Date

⇒ Java Script:-

⇒ Introduction to programming:-

Programming is a way to talk to computers.

A language like urdu, eylish or sindhi can be used to talk to a human but for computer we need straight forward instructions.

⇒ Programming is the act of constructing a program, a set of precise instruction telling a computer what to do.

⇒ What is Ecma script?

Ecma script is a standard on which javascript is based; it was created to ensure that different documents on javascript are actually talking about the same language.

Javascript and Ecmascript, can almost always be used interchangably javascript is very literally in what it allows.

Date

⇒ How to excute Javascript?

Javascript can be executed right inside one's browser you can open the javascript console and start writing javascript there.

Another way to excute javascript is a runtime like Node.js which can be installed and used to run javascript code.

Yet another way to excute javascript is by inserting it iside <script> tag of an HTML document.

Date

Chapter 1: Variables and Data

Just like we follow some rules while speaking english (the grammar), we have some rules to follow while writing a javascript program. the set of these rules is called syntax in javascript.

⇒ What is a Variable?

A variable is a container that stores a value.

This is very similar to the containers used to store rice, water, and oats etc (treat this as an analog!)

The value of a javascript variable can be changed during the excution of a program.

```
var a = 7;  → literal
let a = 7;  ⇒ declaring variables
```

Identifier ↙ ↓ assignment operator

Date

⇒ Rules for choosing variable names:-

→ letters, digits, underscores and $ sign allowed.

→ Must begin with a $, _ or a letter

→ Java script reserved words cannot be used as a variable name

→

⇒ Var vs let in Javascript:-

1- ~~letter~~ Var is globally scoped while let & const are block scoped

2- Var can be updated and re-declared within its scope

3- let can be updated but not re-declared

4- Const can neither be updated nor be re-declared.

5- Var variables are initialized with undefined whereas let and const variables are not intialized.

6- Const must be intialized during declaration unlike let and Var.

⇒ Primitive Data types and objects.

Primitive data types are a set of basic data types in javascript object is a non primitive datatype in js.

Date

⇒ These are the 7 primitive datatypes in javascript.

① Null ② Number ③ String
④ Symbol ⑤ undefined
⑥ Boolean ⑦ BigInt.

⇒ Objects:-

An object in javascript can be created as follows.

```
Const item = {
    name: "Led bulb",
    price: "150"
}
```

Key ← name: "Led bulb", - value

price: "150" - value

Date

Chapter #2 Expression and Conditionals

A fragment of code that produces a value is called an expression. Every value written literally is an expression
exg: 78 or "Name"

= Operator in java script:

① Arithmetic operators:-

+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation
/	Division
%	Modulus
++	Increment
--	Decrement

② Assignment operators-

=	x=y
+=	x=x+y
-=	x=x-y
*=	x=x*y
/=	x=x/y
%=	x=x%y
**=	x=x**y

Date

③ Comparison operators :-

= =	equal to
! =	not equal
= = =	equal value and type
! = =	not equal value and not equal type
>	greater than
<	less than
> =	greater than or equal to
< =	less than or equal to
?	ternary operator

④ logical operators :-

&&	logical and
||	logical or
!	logical not

Apart from these, we also have type and bitwise operators. Bitwise operators perform bit by bit operations on numbers.

operands

7 + 8 = 15 → Result

↳ operator

Date

⇒ Comments in Javascript:

Sometimes we want our programs to contain a text which is not executed by the JS Engine

Such a text is called comment in Javascript.

A comment in javascript can be written as follows:

```
let a=2; This is a single line comment
/*
I am a multiline comment
*/
```

↳ Single line comment

} multi line comment.

Sometimes comments are used to prevent the execution of some lines of code

```
let switch = true;
// switch = false;
```

→ Commented lines wont execute

⇒ Conditional Statements:

Sometimes we might have to execute a block of code based off some condition,

e.g:- a prompt might ask for the age of the users and if its greater than 18 display a special msg.

Date

⇒ In javascript we have three forms of if... else statement.

i) if... statement
ii) if... else statement
iii) if... else if... else statement

⇒ If statement:-

The if statement in javascript looks like this:

```
if (caditian) {
  // excute this code
}
```

The if statement evaluates the caditian inside the ( ) If the caditian is evaluted to true, the code inside the body of if is exeuted else the code is not exeuted.

⇒ if-else statement:-

The if statement can have an optional else clause, the syntax looks something like this

```
if (condition) {
// block of code if caditian true
}
else {
  // block of code if condition false.
}
```

Date

If the condition is true, code inside if is executed else code inside else block is excuted.

⇒ if-else if statement:-

Sometimes we might want to keep rechecking a set of conditions one by one until one matches.
we use if else if for acheiving this
Syntax of if... else if looks like this

```
if (age > 0) {
    console.log("A valid age");
}
else if (age > 1088 age < 15) {
  console.log("but you are a kid");
}
else if (age > 18) {
    console.log("not a kid");
else {
    console.log("Invalid Age")
}
```

⇒ Javascript termary operator:-

Evaluats a condition and excutes a block of code based on the condition

Date

Condition ? exp1 : exp2

e.g: Syntax of ternary operator looks like this:

(mark > 10) ? 'yes' : 'No'

↳ if marks are greater than 10, you are passed else Not.

Date

Chapter#3 Loops and functions:-

⇒ We use loops to perform repeated actions.

e.g: If you are designed a task of printing numbers from 1 to 100, it will be very hectic to do it manually loop help us automate such tasks.

⇒ Types of loops in javascript:-

i) for loop → loop a block of code no of times

ii) for in loop → loops through the Keys in of an object

iii) for of loop → loops through the values of an object

iv) while loop → loop a block based on a specific condition

v) do-while loop → while loop variant which runs atleast once

⇒ The for loop:-

The syntax of an for loop looks something like

```
for ( Statement 1; Statement 2; Statemet 3;){
  // Code to be excuted
}
```

Date

→ Statement 1 is executed one time.
→ Statement 2 is the condition base on which the loop runs (loop body is executed)
→ Statement 3 is executed every time the loop body is executed.

⇒ The for-in loop:-
The syntax of for-in loop looks like this:

```
for (key in object) {
  // code to be executed
}
```

Note: for in loop also work with arrays.

⇒ The for of loop:-
The syntax of for-of loop looks like this

```
for( variable of iterable) {
  // code
}
```

↳ for every iteration
↳ Iterable data structure like Arrays, strings etc.

⇒ The while loop
The syntax of while loop looks likes this:

```
while( condition) {
  // code to be excuted
}
```

Date

Note:- If the condition never becomes false, the loops will never end and this might crash the runtime.

⇒ The do-while loop:-

The do while loop's syntax looks like this:

```
do{
// code to be executed
}
while(condition)
```

⇒ Executed at least once

⇒ Functions in Javascript:-

A javascript function is a block of code designed to perform a particular task.

Syntax of a function looks something like this:

```
function my func(..){
    // code
}
function binod func(p1,p2){
    // code
}
binod Func(7,8)
```

↳ function with parameter

↓ Here the parameter behave as local variables.

⇒ Function Invocation.

Date

⇒ Function invocation is a way to use the code inside the function

A function can also return a value

The value is "returned" back to the caller

```
const sum = (a,b) => {
  let c = a+b;
  return c;
}
```

→ return the score

Another way to create and use the function

```
let y = sum(1,3)
console.log(y)
```

→ Prints 4

Date

Chapter # 4 Strings

⇒ Strings are used to store and manipulate text string can be created using the following syntax

let name = "kiran" → Creates a string

name.length

↳ this property prints length of the string

Strings can also be created using single quotes

let name = 'kiran'

⇒ Template literals :-

Template literals use backticks instead of quotes to defines a string

let name = `kiran`

With template literals it is possible to use both single as well as double quotes inside a string

let sentences = `'the name "is" kiran'`

backtics, double quotes

we can insert variables directly in template literal this is called string interpolation

let a = `this is ${name}` → Prints 'this is a 'kiran'.

name is a variable

Date

⇒ Escape Sequence Characters:-

If you try to print the following string, Javascript will misunderstand it.

let name = 'Adam D'Angelo'

We can use single quote escape sequence to solve the problem

let name = 'Adam D\'Angelo'

Similar we can use \" inside a string with double quotes

other escape sequence characters are as follows:-

\n → New line
\t → Tab
\r → Carriage Return

⇒ String properties and Methods

i) let name = "kiran"
name.length → print 5

Date

ii) let name = "kiran"
name.toUpperCase() → Prints KIRAN

iii) let name = "kiran"
name.toLowerCase() → prints kiran

iv) let name = "kiran"
(0, 1, 2, 3, 4)
name.slice(2, 4) → print ra
(from 2 to 4, 4 not included)

v) let name "kiran premchand"
let newName = name.replace("premchand", "psychologist").

vi) let name1 = "kiran"
let name2 = "Ara"
let name3 = name1.concat(name2, "Yes")
↳ we can even use + operator

vii) let name = "kiran"
(0, 1, 2, 3, 4)
name.slice(2) → prints ran
(from 2 to end)

viii) let name = "kiran"
let newName = name.trim()
↳ removes white space

Date

⇒ Strings are immutable. In order to access the character at an index we use the following syntax

let name = "kiran"

name [0] → Prints K

name [1] → Prints i

Date

Chapter # 5 Arrays

Arrays are variables which can hold more than one value.

```
Const fruits = ["banana", "apple", "grapes"]
Const a1 = [7, "kiran", false]
```

↳ can be different types

⇒ Accessing values:-

```
let numbers = [1, 2, 7, 9]

number [0] → 1
number [1] → 2
```

⇒ Finding the length

```
let numbers = [1, 7, 9, 2]
              0  1  2  3

number [0] → 1
number.length → 4
```

⇒ Changing the values

```
let numbers = [7, 2, 40, 9]

number [2] = 8
```

↳ "numbers" now become [7, 2, 40, 9] Array are mutable, Array can be changed

Date

⇒ In Javascript arrays are objects. The type of operator on array returns object.

```
Const n = [1, 7, 9]
```

type of n → return "object"

Arrays can hold many values under a single name

⇒ Array methods:

⇒ there are some important array methods in javascript some of them are as follows:-

i) to string( ) → converts an array to a string of comma separated values.

```
let n = [1, 7, 9]
n.to string( ) → 1, 7, 9
```

2) join( ) → joins all the array elements using a separator

```
let n = [7, 9, 13]
n.join ("-") → 7-9-13
```

Date

3) pop() → removes last element from the array

let n = [1, 2, 4]

n.pop() → updates the original array
returns the popped value.

4) push() → Adds a new element at the end of the array

let a = [7, 1, 2, 8]

a.push (9) → modifies the original array
↳ return the new array length

5) Shift() → removes first element and returns it

6) Unshift() → Adds element to the beginning
Returns new array length

7) delete → Array elements can be deleted using be delete operator

let d = [7, 8, 9, 10]

delete d[1] → delete is an operator

Date

8) Concat( ) → Used to join arrays to given array

let a_1 = [1,2,3]
let a_2 = [4,5,6]
let a_3 = [9,8,7]

a_1 = .concat(a_2, a_3) → Returns [1,2,3,4,5,6,9,8,

↓

Returns a new array
Does not change existing arrays.

9) Sort( ) → sort( ) method is used to sort an array alphabetically

let a = [7,9,8]
a.sort( )
↳ a change to [7,8,9]
[modifies the original array]

Sort( ) takes an optional compare function. If this function is provided as the first argument, the sort( ) function will consider these values (the values returned from the compare function) as the basis of sorting

Date

10) Splice() → Splice can be used to add new items to an array

```
Const numbers = [1,2,3,4,5]
numbers.splice(2,1,23,24)
```

position to add ↙ | No of element to remove ↘ | elements to be added ↘

Returns deleted items, modifies the Array

11) Slice() → Slice out a piece from an array I creates a new array

```
Const num = [1,2,3,4]
num.slice(2) → [3,4]
num.slice(1,3) → [2,3]
```

12) reverse() → Reverse the elements in the source array

Date

⇒ looping through Arrays:-
Arrays can be looped through using the classical Javascript for loop or through some other methods discussed below

1) for Each loop → Calls a function, once for each array element

```
Const a = [1,2,3]
a.forEach((value, index, array) => {
    // function logic
});
```

2) map( ) → creates a new array by performing some operation on each array element

```
Const a=[1,2,3]
a.map((value, index, array) => {
    return value * value;
})
```

3) filter( ) → Filters an array with values that passes a test. Creates a new array

```
Const a= [1,2,3,4,5]
a.filter(greater-than-5)
```

Date

4) reduce method → Reduce an array to a single value

```
const n = [1, 8, 7, 11]
let sum = numbers.reduce(add)
```

1+8+7+11 ↳ a function

5) Array.from → used to create an array from any other object

```
Array.from("kiran")
```

6) for...of → for-of loop can be used to get the values from an array

7) for...in → for-in loop can be used to get the keys from an array.

Date

Chapter # 6: Javascript in the browser.

Javascript was intially created to make web pages alive. JS can be written right in a web page's HTML to make it interactive.

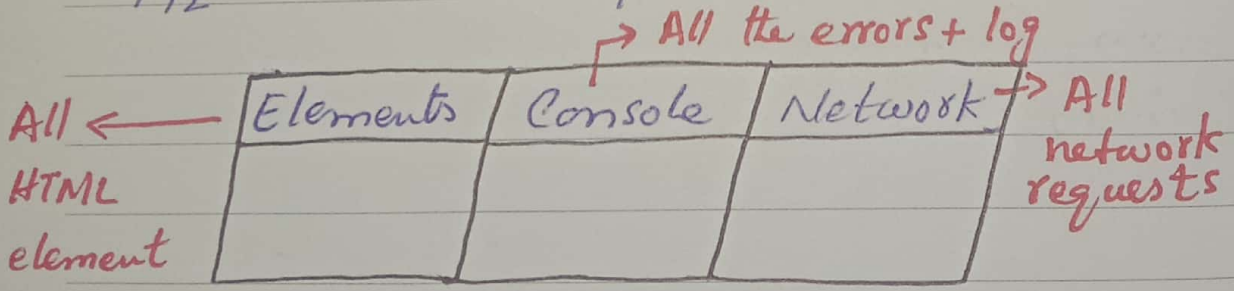
The browser has an embedded engine called the Javascript engine or the javascript runtime.

Javascript's ability in the browser is very limited to protect the user's safety. e.g: a webpage on http://google.com cannot access http://codewear.com and steal information from there.

⇒ Doveloper tools:-

Every browser has some developer tools which makes a developer's life a lot easier.

F12 on chrome opens Dev tools



We can also write javascript commands in the console.

Date

⇒ The Script tag:

The script tag is used to insert Javascript into an HTML page.

The script tag can be used to insert external or internal scripts.

```
<script>
  alert("Hello")
</script>
// or....
<script src="/Js/thisone.js"></script>
```

The benefits of a separate javascript file is that the browser will download it and store it in its cache

⇒ Console object methods:-

The console object has several methods log being one of them. Some of them are as follows:-

i) assert() → Used to assert a condition
ii) clear() → Clears the console
iii) log() → outputs a message to the console
iv) table() → Displays a tabular data
v) warn() → Used for wearning
vi) error() → Used for errors
vii) info() → used for special information

Date

⇒ You will naturally remember some or all of these with time. Comprehensive list can be looked up on MDN

⇒ Interaction: alert, prompt and confirm

i) alert:→ Used to invoke a mini window with a msg alert ("hello")

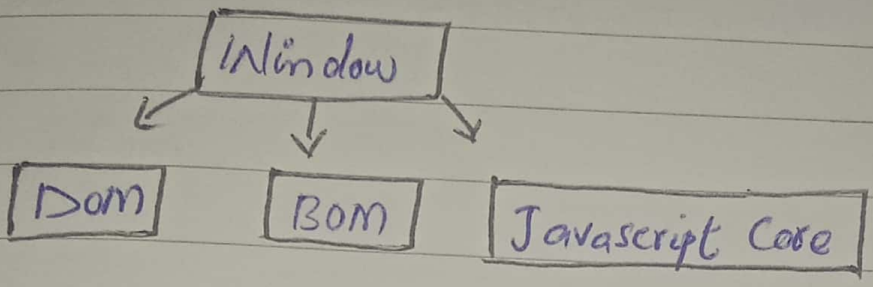
ii) Prompt:- Use to take users input as
string inp = prompt("Hi", "No")
optional
defaut value

iii) Confirm:- Shows a message and waits for the user to press Ok or cancel. Returns true for ok and false for cancel

The exact location and look is determine by the browser which is a limitation.

Date

⇒ Window object BOM and DOM
we have the following when Javascript runs in a browser



⇒ Window object represents browser window and provides methods to control it is a global object

⇒ Document object Model (DOM):-
Dom represents the page content as HTML

document.body → Page body as Js objec
document.body.style.background = "green"
↳ change page background to gree

⇒ Browser object Model (BOM):-
The browser object Model (BOM) represents additional object provided by the the browser (host environment) for working with everything except the document

Date

⇒ The function alert / confirm / prompt are also a part of the BOM

location.href = "http://codecite.com"

↳ redirect to another URL