Milestone-4 All Module

Module_16_Introduction_to_JavaScript

16-1 Introduction to JavaScript, Run JavaScript in VSCode

- JavaScript is a scripting (interpreted) language.
- We can run JavaScript in many ways. Such as **browser console**, on terminal using **node**, or using online IDE like codepen.io.

16-2 What is variable, five things you need to declare a variable

- A Variable is like a container that contains various kinds of data for future uses.
- In JavaScript, we have needed five things to declare a variable. They are **var (**keyword**), variableName, = (**equal sig**n), a value, and, a ';' (**semicolon at the end**).**
- Some example: **var price = 21; var age = 24; var temp = 37; etc.**

16-3 Variable type, Numeric, String, Boolean

- There are three types of variables.
- **Numeric:** All of the numeric values are included in **numeric type or number.** i.e **var age = 34; var price = 343;**
- **String:** One or more characters of letters is called string. This type of variable is needed to keep into single or double-quotes. **i.e: var name = "Israfil"; var address = "Koyra Khulna";**
- **Boolean:** This is a special type, it has only two values. They are **true & false. i.e: var pass = true; var theft = false;**

16-4 Variable name naming convention and best practice

- JavaScript **keywords** are not allowed
- The name should have to be one word. Use the camel case without any space if need more than one word.
- Quotes are not allowed.
- The first letter must not be a number or symbol except '_' and '$'.

16-5 Simple Mathematical operations in JavaScript

- We can do mathematical operations using JavaScript. We use **"+" (plus sign)** for addition, **"-" (minus sign)** to subtraction, **"*" (astricts sign)** to multiply, and **"/" (forward slash)** to division.

16-6 (advanced) Mathematical operation shorthand

- Do not need to use the **var keyword** to re-declare an existing variable.
- To increase the value of an existing variable (var x = 10), we can use,
    - **x = x + 10;** // x will be 20, or
    - **x += 10;** // same as x = x + 10; (it is called shorthand), same thing is true for **substraction(-), multiplication(*), and division(/).**
    - To decrease or increase the value of a variable by one using **-- or ++; i.e, x--. X++**

16-7 (advanced) Integer float parseInt parseFloat type conversion

- We can **concatenate two strings** together using plus sign (+). **i.e "Israfil" + " " + "Hossen"; // will be - Israfil Hossen**.
- We can convert a number string to the integer number or floating number using **parseInt() and parseFloat(). i.e** *string "5" covert to number 5 using* ***parseInt("5")*** *and the string "5.5" covert to float number 5.5 using* ***parseFloat("5.5").***

16-8 Different variable types and use toFixed with parseFloat

- We can check variable type using **typeof** function. i.e var x = 5; **typeof x**; // will return number & var name = "Sakib"; **typeof name;** // will return string. Etc.
- We can fix the number of the fractional parts of a float using **toFixed()** fucntion. *i.e. var pie = 3.1416;* ***pie.toFixed(2);*** *// will return 3.14. And* ***pie.toFixed(1);*** *// will return 3.1;*

16-9 Module Summary and remainder modulus

- We can determin remainder of two number division using modulo operator (%). i.e **10 % 3;** // will be 1. And **57 % 12;** // will be 9. etc.

Module_17_Fundamental_Concepts_Array_and_Conditionals

17-1 Module Introduction and Variable recap

- Three types of variables are **number, string, and boolean.**

17-2 Declare Array, array length and array index

- An **Array** is a collection of zero or more same or different types of elements. In JavaScript array declare using square brackets []; Elements of array keeps into it by separated with commas. Array's indexing starts from 0.
- We can easily check the number of elements in an array by using **length property. i.e var age = [3, 53, 23]; age.length; //** will return 3.

17-3 Array index, get and set by index, indexOf

- We can get the index of an element from an array using the **indexOf()** method. i.e **var arr = [3, 4, 5, 9, 2]; arr.indexOf(4);** // will be 1; and **arr.indexof(9);** // will be 3; Index counting starts from 0 in every programming language.
- We can also get an element from an array by inserting its index value. i.e **arr[2]; //** will be 5, and **arr[0];** will be 3;
- If we want to want to find the index of an element that is not in the array using **indexOf(),** it will return **-1. i.e arr.indexOf(7); //** will be **1;** and if look for a index that is not exists in the array will return **undefined**. i.e **arr[34];** // will be **undefined**.

17-4 Add or remove element from array using push, pop

- We can use **push()** and **unshift()** methods to add an element at the end and the beginning of an array. i.e **var age = [3, 5, 8]; age.push(9);** // will be [3, 5, 8, 9] and **age.unshift(4); //** will be [4, 3, 5, 8 9];
- We can use **pop()** and **shift()** methods to remove an element from the end and beginning of an array. i.e. **age.pop();** // will be [4,3, 5, 8]; and **age.shift();** // will be [3, 5, 8];

17-5 Compare variables and Comparison operator

- There are many comparison operators in JavaScript. Few of them are like mathematics. They are **> (greater than), < (less than), >= (greater than or equal to), <= (less than or equal to).** For equality **"==" (double equal), or "===" (triple equal)** are used. **"!=" (bang equal)** is used to means, not equality.

- Ther are also **“&&” (and operator),** and **“||” (or operator). “&&”** will be true when both of the conditions are true, otherwise false. i.e. **(isLove && isCare);** // will be true if isLove and isCare both are true. **“||”** will be true if both or only one is true.

17-6 Make conditional decision, if-else, comparison

- We can make conditional desition using JavaScript **if-else** statement. Structure would be, **if (condition) { // code to be executed} else { // code to be executed }. i.e.**
  ```
  If (57 > 37) {
     // code here
  } else {
     // code here
  }
  ```
- We can set various conditions using different kinds of comparison operators.

17-7 Handle multiple conditions, and or

- We can handle multiple conditions **&&, ||** operators. i.e.
  ```
  If (hasFlat == true && moneySave == true) {
     // code here. Code will execute if both of the conditions are true
  } else {
     // code here. This code will execute if any of them is false.
  }
  ```

17-8 (Advanced) Multi stage condition and nested conditions

- We can check multi-stage conditions using **if-else-if. i.e.**
  ```
  If (condition) {
     // code
  }
  else if (condition) {
     // code
  }
  else if (condition) {
     // code
  }
  else {
     // code.
  }
  ```

- Sometimes we need to use the nested conditions like below…

```
If (condition) {
    If (condition) {
        // code
    }
    else {
        // code
    }
}
else {
// code
}
```

Module_17_5_Concept_Recap_and_Loop

17_5-1 Variable array and conditionals revision

- Review some of the topics from previous notes.

17_5-2 While loop, debug and understand while loop

- Syntax for while loop is **while (condition) { // code to be executed }. i.e.**
```
Var x = 0;
While (x < 5) {
   console.log("x is now" + x);
}
```
- We have need to initialize, increment, and set conditions for an effective while loop. Set condition in such a way that it will be false after a certain iteration. Otherwise, it will be an infinite loop.

17_5-3 More while loops, odd numbers, even numbers

- Set loop variable to set initial number, set upper limit (say 20), increment by 2. Start from 1 for odd and 2 for even number using while loop.

17_5-4 For loop, how for loop works, while vs for loop

- For loop is another kind of loop, does the similar task as while loop, its structure is a little different. **I.e. -** *// print out numbers from 1 to 10*
```
For (var i = 1; i <= 10; i++) {
        console.log(i);
}
```

17_5-5 Recap loop, run a loop for each element of an array

- To dynamically loop through an array, set the condition loop variable < array name.length.

Module_18_Core_concepts_functions_and_objects

18-1 Module Introduction and concept Recap

- Variable, array, conditionals, and loop (while loop and for loop) were reviewed.

18-2 Declare a Function, call function, function vs loop

- Sytax for declearnign a function in JS is **function functionName () {** *// code here* **}**
- For calling declared function use **fucntionName();**

18-3 (advanced) Function parameter, function return

- Sometimes function takes an input, this input is called a **parameter.** Some functions **return** some value.

18-4 (advanced) Multiple parameter add, multiplication, etc

- We can pass multiple parameters into a function by separating them using commas.

18-5 Declare multiple objects with multiple properties

- An object is a collection of keys (property) and values. **I.e:**
```
var student1 = {
        name : “Aktarul”,
        class : 9,
        roll : 34
}
```

18-6 multiple ways to get and set object property

- in order to get a property value from an object **objectName.propertyName;**
- To add or change **- objectName.propertyName = value;** or **objectName[“propertyName”] = value;**

18-7 (optional) Javascript switch case break and default

- Sometimes it is better to use a **switch** statement instead of if-else. Because it is fast and straightforward. Syntax is:

```
switch (expression) {
   case value1 :
      // code
     break;
   case value2 :
      // code
      break;
      … … ....
      … … …
   default :
     // code
```

18-8 (advanced) while and for loop break and continue

- Sometimes we have need to **break** (exit loop) or **continue** (skip a step) a loop (while or for) under some conditions. **break** completely exit the loop, **continue** to skip the current iteration, and go for the next.

Module_19_Apply_JavaScript_Concepts

19-1 Module Introduction Apply JS and let, const

- The latest standard of declaring a variable is to use **let** instead of **var.** The **const** keyword is used in some cases where the variable value is like a constant that will not change later.

19-2 Unit Convert Inch to Feet, miles to kilometer

- Write two functions. The first one is for converting inverting inches to feet, and the second one is for converting miles to kilometers.

19-3 Check even and odd number using function

- Write two functions. **isEven(num)** to check a number is even. If even return true. **isOdd(num)** returns true if a number is odd.

19-4 Check whether a year is a Leap Year or not

- Write a function for checking if a year is a leap year or not.

19-5 Calculate Factorial of a number using for loop

- Write a for loop for calculating **factorial** of a number.

19-6 Recalculate factorial multiple times using a function

- Write a function that takes a number as input and returns the factorial of that number as output.

19-7 Factorial using a while loop or a decrementing loop

- Write the factorial function again, but in this case, determine factorial using a while loop and multiply reverse order.

19-8 (advanced) Calculate Factorial in a Recursive function

- Write a recursive function for calculating factorial.

19-9 Module Summary and Simple JavaScript debug

- We can simply debug our code using **console.log()** where the confusion was created. The run and debug option of Node.js can also be helpful using breakpoints.

Module_19.5_JS_Concept_Recap

19_5-1 Javascript concepts and apply revision with four challenges

- Create a function to calculate temperature celsius to Fahrenheit
- Create a function to calculate temperature Fahrenheit to celsius.
- Write a function to calculate Grade.
- Write a function to calculate simple interest.

Module_20_JavaScript_Simple_Coding_Problems

20-1 Module Introduction, Math and Random number

- JavaScript Math object has many useful methods for doing various mathematical tasks very easily. Property List of Math object MDN.
- **Math.random() -** returns a random number between 0 (inclusive) and 1 (exclusive) . Actually, it is a number of 16 decimal value after decimal points.
- **Math.abs()** - returns the absolute value of a number. i.e. **Math.random(-5);** *// will be 5*
- **Math.ceil()** - returns the ceiling (imidiate upper integer value) of a number. I.e **Math.ceil(5.4);** *// will be 6*
- **Math.floor()** - makes a number floor (lower integer value). I.e. **Math.floor(5.98);** *// will be 5*
- **Math.round()** - returns round (nearest integer value) of a number. I.e **Math.round(5.49);** *// will be 5;* **and Math.round(5.5);** *// will be 6*

20-2 Swap variable, swap without temp, destructing

- We can swap the value of two variables by using another temporary variable.
  i.e. let first = 5;   let second = 8;   let tmp = first;   first = second;   second = tmp;
- Another way of swap two values using **destructuring**.
  i.e: [first, second] = [second, first];

20-3 Find max of two values, find max of three values

- **Math.min()** and **Math.max()** take multiple numbers and return their *minimum* and *maximum*.

20-4 Sum of all numbers in an array

- Create a function that takes an array as input, write a for loop in the function to loop through all the elements of the array, and calculate the sum of them.

20-5 Find the largest element of an array

- Create two functions that take an array of numbers and return their largest and smallest value. Use for loop to loop through the elements of the array.

20-6 Create a Fibonacci Series using a for loop

- First of all, declare an array by inserting the first two numbers of the Fibonacci sequence.
- Use a for loop, loop variable starts from 2 because the first two numbers of the Fibonacci sequence have already been defined.
- If the array name is **arr,** loop iterate like this: **arr[i] = arr[i - 1] + arr[i - 2]**; that will generate the next number and enter it at the perfect position of the array.

20-7 Handle unexpected input using simple return

- Keep the Fibonacci sequence generator loop into a function that takes a number for how much the Fibonacci number will create.
- Validate the user input using **typeof input == "number",** return an error message if not match. Also, check the input is greater than or equal to 2.

20-8 (advanced) Fibonacci Element and series Recursive way

- Didn't understand recursive at all. Google it. Spend a few hours to understand JavaScript recursive and ***come back*** to this video again.

20-

9 Module summary and Create Fibonacci series in a recursive way

Module_21_More_JS_Problems

21-1 Introduction and increase problem solving ability

- Write two functions for finding the largest and the smallest value from an input array.

21-2 Remove duplicate items from an array

- JavaScript has another kind of for loop that is quite similar to python for loop.
  i.e. **For (loopVariable, array) {** *// code* **};**

21-3 Explore string nature and reverse a string

- Some **array** properties can be used with **string**, like **string.length, string[***index***],** etc. But the **string** is immutable. Any character of string can not be changed, delete, or add after it is declared. Need to redeclare the string.

21-4 Handle unexpected function input parameter error

- Sometimes functions return **NaN** because of input error. Use debugger and *console.log()* to debug (fix) this kind of problem.

21-5 Use add and multiplication to calculate wood requirements

- Write a woodCalc() function for calculating the total amount of wood needed to make the chair, table, and bed. For 3cft/chair, 10 cft/table, 50 cft/bed wood required.

21-6 (optional)Write foo, bar, foobar if divisible by 3 or 5 or both

- Write a function that takes a number and console the from 1 to that number, but console **'foo'** for divisible by 3 (*num % 3 == 0*), **'bar'** for divisible by 5 (*num % 5 == 0*), and **'foobar'** for divisible by 3 & 5 (*num % 3 == 0 && num % 5 == 0*);

21-7 Find the cheapest phone from an array of phone objects

- Given an array named **phones,** containing a few objects of phone model, price, storage, etc. Write a function that takes this phone array and returns the cheapest price phone object. Hints: use for-of loop, dot notation, if-else statement.

21-8 Calculate the total cost of the products in a shopping cart

- Given an array named **cart,** containing a few objects of products (included their name, price, quantity). Write a function that takes this **cart** array and return their total price. Hints: use for-of loop, if-else, dot notation.

21-9 (advanced) Travelling in a Jungle and counting wild animals

- Write a function name **animalCount(),** that will take a numeric input (how many miles we went inside the jungle) and return the total number of animals. Animal density in the jungle is 10 animals/mile for the first 10 miles, 50 animals/mile for 2nd 10 miles, and 100 animals/mile for the rest of the miles.

21-10 Module Summary and important of problem solving

- Boss-level problem solving is an important skill for a developer, but it is not the most required skill for a junior web developer. We can start a job as junior web developers with basic or intermediate level problem-solving skills and then upgrade it with time.