIP MANILA
Ticket price: 5000
Console >_
Mary Stuart
```

Console >_

Title: Speaking JavaScript, Author: Axel Rauschmayer, Pages: 460
Title: Programming JavaScript Applications, Author: Eric Elliott, Pages: 254
Title: Understanding ECMAScript 6, Author: Nicholas C. Zakas, Pages: 352

Console >_

4
Speaking JavaScript
Programming JavaScript Applications
Understanding ECMAScript 6
Learning JavaScript Design Patterns

3
Programming JavaScript Applications
Understanding ECMAScript 6
Learning JavaScript Design Patterns

4
Speaking JavaScript
Programming JavaScript Applications
Understanding ECMAScript 6
Learning JavaScript Design Patterns
3
Programming JavaScript Applications
Understanding ECMAScript 6
Learning JavaScript Design Patterns
pages: 860

Console >_

username_John

3.1.1.12 Operators - Tasks

app.js

Console >_

7
16
5
39

app.js

Console >_

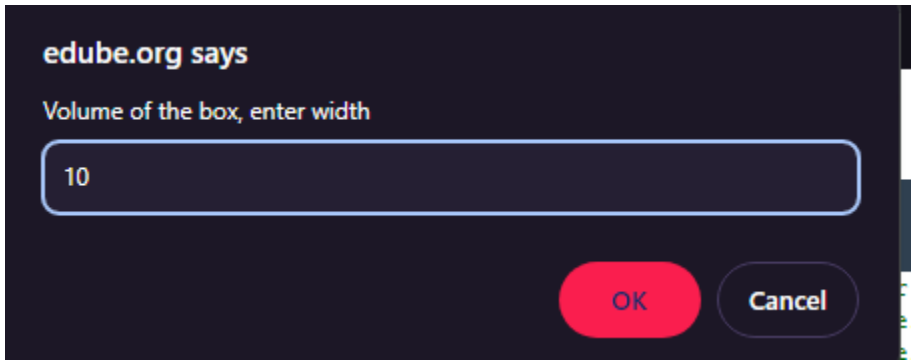
```
true
true
true
true
true
```

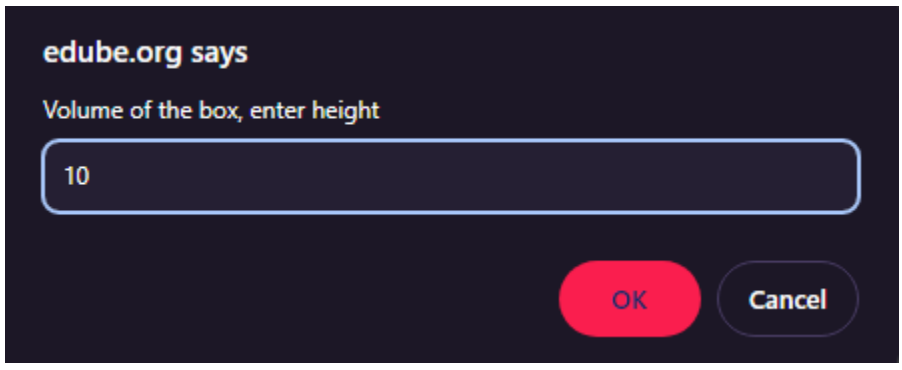
app.js

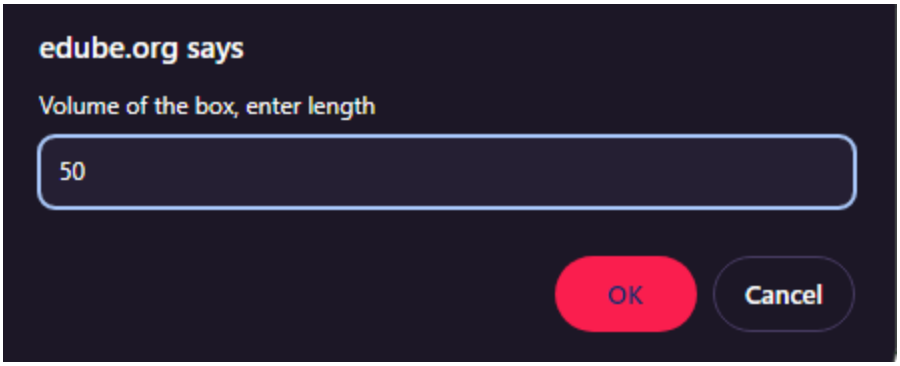
Console >_

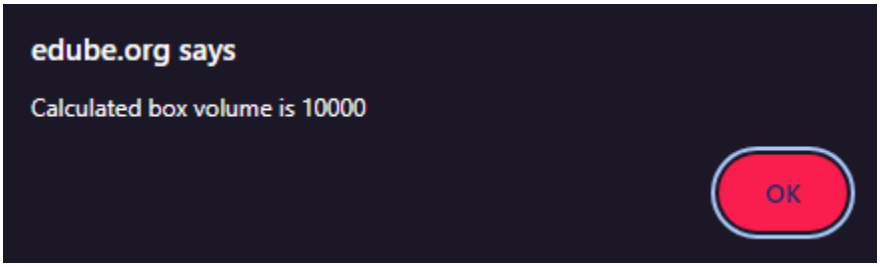
```
true
true
true
true
```

3.3.1.7 Interaction with the user - Tasks

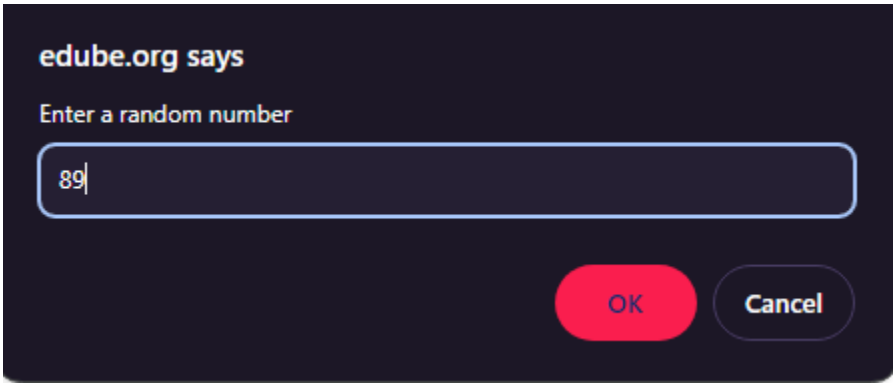


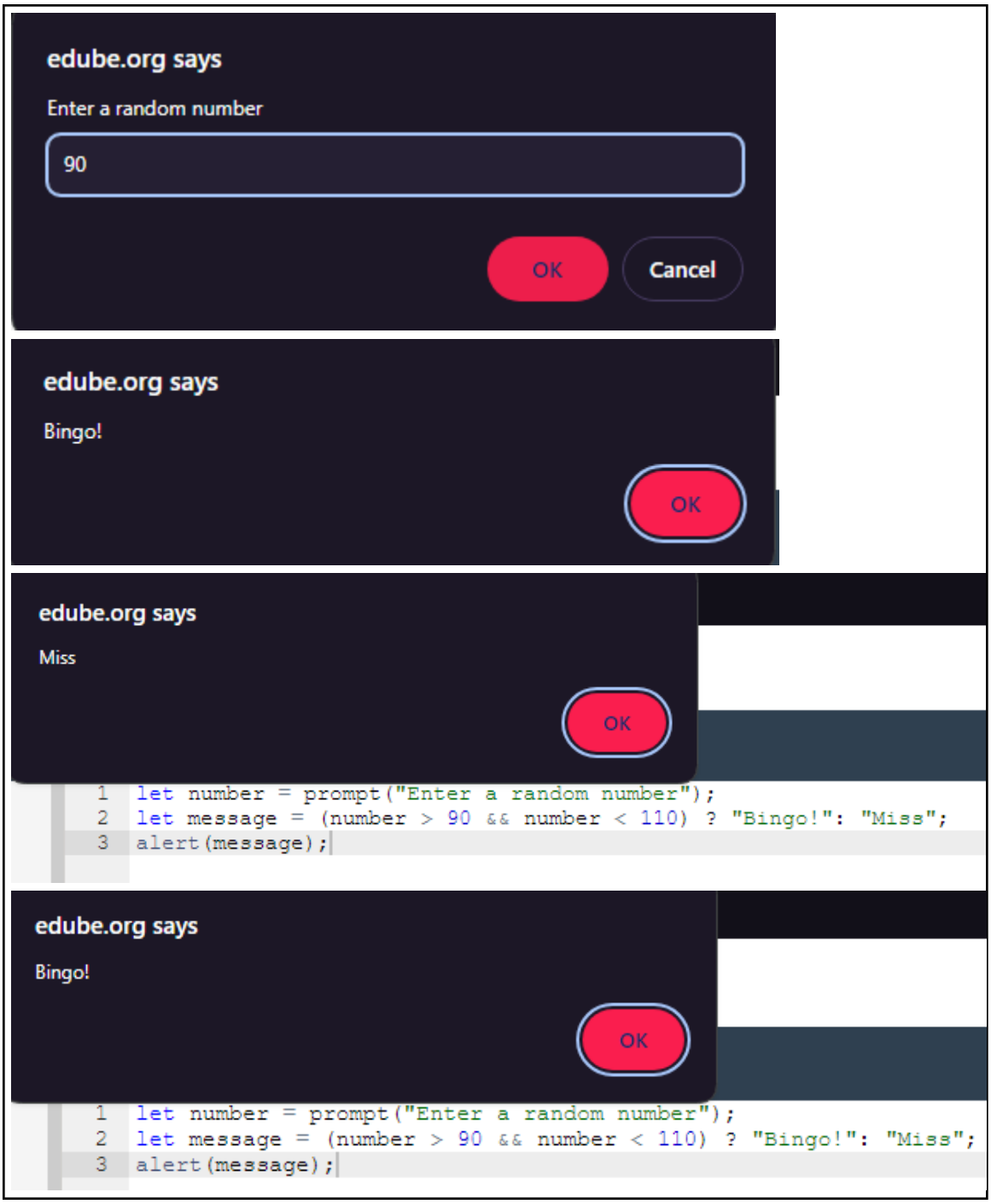


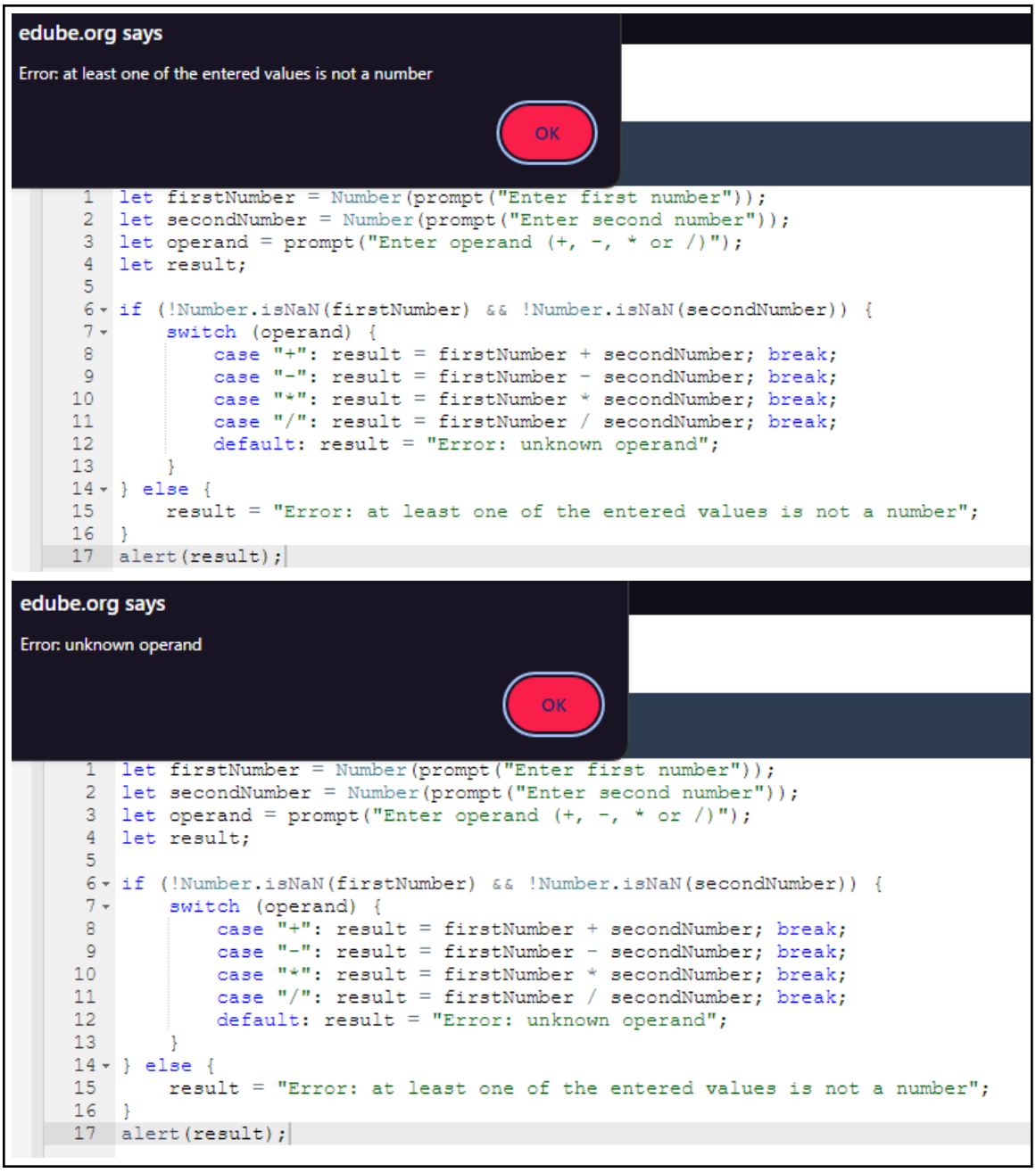




4.1.1.11 Conditional - Tasks

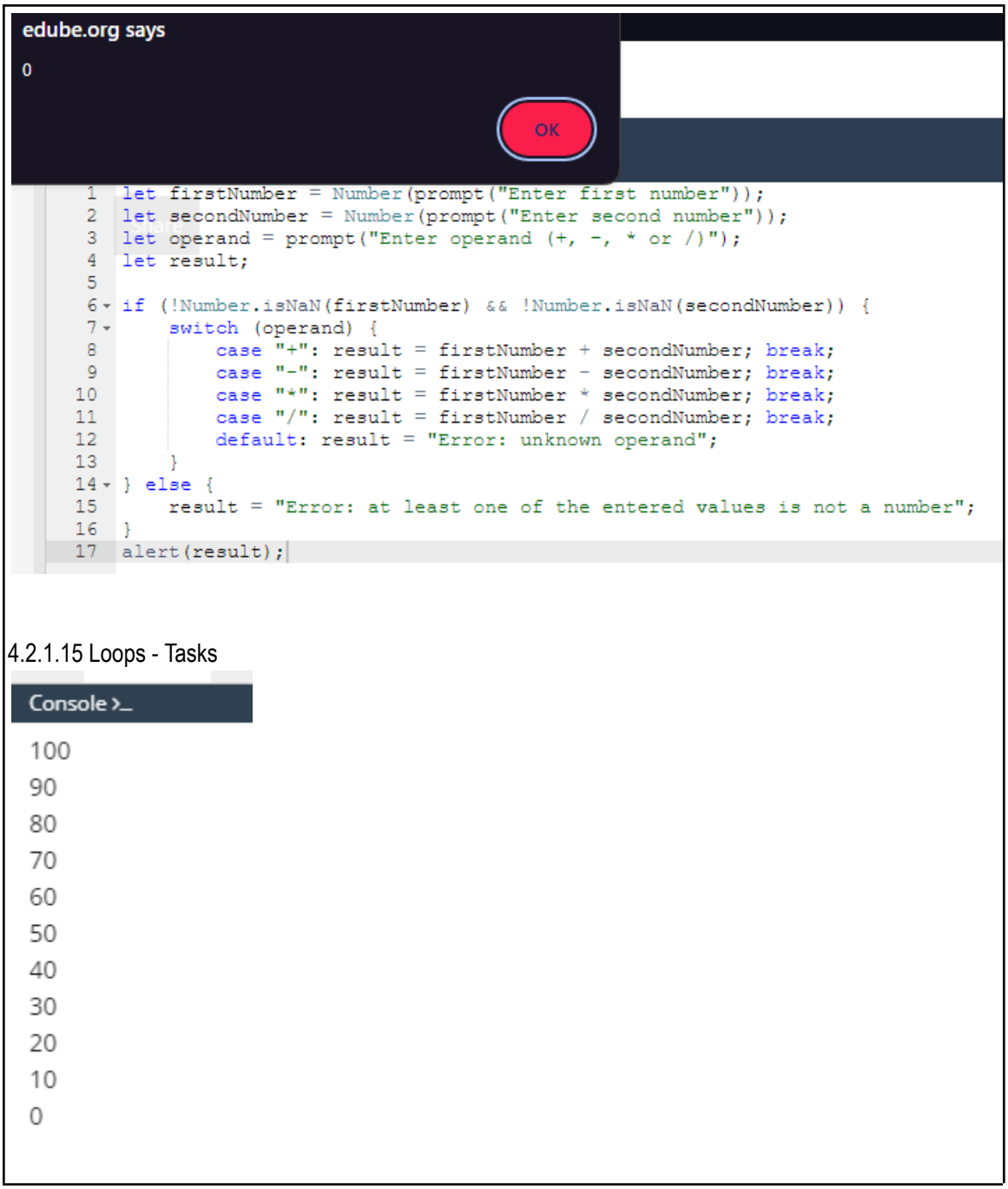



edube.org says
Enter a random number
90
OK
Cancel
edube.org says
Bingo!
OK
edube.org says
Miss
OK
1 let number = prompt("Enter a random number");
2 let message = (number > 90 && number < 110) ? "Bingo!": "Miss";
3 alert(message);
edube.org says
Bingo!
OK
1 let number = prompt("Enter a random number");
2 let message = (number > 90 && number < 110) ? "Bingo!": "Miss";
3 alert(message);




```
let firstNumber = Number(prompt("Enter first number"));
let secondNumber = Number(prompt("Enter second number"));
let operand = prompt("Enter operand (+, -, * or /)");
let result;

if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
    switch (operand) {
        case "+": result = firstNumber + secondNumber; break;
        case "-": result = firstNumber - secondNumber; break;
        case "*": result = firstNumber * secondNumber; break;
        case "/": result = firstNumber / secondNumber; break;
        default: result = "Error: unknown operand";
    }
} else {
    result = "Error: at least one of the entered values is not a number";
}
alert(result);
```

```
let firstNumber = Number(prompt("Enter first number"));
let secondNumber = Number(prompt("Enter second number"));
let operand = prompt("Enter operand (+, -, * or /)");
let result;

if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
    switch (operand) {
        case "+": result = firstNumber + secondNumber; break;
        case "-": result = firstNumber - secondNumber; break;
        case "*": result = firstNumber * secondNumber; break;
        case "/": result = firstNumber / secondNumber; break;
        default: result = "Error: unknown operand";
    }
} else {
    result = "Error: at least one of the entered values is not a number";
}
alert(result);
```



4.2.1.15 Loops - Tasks

Console >_

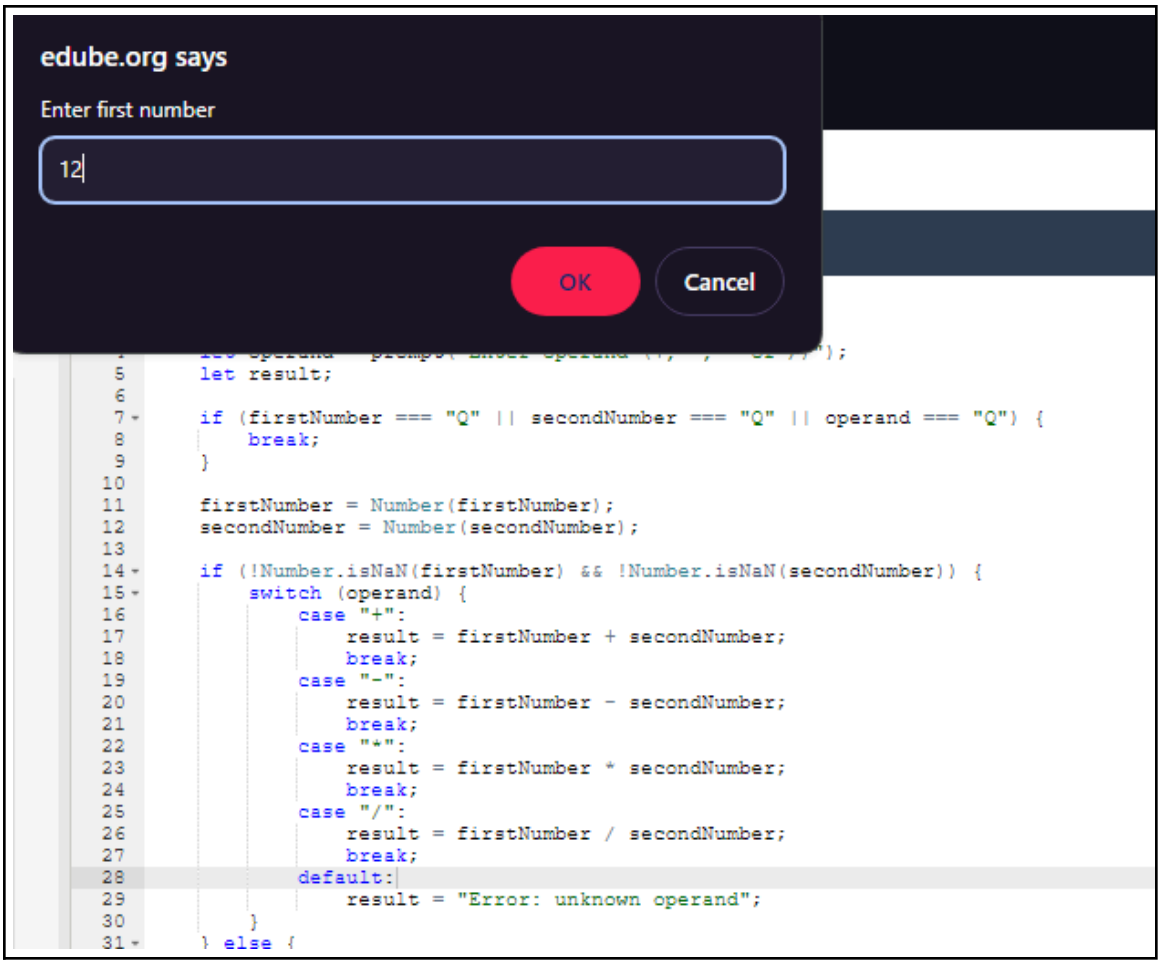
21
45
100
12
11
78
61
4
39
22
100
12
78
4
22

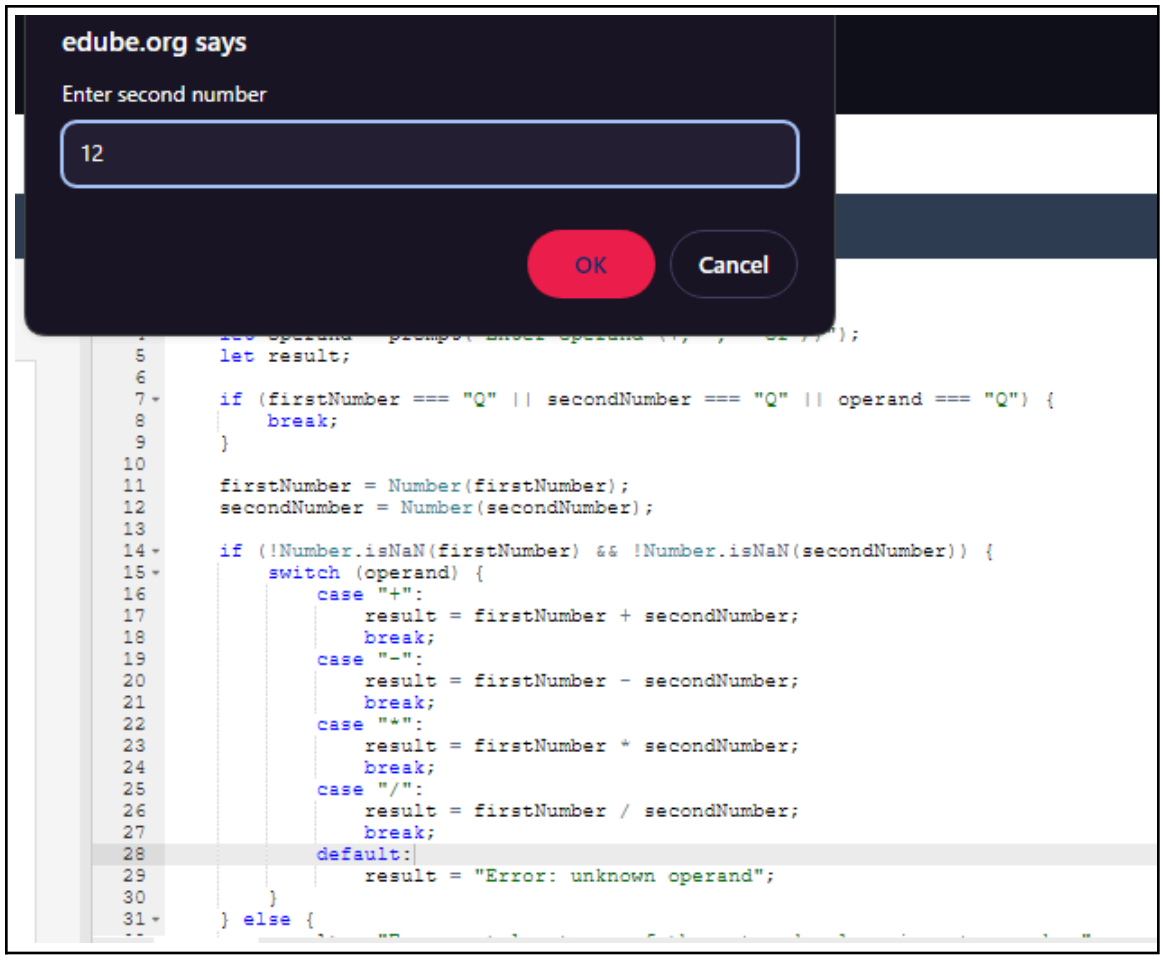
All with ratings under 7:
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
${movie.title} (${movie.rating})
All with ratings over 7:
${movie.title} (${movie.rating})

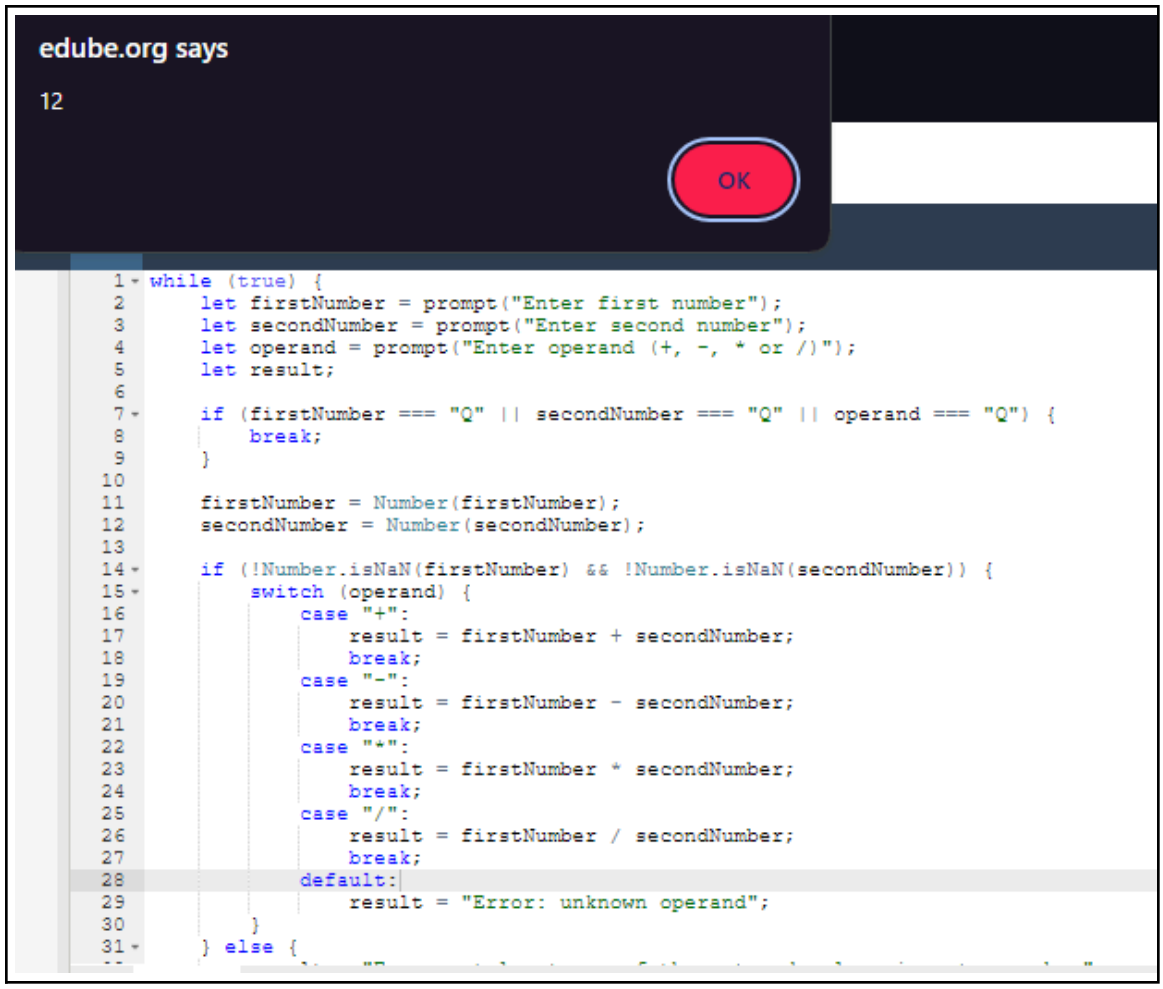
app.js

Console >_

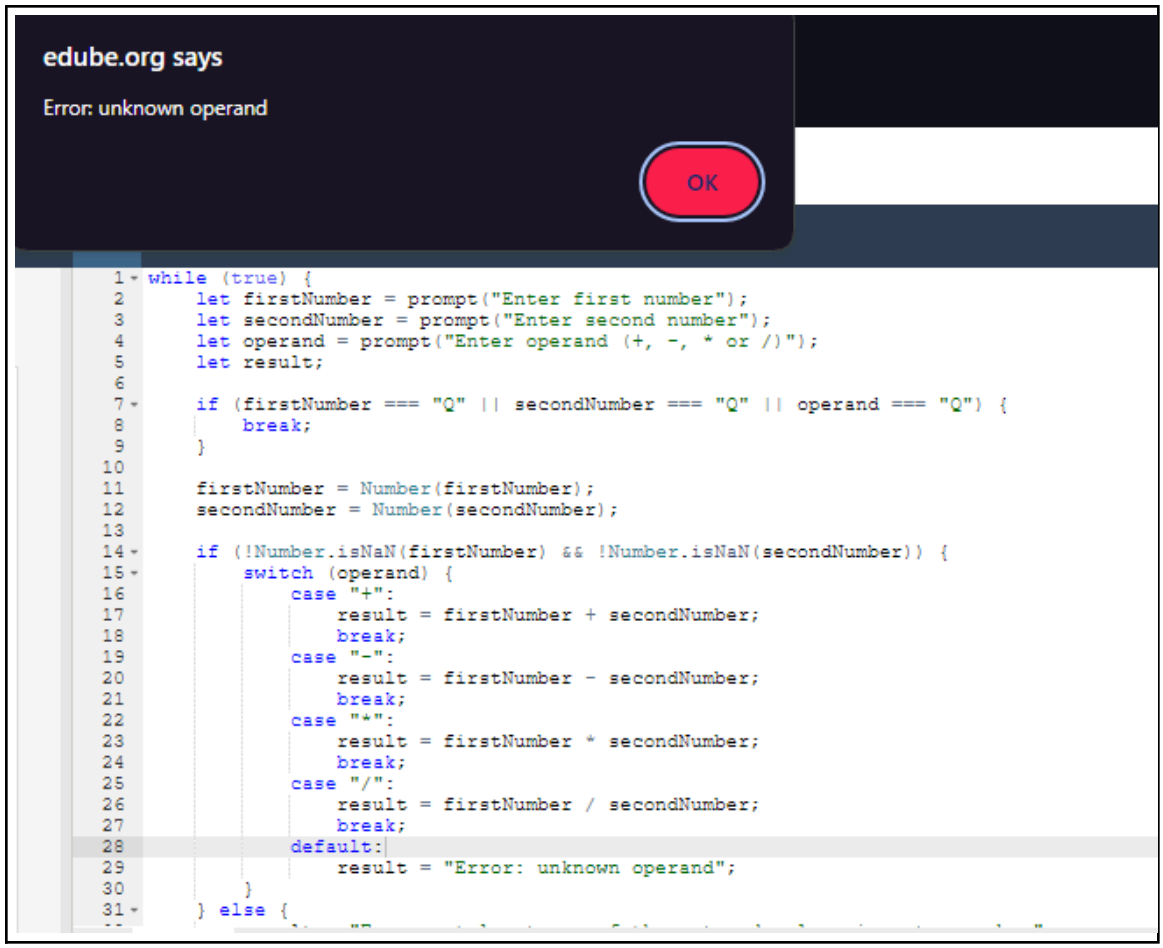
${key} -> ${vessel[key]}
${key} -> ${vessel[key]}
${key} -> ${vessel[key]}
${key} -> ${vessel[key]}
${key} -> ${vessel[key]}
${key} -> ${vessel[key]}


edube.org says
Enter first number
12
OK
Cancel
let result;
if (firstNumber === "Q" || secondNumber === "Q" || operand === "Q") {
break;
}
firstNumber = Number(firstNumber);
secondNumber = Number(secondNumber);
if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
switch (operand) {
case "+":
result = firstNumber + secondNumber;
break;
case "-":
result = firstNumber - secondNumber;
break;
case "*":
result = firstNumber * secondNumber;
break;
case "/":
result = firstNumber / secondNumber;
break;
default:
result = "Error: unknown operand";
}
} else {

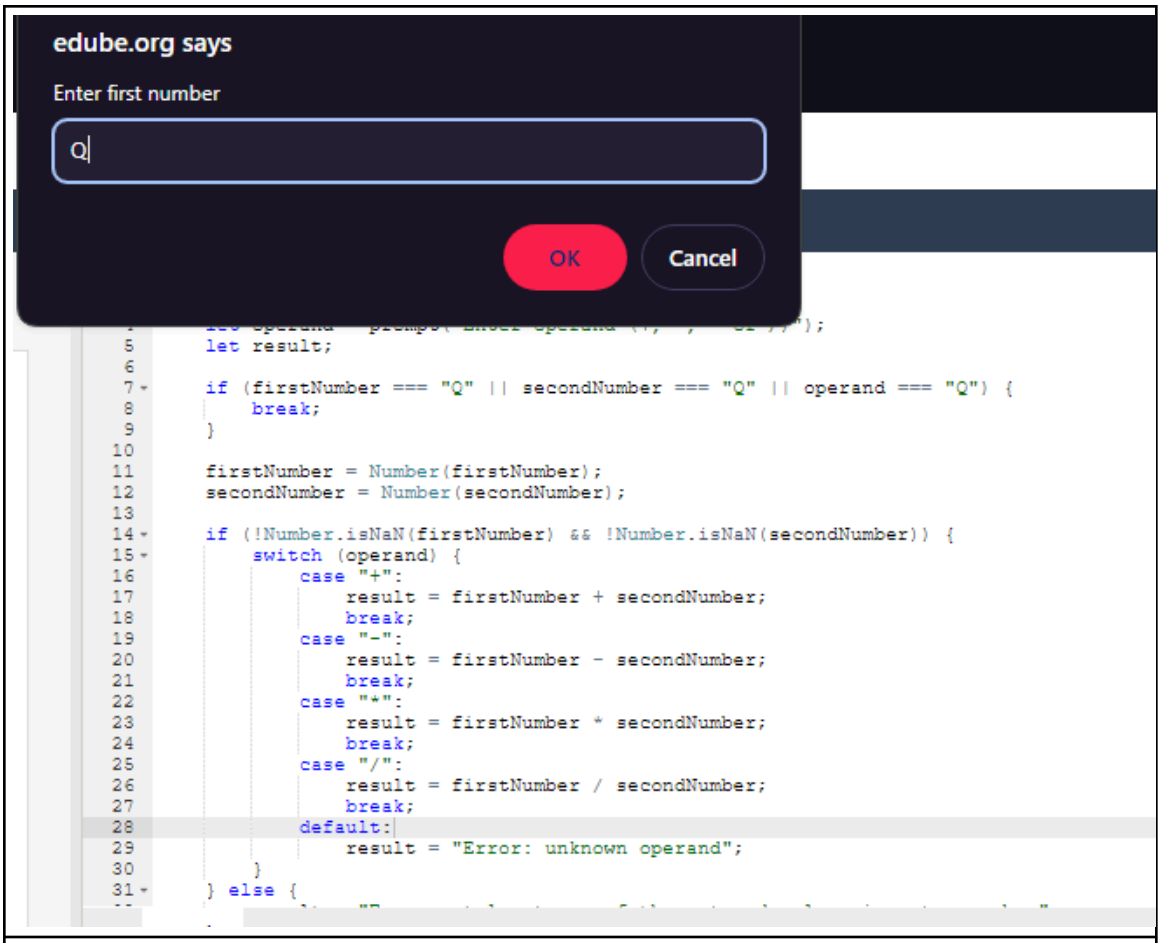

edube.org says
Enter second number
12
OK
Cancel
let result;
if (firstNumber === "Q" || secondNumber === "Q" || operand === "Q") {
break;
}
firstNumber = Number(firstNumber);
secondNumber = Number(secondNumber);
if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
switch (operand) {
case "+":
result = firstNumber + secondNumber;
break;
case "-":
result = firstNumber - secondNumber;
break;
case "*":
result = firstNumber * secondNumber;
break;
case "/":
result = firstNumber / secondNumber;
break;
default:
result = "Error: unknown operand";
}
} else {




```
while (true) {
    let firstNumber = prompt("Enter first number");
    let secondNumber = prompt("Enter second number");
    let operand = prompt("Enter operand (+, -, * or /)");
    let result;

    if (firstNumber === "Q" || secondNumber === "Q" || operand === "Q") {
        break;
    }

    firstNumber = Number(firstNumber);
    secondNumber = Number(secondNumber);

    if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
        switch (operand) {
            case "+":
                result = firstNumber + secondNumber;
                break;
            case "-":
                result = firstNumber - secondNumber;
                break;
            case "*":
                result = firstNumber * secondNumber;
                break;
            case "/":
                result = firstNumber / secondNumber;
                break;
            default:
                result = "Error: unknown operand";
        }
    } else {
```



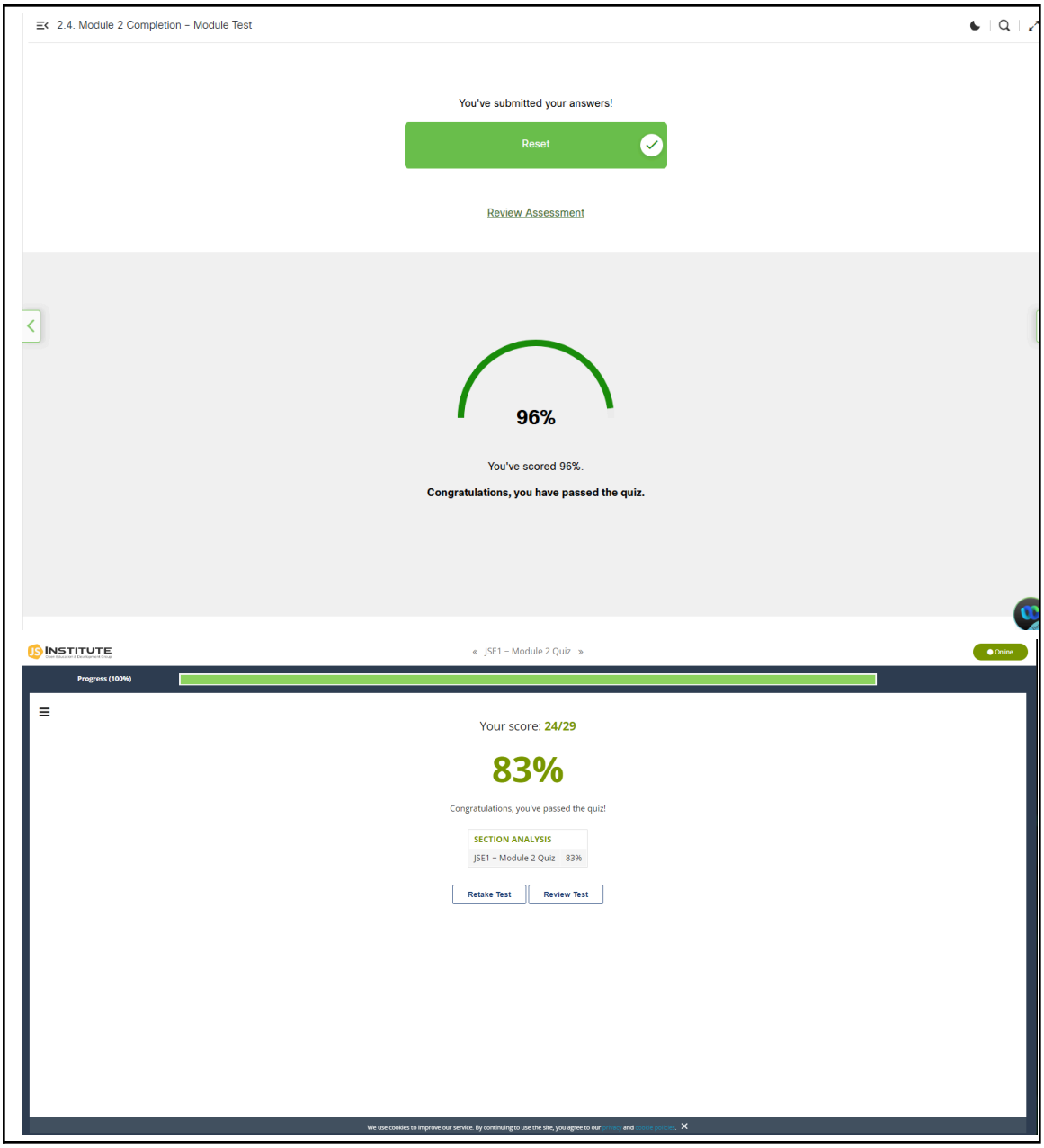

```
while (true) {
    let firstNumber = prompt("Enter first number");
    let secondNumber = prompt("Enter second number");
    let operand = prompt("Enter operand (+, -, * or /)");
    let result;

    if (firstNumber === "Q" || secondNumber === "Q" || operand === "Q") {
        break;
    }

    firstNumber = Number(firstNumber);
    secondNumber = Number(secondNumber);

    if (!Number.isNaN(firstNumber) && !Number.isNaN(secondNumber)) {
        switch (operand) {
            case "+":
                result = firstNumber + secondNumber;
                break;
            case "-":
                result = firstNumber - secondNumber;
                break;
            case "*":
                result = firstNumber * secondNumber;
                break;
            case "/":
                result = firstNumber / secondNumber;
                break;
            default:
                result = "Error: unknown operand";
        }
    } else {
```

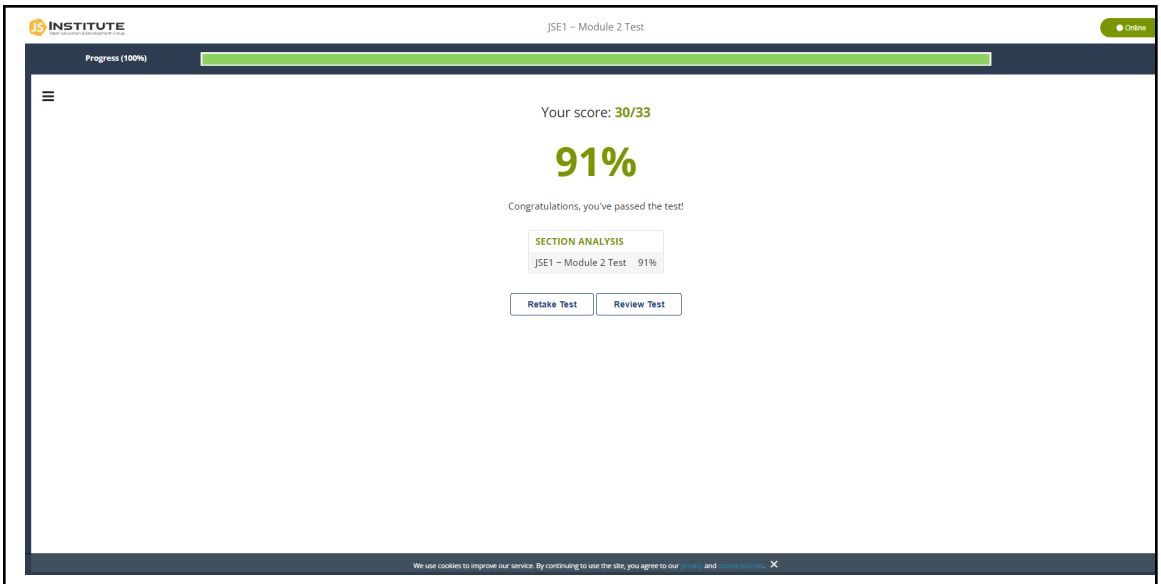


5. Supplementary Activity

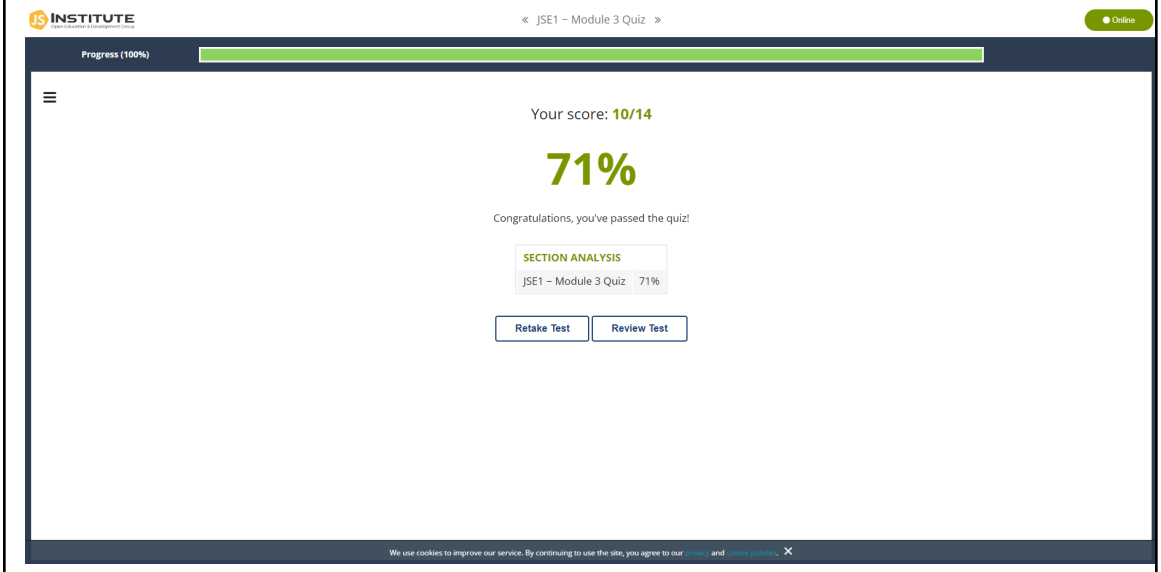
Include here screenshots of the module completion test.

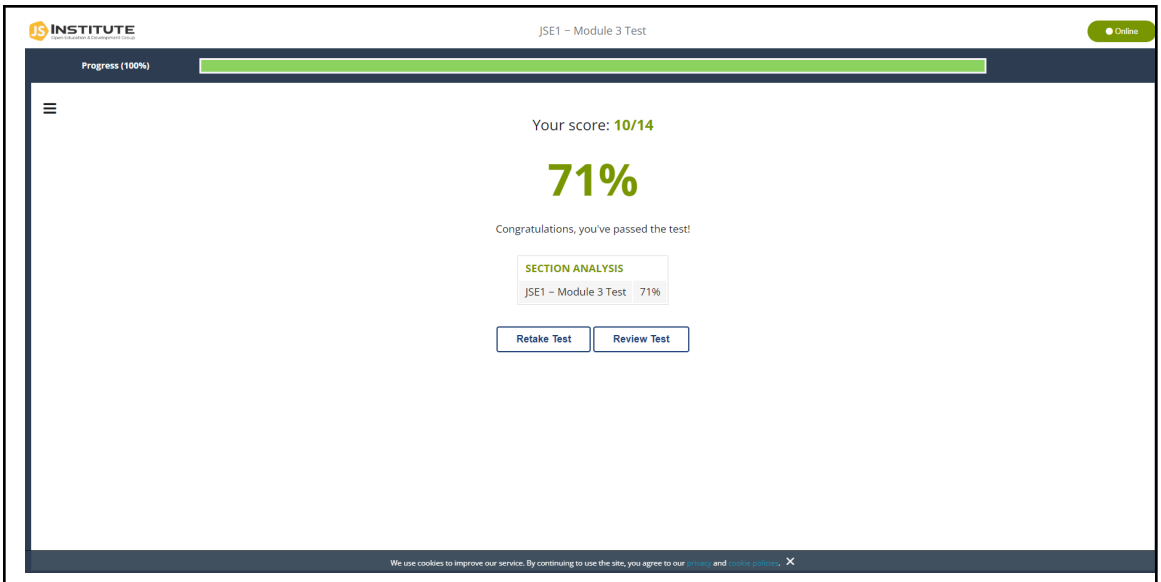
Module 2 Completion test :


2.4. Module 2 Completion – Module Test
You've submitted your answers!
Reset
Review Assessment
96%
You've scored 96%.
Congratulations, you have passed the quiz.
INSTITUTE
« JSE1 – Module 2 Quiz »
Online
Progress (100%)
Your score: 24/29
83%
Congratulations, you've passed the quiz!
SECTION ANALYSIS
JSE1 – Module 2 Quiz 83%
Retake Test
Review Test
We use cookies to improve our service. By continuing to use the site, you agree to our

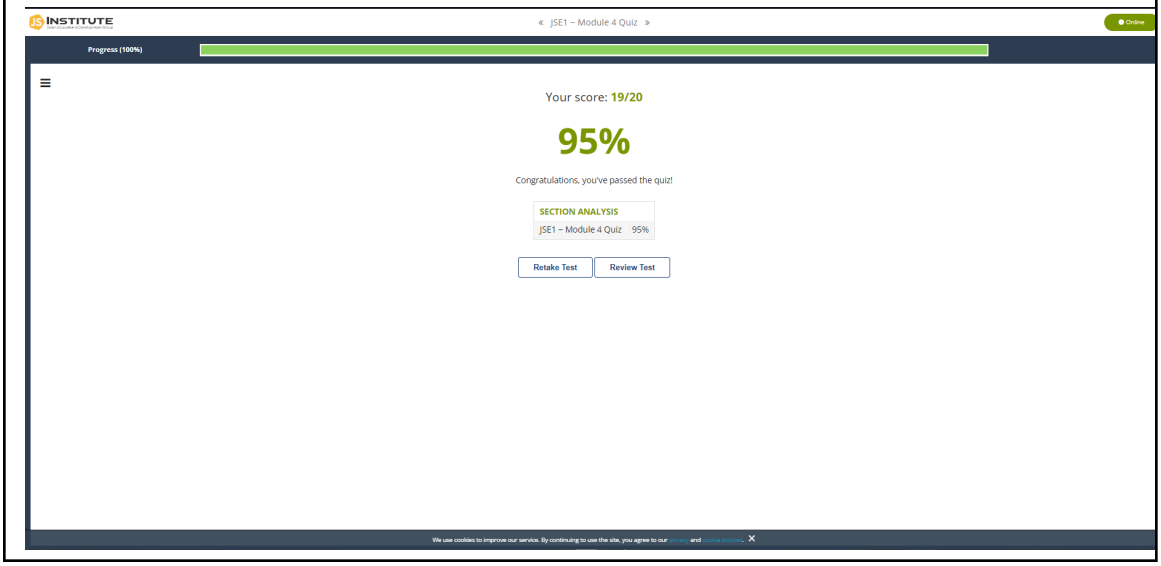


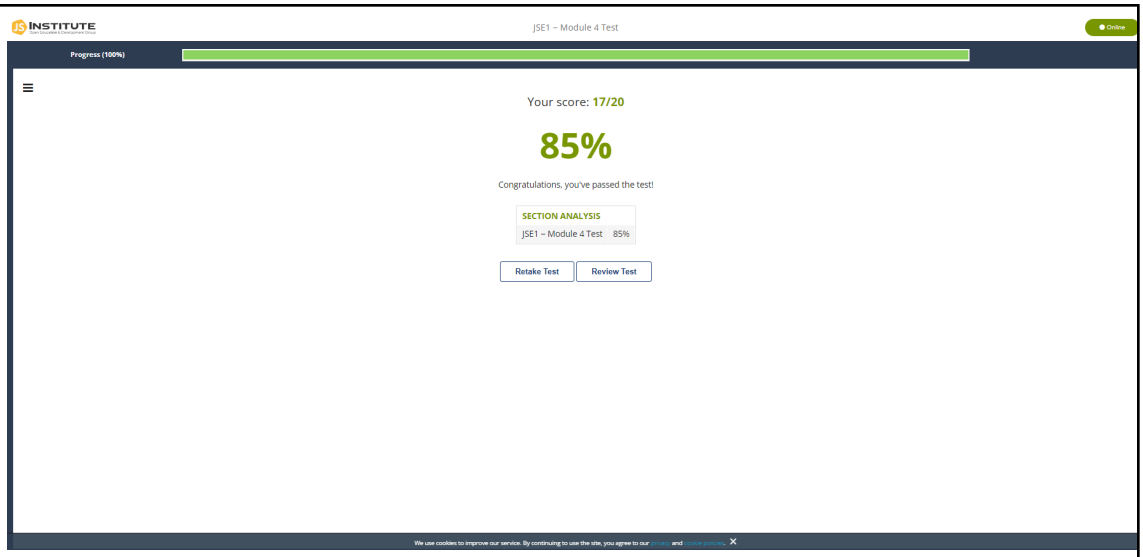
Module 3 Quiz





Module 4 Quiz and test





6. Assessment Rubric

Lab Activity Rubric

Criteria	Ratings						Pts
SO 7 PI 1 **Student Outcome 7.1** Acquire and apply new knowledge from outside sources. threshold: 4.8 pts	6 pts Excellent | Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently and applies knowledge learned into practice	5 pts Good | Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently	4 pts Satisfactory | Look beyond classroom requirements, showing interest in pursuing knowledge independently	3 pts Unsatisfactory | Begins to look beyond classroom requirements, showing interest in pursuing knowledge independently	2 pts Poor | Relies on classroom instruction only	1 pts Very Poor | No initiative or interest in acquiring new knowledge	6 pts
SO 7 PI 2 **Student Outcome 7.2** Learn independently threshold: 4.8 pts	6 pts Excellent | Completes an assigned task independently and practices continuous improvement	5 pts Good | Completes an assigned task without supervision or guidance	4 pts Satisfactory | Requires minimal guidance to complete an assigned task	3 pts Unsatisfactory | Requires detailed or step-by-step instructions to complete a task	2 pts Poor | Shows little interest to complete a task independently	1 pts Very Poor | No interest to complete a task independently	6 pts
SO 7 PI 3 **Student Outcome 7.3** Critical thinking in the broadest context of technological change threshold: 4.8 pts	6 pts Excellent | Synthesizes and integrates information from a variety of sources; formulates a clear and precise perspective; draws appropriate conclusions	5 pts Good | Evaluate information from a variety of sources; formulates a clear and precise perspective.	4 pts Satisfactory | Analyze information from a variety of sources; formulates a clear and precise perspective.	3 pts Unsatisfactory | Apply the gathered information to formulate the problem	2 pts Poor | Gather and summarized the information from a variety of sources but failed to formulate the problem	1 pts Very Poor | Gather information from a variety of sources	6 pts
SO 7 PI 4 **Student Outcome 7.4** Creativity and adaptability to new and emerging technologies threshold: 4.8 pts	6 pts Excellent | Ideas are combined in original and creative ways in line with the new and emerging technology trends to solve a problem or address an issue.	5 pts Good | Ideas are creative and adapt the new knowledge to solve a problem or address an issue	4 pts Satisfactory | Ideas are creative in solving a problem, or address an issue	3 pts Unsatisfactory | Shows some creative ways to solve the problem	2 pts Poor | Shows initiative and attempt to develop creative ideas to solve the problem	1 pts Very Poor | Ideas are copied or restated from the sources consulted	6 pts
							Total Points: 24