20th Jan 24.

Java Script Notes.
First class.

Topics:- INTRODUCTION. AND Js File Extention.

- Java Script was invented by Brendan Eich in 1995.
- It was developed for Netscape2, and became the Ecma-262 standard in 1997.
- European Computer Manufacturers Association Ecma International (formally European Computer Manufacturers Association) is an organization that develops standards in computer and technology.
- ES1 to ES5 (1997 to 2009).
- After that in 2015 (major changes to follow the rules and regulations) this is called Ecma Script / Es 2015 / ES6
- ES6 is standard for javascript after that every year new changes came ES7, ES8 ES10 etc.
- Js is a light weight object oriented Programming language.
- Use in form Submit.
- in client side validation.
- Popup / event on click.

Uses:

- Client side execute / browsers. (Js query, React Js, angular Js)

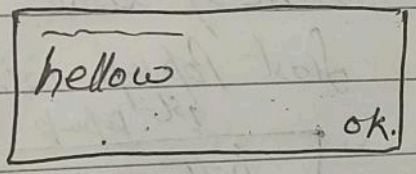
- Website Server Side (node.js, Express js).
- Mobile Development (Hybrid App) (frame work for mobile app react native, phone gap etc)
- Software Development (Ectron js, Ex-Vscode, frame work etc).

index.html ×

index.html > body > script

```
1 <!Doctype html>
2 <head>
3 <meta charset = "UTF-8">
4 <meta ------
5 <title> First class of js </title>
6 </head>
7 <body>
    <script>
      atert ("hello");
    </script>
    </body>
    </html>
```

Result in the form of Popup.

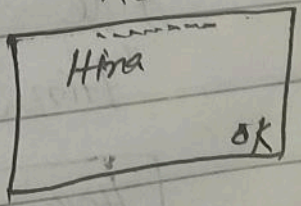


read become
do not want this attribute.

☆ js file. script.js ×

```
1 alert ("Hina");
```

Result:



Result will be Popup of Hina.

Boolen attribute

by putting attribute "defer"

```
<head>
    <script defer src="script.js"></script>
</head>
<body>

    <script src="script.js"></script>
```

if we will link our js file in the body tag before the closing body tag the result will also show in the form of popup.

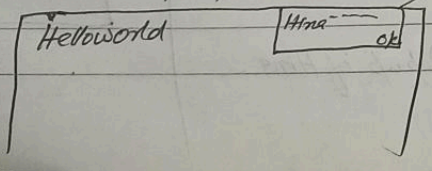
```
<body>
    <h1> Hello world </h1>

<script src="script.js"></script>
</body>
</html>
```

External file. JS script.js

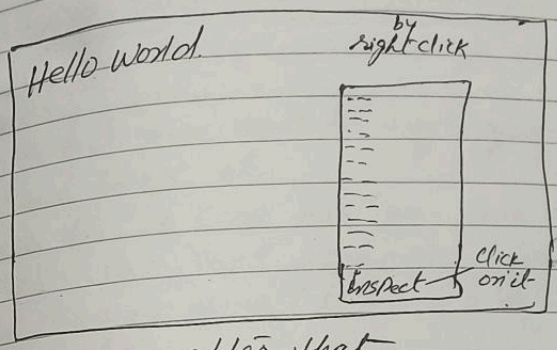
```
1  alert("Hina")
```

Result: First of all on executing the file, hello world shows and then "Hina" popup comes.

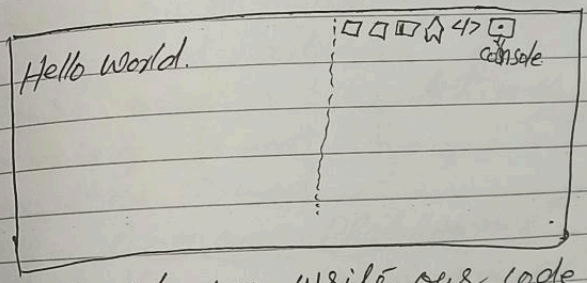


FOR CONSOLE

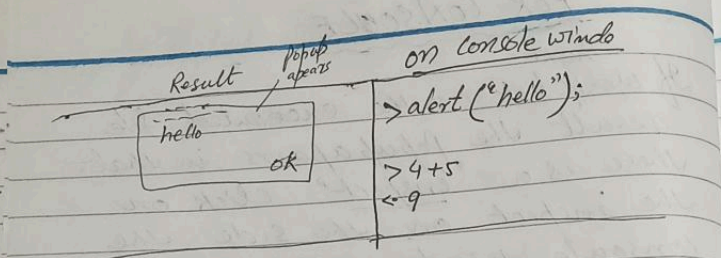
If we right click on the executed file. result the popup apears in that there is a "inspect" click on the inspect on the side the console windo will open



after that



on console we write our code and can execute it over there.



"2nd+3rd class. JavaScript"

Variables:-

- Variable is just like a container.
- Variable is used to store information.
- It reserves space in memory. Its data can vary but memory location will always remains same.

Naming Variables in JavaScript

Rules:

- Variable's name can't be any keyword. eg alert prompt etc
- Variable is case sensitive. Same name in capital and small letters are different. eg Name or name (both are 2 different variables).
- Variables can be consist of alphabet, number, dollar sign and underscore.
- Variable name can't be start with digits (number) its first letter.
- no space allowed.

"AS A GOOD PROGRAMMAR"

- Your variable name should match with its contents.
- When you want 2 words join in variable name, so first word start with small letter

and 2nd word start with capital letter.
eg. fullName, rollNumber etc.

TYPES OF Variables

- **Var** (used before Ecma Script. this type of variable can be declare again and again in js).
  After ES6 in modern or advance javascript these 2 keywords use for declaration variables.

- **Let** (its value can change any time in programming language and can declare and assign in 2 steps.
  eg let name; (declare)
  name = "hellow"; (assignment)

- **Const** (it use for constant value eg Pi value i.e 22/7). it value can't be changed. its value must be assigned at the time of declaration.
  eg const name = "Hina";
  (declare and assign in same sentence).

Variable Scope

1) Block Scope Variable:-
   if variable declare in block of codes (in curly braces {}).
   it will alive only in block and will not be accessable after curly braces.

2) Global Scope Variable: these variables used globally in whole program.

Comments in JS

- Single line: // let name = "hina";
- multi line: /* */

Note:
- Comments means line will not execute.
- Browser is a complete document it creats a windo object and 2nd object is document.

Print / Display in js.

- On Browser: ⟶ window.document.write("hina");
- in console ⟶ console.log("hina");
- Popup ⟶ window.alert("hina");

js file

text string

Js variables.js

```
1  window.document.write("hello");
```

Note:- we can write one or more HTML expressions to a document in the specified window.

<> index.html x js variables.js

```
10  <body>
11      <h1> variables rules </h1>
12      ---
        ---
        ---
    <script src="varibles.js"></script>
```

Result ⟶

Variables rules
hello

for printing on windo we can only write.

```
document.write("world");
```

windo is a complete object and its sub object is document.

<> Js variables.js x

```
1  windo.document.write("hello");
2      document.write("world");
```

Result will be

Variable rules.
helloworld

We will not write "window" because sentence becomes to long. we write only document.write to print on the browser.

In console

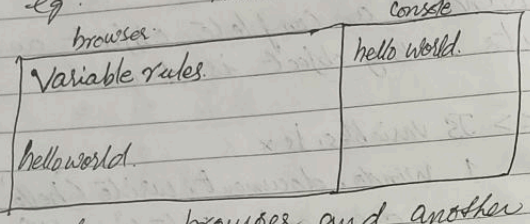
for print on console we write

```
consol.log("hina");
```

اسکو بھی ونڈو کے ساتھ لکھ سکتے ہیں

its result will not show on browser but it shows on the console windo.

eg. Result



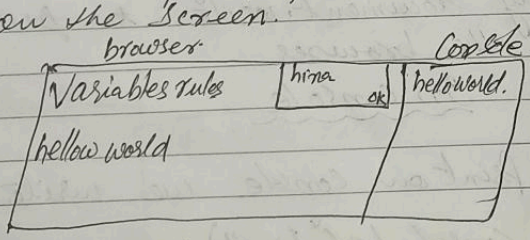
one print on browser and another print on console.

POPUP

JS variable.js x

4. Windo. alert ("hina");
   or
   alert ("hina");

we can write ~~windo.~~ alert ("hina") (by default)
or windo. alert ("hina")
u
for both way popup will appears on the screen.



Print / Display in js

Taking input from users in js.

- Prompt: for javascript, we use the prompt() function to ask the user for input. As a parameter, we input the text we want to display to the user. Once the user presses "OK" the input value is returned. We typically store user input in a variable so that we can use the information in our program.

let answer = Prompt ("Do u want to send payment Y/n ?:");

in answer variable value will be store and we can print it.

JS variable.js x

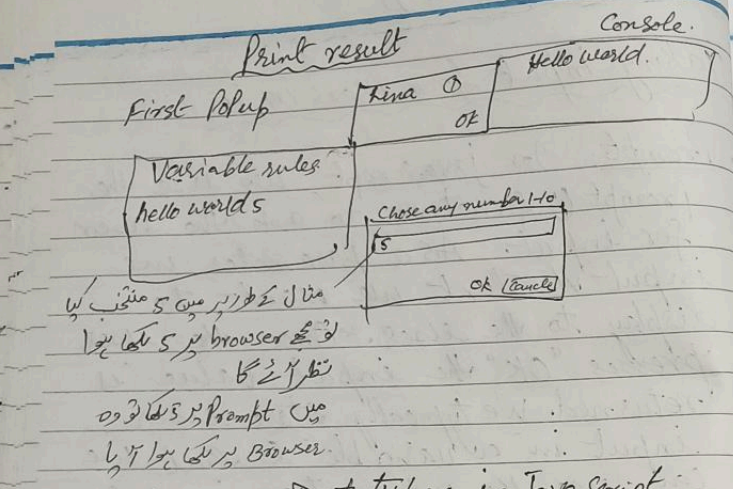
```
1. document. write ("hello");
2. document. write ("world");
3. console. log ("hello world");
4. alert ("hina");
5. let num = Prompt ("choose any number (1-10");
6. document. write (num);
```

(num) ← Variable

document. write → for print

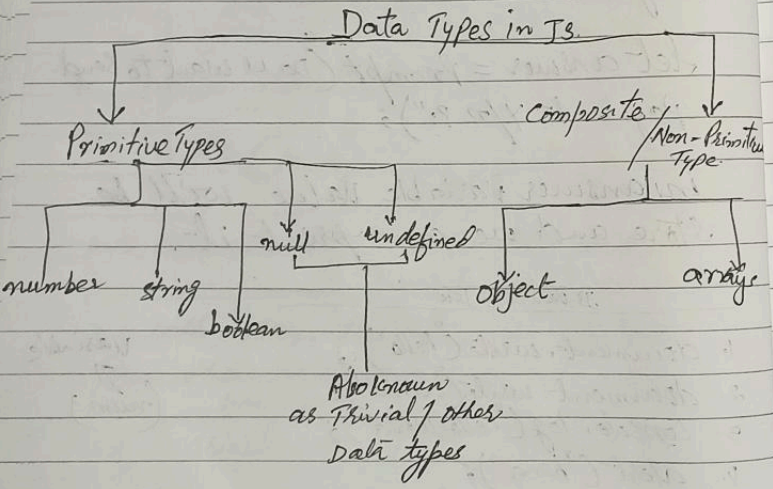
we want it value, its a variable. that why we donot put " " here.

Print result



مثال کے طور پر میں 5 منتخب کیا
جو کہ براؤزر پر 5 لکھا ہوا
نظر آئے گا
دوسرا کہ Prompt میں
لکھا ہوا Browser

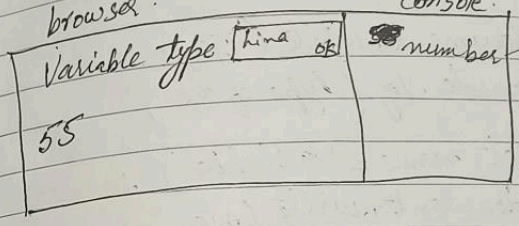
Data types in JavaScript



Js variable.jsx

// number.

```
let age = 55;
document.write(age);
console.log(type of age);
```



// String

Primitive Data types

To check data type
type of Variablename

- number → let rollNo = 56;
- String → let name = "Alishba";
- boolen → let isPass = true;
- undefined → let Percentage;
- object (null) → let class = null;

Variable.js

5- ~~Data name~~

// number

let age = 55;

document.write(age);

console.log(type of age)

let name = "alishba";

document.write(name);

console.log(typeofname);

Result will be

browser	Console
Variable values 55 alishba	number String.

By changing the value we will write another name with single ' '

// number

let age = 55;

document.write(age)

console.log(typeofage) → let name = "alishba";

name = 'kiran';

document.write(name); console.log(typeofname)

what will come

In this condition it will give me first in the result

I just assign the value not declare writing "let"

by changing the value new name will come i.e kiran.

Result will.

windo	Console
Variable values 55 kiran	number String.

if we donot give any value its data type is undefine,

ایک ہوتا ہے null, یہ بھی ایک طرح کا object ہوتا ہے

یہاں کوئی value نہیں لکھی گئی

boolean

it return us true and false

now make a variable.

let ispass = true; → writing without " "

JS Variable.JSX

1 // number
2 let age = 5
3 document.write (age)
4 console.log (type of age)
5 let name = "alishba";
6 ~~name~~ = 'Kiran';
7 document.write (~~name~~)
8 console.log (type of ~~name~~);
9 let ispass = true;
10 document.write (isPass);
11 console.log (typeof isPass);
12 let class; — null
document.write (class);
console.log (typeof class);
let abc = null;
document.write (abc);
console.log (typeof abc);

Result

Browser	Console
Variable rules:	numbers
55 Kirantrue undefinenull	string
	boolean
	undefined
	object

(numbers, string, boolean, undefined, object → datatypes)

We can not use any Keyword as a variable name

Composite / Non-Primitive Types

(ڈیٹا multiple کو ایک ساتھ store کرتا ہے)

1. **Array:**
   - Store multiple value in single variable.
   - Values written in square backets []

Syntax:- variable (role no, name, class)

```
let info = [5, "hina", "computer"];
console.log (info);
document.write (info);
document.write (info[1]);
```

} Two ways to print

Print → document.write (info[1]);

JS Variable.js x

values

```
1 // array.
2 let info = [5, "hina", "computer"];
  document.write (info);
```

اگر ہم values کو الگ الگ بھی پرنٹ کروا سکتے ہیں اگر میں چاہوں کہ پہلی value کو پرنٹ کروانا چاہوں تو میں اس طرح پرنٹ کرواؤں گی
Values کمپیوٹر میں پہلی value 0, 1, 2
5 hina, computer

Result

Variable rules.
5, hina, computer

نوٹ:
index 0
index 1
index 2

اگر مجھے پہلی ویلیو پرنٹ کروانی ہے تو میں [0] لکھوں گی اگر index 1 [1]
[2] index 2

اگر میں

```
// array
1. let info = [5, "hina", "computer"];
2. document.write (info [1]);
```

Result

Variable rules.
hina

② Object

- Store multiple value in single variable.
- Value written in curly brackets { } in pairs with keys.

Syntax:

```
let student = {
    name: "Lina",
    rollno: 23,
    class: "computer"
}
```

Print →

```
document.write(student);
document.write(student, rollno;
```

نوٹ: Array میں ہم Values کو define نہیں کرتے تھے
میں نے 5 رول نمبر نہیں بتایا hina کو نام نہیں بتایا اور
Computer کو کلاس نہیں بتایا جبکہ Object میں ہمیں
Values کو define کرنا پڑتا ہے۔

// object

```
let student = {
    rollno: 5,
    name: "Dania",
    sub: "Maths",
}
document.write(student);
console.log(typeof student);
```

Print

Result

?

daraz.com → یہاں پر datatype کی پریکٹس کرنی ہے

کوئی بھی ایک چیز لے کر اسکی object بنا لیں۔

1.

```
let item = {
    name: "lipstick",
    Price: 250,
    rating: 4,
    Available: true,
    offer: 20%
}
~~document.write(item~~
console.log(item);
```

on Console

browser

▸{name: 'lipstick', price: 250, rating: 4, available: true, offer: 20}
> typeof item
< "object"
> item.price
< 250
> item.offer
< 20
> item.available
< true

"FUNCTIONS" –

we need two things. our Compiler will not read the function until it will not invoke (mean call) we can call it in js or html file.
for defining function we have to write function

Js variable.js x

```
1. function welcome(){
   {
     document.write("welcome");
   }
   function sum(a, b)
   {
     let c = (a+b).
     return c
   }
```

but it will not execute.

4th class javaScript

Operators in JS

1. Arithmetic Operator
2. Assignment Operator
3. Comparison Operator
4. Logical Operator

To Perform any function we use operators

function invoke

variable.js x

```
① function welcome( )
  {
    document.write("welcome");
  }
invoke → welcome( );
```

Result
Welcome

```
② function Sum (a,b)
  {
    let c = a+b;
    return c
  }
let answer=Sum (4, 5);
document.write(answer);
document.write(typeof Sum);
```

{a = 4, b = 5}

Result-
9 function

Class ④

OPERATORS JS (Part 1)

They used to Perform any task.

1. Arithmetic Operators
2. Assignment operators
3. Comparison operators
4. Logical Operators
5. Conditional operators

Arithmetic Operators

a+b, 4+5 (operand)

+ → operator

For these we need two values.

1. + (Addition)
2. - (Subtraction)
3. × (multiplication)
4. ÷ % (Division).
5. % (modulus / remainder)
6. Exponentation
7. increment
   Decrement

They work on single operaned.

Unary Operators
- Post increment a^{++}
- Pre increment ++a
- Post decrement a^{--}
- Pre decrement --a

Coding

```
1  // Arithmetic operators
2  let a = 9;
3  let b = 5;
```

// addition

```
document.write(a+b);
```

or

```
document.write(a, "+", b, "=", a+b);
```

Result 9+5=14 on browser

answer will be (14) on browser.

نوٹ:
اگر ان کی value کسی Variable میں رکھی ہے تو اور براہ راست کرنا چاہیں تو ان Print کو direct کروا سکتے ہیں۔

Subtraction

```
document.write(a, "-", b, "=", a-b, "<br>");
```

9-5 = 4.

multiplication

```
document.write(a, "*", b, "=" a*b, "<br>");
```

a×5=45

Division

```
document.write(a, "/", b, "=" a/b, "<br>");
```

9/5 = 1.8

% modulus

```
document.write(a, "%", b, "=" a%b, "<br>");
```

Exponentation

$2^2, 2^3, 2^4$ — exponention

```
let a = 5;
let b = 3;
document.write(a, "**", b, "=", a**b, "<br>");
```

a to mean $5^3 = 125$

- increament ++
- Decrement --

// Uranary Operand

// a++; a = a+1;

// a--; a = a-1;

a++ mean one +1

```
let a = 5;
let b = 3;
a++;
document.write(a);
```

Result (5+1 = 6)

```
b--;
document.write(b);
```

(3-1 = 2)

اگر ہم a++ کو direct پرنٹ کروا دیتے ہیں تو اسکا جواب 6 نہیں بلکہ 5 ہی آئے گا کیونکہ پہلے a کی Value آئے گی اسی کو ~~پوسٹ~~ post increment کہتے ہیں

eg.

```
let a = 5
document.write(a++);  →
```

Post increment Result: 5 not 6

لیکن اگر اس میں سے ++ ہٹا دیں تو

```
document.write(a),  →
```

Result will be 6

Post decrement

```
let a = 5
document.write(a--);  →  5
document.write(a);    →  4
```

Result

~~Pre increment~~

Pre decrement

```
let a = 5;
document.write(--a);  →  4
document.write(a);    →  4
```

Result

کیونکہ ہمارا decrement پہلے ہی ہو گیا۔

Pre increment

```
let a = 5;
document.write(++a);  →  6
document.write(a);    →  6
```

Result

☆ Assignment Operators (assign value)

=	a=2; left side = right
+=	a+= 4 / a = a+4;
-=	a-= 4 / a = a-4;
*=	a*= 4 / a = a*4;
/=	a/= 4 / a = a/4;
%=	a%= 4 / a = a%4;
**=	**= a** = 4 / a = a**4;

Assignment operators value variable

1. // Assignment operators.
2. let a = 2;
   a+=4;
   document.write(a); // 6.

Result

let a = 8;
a-=3;
document.write(a); // 8-3 = 5

Result

let a = 8
a*=3; → 8×3 = 24.
document.write(a);

let a = 8
a/=3;
document.write(a); → 8/3 = 2.666

operand (modula) 3×2 = 6, 8-6 = 2 operand. 2 remainder

let a = 8;
a%=3;

let a = 8; (it means: 8³ = 8×8×8 = 512)
a**=3;
document.write(a);

Result 512.

Class (5) Javascript

Operators in JS (Part 2)

Comparison Operator

same ہونا

== (equal to) === (equal to + same data type)

!= (not equal to) !== (not equal to + data type)

> greater than

>=

< less than

<=

== (equal to) اسکا نتیجہ Bodean میں آتا ہے

یعنی اگر ہم لکھتے ہیں 2==2 تو ہمیں true یا false

اسکا رزلٹ yes میں ملے گا۔

! کا مطلب کمپیوٹر کی زبان میں not ہوتا ہے یعنی

1 let a = 2;
let b = 2;
a == b → it returns true

اور اگر میں اس میں b کو string بنا دو تو بھی یہ true ہی نتیجہ دے گا

جیسے

1. let a = 2;
2. let b = "2";
3. a == b
4. 2 == 2

اگر ہمیں Perfect کام چاہیے تو ہم === لگاتے ہیں۔

یعنی

1. let a = 2; → "data type number"
2. let b = "2"; → "data type string"
3. a == b → true (give)
4. a === b → false
   2 === 2

کیونکہ دونوں کی data type مختلف ہے اسلئے وہ ہمیں false نتیجہ دے گا اس کی وجہ یہ ہے کہ دونوں data type مختلف ہونے کی وجہ سے ہمیں data type بھی دونوں کے لئے === Perfect کام نہیں مل رہا۔ Same ہونے چاہیں پہلی صورت میں وہ چیک کرے گا کہ چیزیں Same ہیں یا نہیں جبکہ دوسری صورت (===) میں وہ چیزوں کے ساتھ ساتھ data type بھی چیک کرے گا

اب اگر ہم not-equal != لگائیں تو۔

1 let a = 2;
let b = "2";
a != b false
a !== b true — it checks datatype as well as values

condition

کیونکہ datatype مختلف ہیں اس لئے یہ برابر نہیں ہے تو نتیجہ true آئے گا۔

LOGICAL OPERATOR.

- Logical AND && → اس کے دونوں condition چیک ہوتی ہیں
- logical OR ||
- Logical NOT ! اگر condition جو کہ not ہے تو اسکو not بنا دے گا

CONDITIONAL OPERATOR.

if statement → اس میں صرف ایک ہی کام دے گئے کہ پورا کر دے
if-else statement → اگر ایک کام صحیح ہے تو ٹھیک ہے ورنہ پورا کر دیں
if-else if statement

TERNARY OPERATOR.

condition ? true output : false output;

eg:

age > 18 ? "adult" : "not adult";

JS Variables x.

1. // Comparision Operator
2. let a = 5;
   let b = "5";
   if (a = b)
   {
   document.write("no")
   }

اگر a == b ہے تو
تو مجھے پرنٹ کر کے دے
چونکہ datatype
کو نہ اسلئے یہ true
یا false رزلٹ نہیں دے گا

Result on browser

no

اس کا یہ مطلب ہوا کہ یہ true تھا

اگر میں no کی جگہ کچھ اور لکھ دیتی ہوں جیسے

1. let a = 5;
2. let b = "5";
   if (a = b)
   {
   document.write("hello")

اگر a = b ہے تو مجھے hello پرنٹ کر دے۔

Result on browser

hello

اس کا مطلب ہے کہ یہ true ہے کیونکہ statement صحیح ہے۔

اگر میں یہی کام accurate چیک کرنے کے لئے کرتی ہوں
تو میں === لگا کر چیک کروں گی تو دیکھوں گی کہ یہ
hello پرنٹ کرنے دیتا ہے یا نہیں۔ یہ نہیں دے گا
کیونکہ یہ datatype بھی چیک کرے گا جو کہ مختلف
String اور Number ہے یعنی

e.g.

1. let a = 5; → datatype ~~String~~ Number
2. let b = "5"; → " String

```
if (a===b)
{
  document.write ("hello");
}
```

Result on browser

Empty
یعنی hello پرنٹ نہیں ہوا

اگر میں ان کی datatype بھی ایک جیسی کر دوں تو پھر
مجھے blockofcode پرنٹ کر کے دے گا۔
جیسے کہ

1. let a = 5; → datatype number
2. let b = 5; → number

```
if a===b
{
  document.write ("hello");
}
```

Result on browser

hello.

1 == condition

1. let a = 5; → number
2. let b = 5; → number
3. if (a!==b)

```
{
  document.write ("hello");
}
```

اس صورت میں بھی hello پرنٹ نہیں ہوگا چاہے datatype ایک جیسا ہو یا مختلف ہو کیونکہ اسکو پتہ ہے کہ یہ same ہیں اور not جو کہ compare کرتا ہے

Result on browser

Empty

لیکن اگر میں اس میں مختلف datatype کردوں تو
اب یہ hello پرنٹ کرکے دے گا کیونکہ اپنے dala type
جیسے نیچے میں مثال کے طور پر

1. let a = 5; number
   let b = "5"; string
   if (a==b)
   { document.write("hello");

Result on browser

hello

if else condition

operator

let age = 14;
if (a>=18)
{ document.write("You can vote")
}
else{ document.

Result

else { document.write("You can not vote");

Result

You can not vote

اگر میں age تبدیل کردیتا ہوں

let age = 23;
if (a>=18).
{ document.write("You can vote");
else{ document.write("You can not vote");

Result

You can vote

اگر میری عمر 18 سے زیادہ ہے تو

let age = 23 // number.
if (age<18);
{ document.Write("You are child");
} else if (age > 40)
{ document.write("You are old");
}

space دیتا ہے else if

else
{
document.write("You are young")
}

Result

| You are young |

23 کی جگہ اگر عمر 70 لکھ دی جائے تو نتیجہ

| You are old |

23 کی جگہ عمر 7 لکھ دی جائے تو نتیجہ

| You are child. |

LOGical OPERATOR

اس میں دو condition چیک ہوں گی
&& condition 1 & condition 2

اگر دونوں condition true ہوں گی تو پروگرام چلے گا ورنہ پروگرام سے آؤٹ ہو جائے گا

|| یا

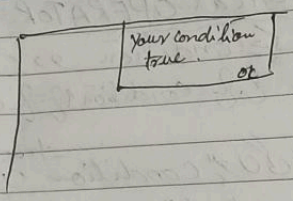
اس میں اگر دونوں میں سے ایک بھی condition ٹھیک ہو جاتی ہے تو یہ execute ہو جائے گا۔

// Logical Operater

```
let a = 5;
let b = 6;
if (a>2 && b>5).
{
alert ( both condition are true);
}
{
alert ( your ~~both~~ condition ~~are not~~ true);
}
```

Result →

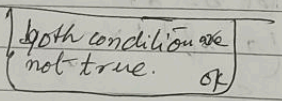
Result



اگر ایک condition false کر دیں تو

```
let a = 5;
let b = 6;
if (a>7&& b>5);
{
alert("both condition are true");
}
else {
alert("both condition are not true");
}
```

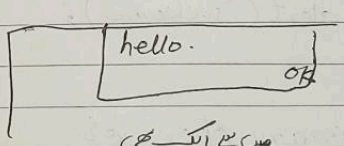
Result browser



|| OR

```
let a = 5;
let b = 6;
if (a>7 || b>3)
{ alert("hello");
}
else { alert("welcome");
```

Result



میں سے ایک بھی

اگر دونوں condition ٹھیک ہو جائے تو نتیجہ hello آئے گا۔ یہ value true پرنٹ کرے گا۔
اور اگر دونوں condition ٹھیک نہیں ہیں تو hello پرنٹ کر دے گا

جیسے

```
let a = 5;
let b = "6"
if (a>7 || b>9);
{alert("hello")
} else { (alert(welcome); }
```

اگر condition میں a، b سے چھوٹا ہے تو یہ true ہو جائے گا اور یہ چھوٹا نہیں ہے تو false ہو گا تو یہ welcome کو execute کرے گا۔

```
let a = 5;
let b = 6;
if (!(a>b)) // true
{
  alert("hello");
}
else{
  alert("welcome");
```

اگر
اندر والا بیان ہو گیا۔
کا مطلب نہیں کا ہے۔
!
اسکا مطلب ہے a بڑا نہیں
ہے b سے تو بات ہماری
درست ہو گئی تو یہ ہمیں
true واپس کرے گا اور
پرنٹ hello ہو گا

Ternary operator

اگر Condition ٹھیک true ہے تو بیچ والا کام کرے اور یہ ایک لائن کا کوڈ ہوتا ہے۔ اگر بیچ والا true نہیں ہے تو آگے والا کام کرے if اور else ایک ہی لائن میں لکھ سکتے ہیں۔

// ternary operator / Condition checked logically

~~let age = 20;~~ let age = 20;
~~let b = 6;~~ let result;
result = age>18? "adult": "not adult";
~~document~~ alert(result).

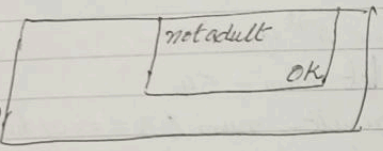
Result → adult
~~on~~ Popup ok

Ternary Operator

condition ? true output : false output;
age > 18 ? "adult" : "not adult";

let age = 10
let result;
result = age>18 ? "adult" : "not adult";
~~or~~ alert(result).

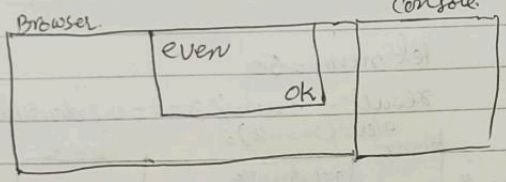
Result on ~~Console~~ Popup →



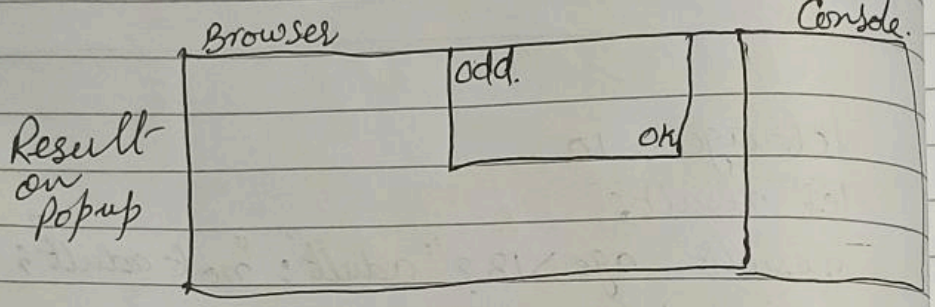
// Practice work.

let num = 40;
result = num%2===0? "Even": "odd";
alert(result);

Result on Pop up

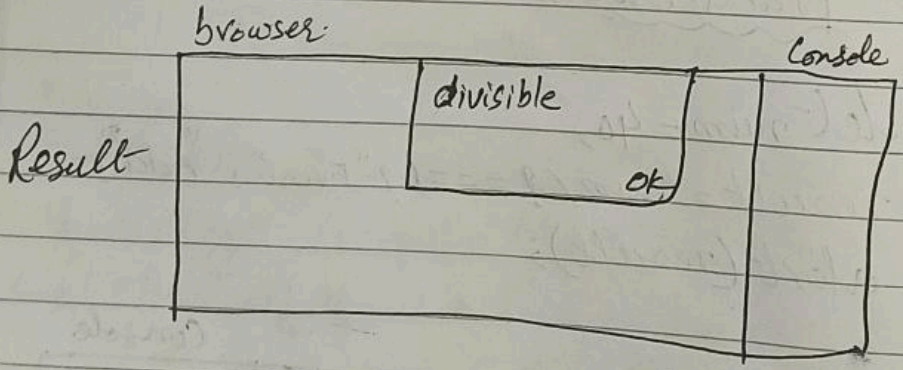


let num = 45,
result = num % 2 === 0 ? "Even" or "odd"
alert (result);



Practice 2nd question

② let num = 54.
result = num % 3 === 0 ? "divisible" or "not divisible"
alert (result);



let num = 50
result = num % 3 === 0 ? "divisible" or "not divisible"
alert (result);