Student: Zainab Asif JS2(028)

Intro.

Q. What is JavaScript?

A. JS is a programing language. We use it give instructions to the computer.

Input (code) → JS → Computer → Output
Result
1+2=3 = print

↓

Instruction

for example 1
2
+

Simple way to run codes should be use Browsers.

Q. What is Console?

A: Console is use for run codes and statement

Q. What is alert?

A. Use for run pop-up. for example.

alert ("Zainab Asif");

It's direclly run in JS. (it's a simple way to run code)

- It's a temporary type of write JS in console.

1st JS Code

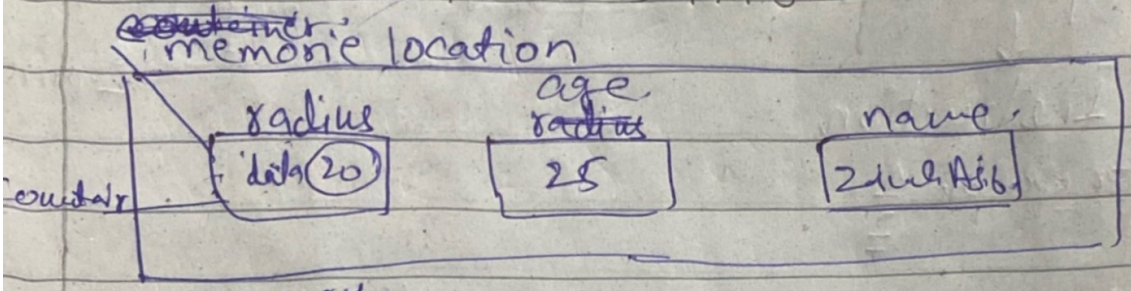
Console.log is used to log (print) a message to the console.

Zainab Asif JS2-02

for example:-
console.log("Zainab Asif");
Zainab Asif

for link (html and JS file).
use in html file befor closing body tag.
<script src="first.js"></script>

Varibles in JS



Varibles store deta and this deta will changed means veribles are changable.
for example:-
veribles:-

age = 24
name = Zainab Asif.
price = 250.14

Full Name = "Zainab Asif"
console.log(full Name)

full Name = "Hina Imtiaz"
age = 24;
price = 99.99;
console.log(price);

Zainab Asif JS2-028

fullName = XY2 <--> String value.
age = 24 <--> Number value.
price = 99.9 <--> Number value.
radius = 14 <--> " "
a = null <--> Absent value.
y = undifined <--> don't know

As a good programer choose nice varible names.

is Follow: ~~false~~ true/false; <--> boolen value.

In Jawascript you save directly variables value

Assing meni Opreater:-

a = b ~~sat~~ save value, write to left.

Variables Rules:-

- Variable names are case sensitive; means "a" and "A" is differenit.
- Only letters, digits, underscore (_) and $ is allowed. (not even space).
- Only a letter, underscore (_) or $ shaild be 1st elarcter.
- ~~Reve~~ Reserved words cannot be varible name.

Zainab Asif. JS2-028

- fullname = "Zainab Asif"; } Case sensit
  fullName = "Hina Imtaiz"; } Two variable
  console.log(fullname); } shows bo
  console.log(fullname); } are print.

- full name = "Hina Imtaiz"; → show err
  ↑ unexpected keyword why
  Space is not allowed in veriables.
- full name $ $ = ✓
  full name_ = ✓
  full name@ = X @ symble case no allowed.

- fullname = ✓
  _fullname = ✓
  $ fullname = ✓
  1234 fullname = X

- Reserved word not allowed
  means fixed word are not allowed.
  for example console is a fixed word its not ~~or veriable~~ use in veriab name.

- Cases of vanble.
- ~~full~~

Zainab Asif JS2-028.

Cases of variables:-

fullName — Camel Case ✓.
fullname
full_name ✗ — Snake case ✗
full-name ✗ — Kababcase ✗
FullName ✗ — Pascal case ✗
we use only camel case.

Key words in JavaScript.

let, const and var:-

var : Variable can be re-declared and updated. A global scope variable

let : Variable cannot be re-decleared but can be updated. A block scope variable.

const : Variable cannot be re-declared or updated. A block scope variable.

const are not change.
only use let and const variable.
Block scope means { } 1 block.
{
}

Zainab Asif JS2-028

{let a=5;
console.log=(a);} ✓ print 5

{let a=5;
let a=10;
console.log=(a);} ✗ error.

{let a=5;
console.log=(a);} ✓

{let a=10;
console.log=(a);} ✓

Both value are print.

Blocks are working like this.
Var is old version. let and const are new version.

Data Types:-

For example facebook data.

- Name — String Type data store ten
- Number of follower — That is going to be a number / numeric.
- follower ⟷ Bollen Type. data. True / false

Zainul Asif JS2-028.

Types of Data in JS.

- Number
- String
- Boolean
- Undefined
- Null
- Bigint
- Symbol

pimative (7)

- let age = 25;
  type of age.
  number.
  let price = 99.9.
  } Number.

- let name = Hina.
  type of name.
  } String.

isFollow = true.
type of isFollow
} boolean. true | false.

Non-primative.

objects.
(Array, Function)

```
//array
let info = [5, "Hina", tws
//document.write(info);
document.write(info)
[2]);
//object
let student = {
roll no: 28,
name: "Zainul",
sub: Javascrip
}
```

Zainul Arif JS2-028

- let x = ◯ ? } Undifined.
  typeof x is undifined. { By defult all varibles are undifined bcz value is miss

- let y = null
  typeof ~~null~~ is object. } Absent of an obje

- BigInt . Symbol } Not is normally use.

  let x = BigInt ("123");
  ~~typeof~~ typeof BigInt
  print 123n

  let y = Symbol ("Hello");
  typeof ~~Symbol~~ is Symbol.

Non-Primative.
↓
Objective
↓
Collections of value.

Zeeshan Asif JS-2-028

Student.
- name — string
- age — number
- marks — number
- ispass — boolean

} primative

↓ Collection

↓ Object

in objects key : value pairs in Object.

{ age : 24 ;
name : "Hina" ;
}

object;

let student = {
fullname : "Hina",
age : 25,
cgpa : 8.10,
ispass : true,
};

student["age"] : student["age"] + 1;

console.log(student["ispass"]); → print.

→ for value change.

Student["age"] = student["age"] + 1 ;

let is update. ✓
const is not update. ✗
Const key is changeable. ✓

Zainab Asif JS2-028

Q. Create a let/const object called "product" to store information shown in website pic?

```
let product = {
    title: "Ball",
    rating: 4,
    offer: 5,
    price: 350,
    available: true,
};
```

console.log(product); → print.
typeof
object.

Q2. Create a let object called "profile" to store information shown in web page?

```
let profile = {
    username: "@Teach Force";
    isFollow: true,
    followers: 548,
    following: 1002,
};
```

print in console.

Zaid Asif JS 2.028

~~console.log~~ console.log (type of profile);
object.

↓

consol.log (type of "username");
String. etc. etc.

Chapter : 2.

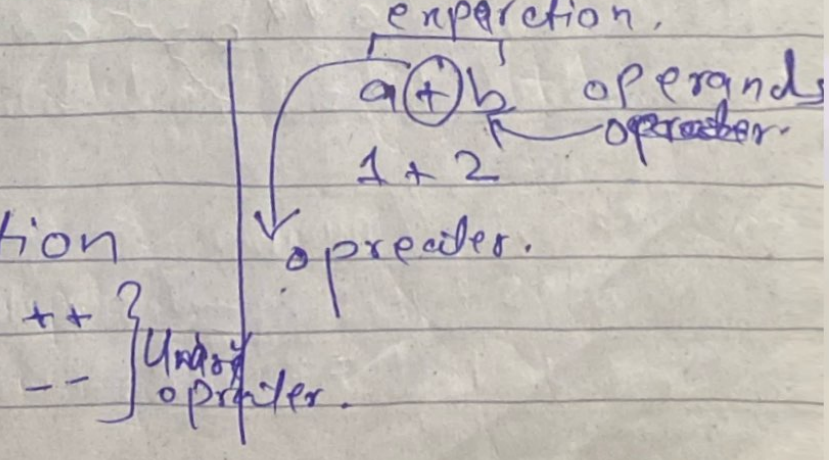
Comments in JS
Part of code which is not excuted.

- // } Single line.
- /* ____ */ } multi lines.

Oprators in JS
Used to perform some opreate on data.

Aritmetic Opreators.

+, -, *, /
- Moduls
- Exponeniation
- Increment ++ } Unary oprater.
- Decrement -- }



Zainab Asif JS2-028

```
let a = 2;
let b = 5;
```

a++ (post) a-- (post)
++a (pre) --a (pre)
post is change value first and pre first change value.

~~console.log ("a+b");~~

```
console.log("a=", a, "& b=", b);
"          ("a + b = ", a+b);
"          ("a - b = ", a-b);
"          ("a * b = ", a*b);
"          ("a / b = ", a/b);
```

Modulus → % a % b a ⟌ b remainder

5 % 2 → = 1

$2\overline{)5}$ → 2, 4, 1 remainder.

```
console.log("a % b = " a % b);
("a ** b = " a ** b); // 2 * 5
```

Zainab Asif

JS2-028

Exponentiation — **

$2^2 = 2 \times 2$

$2^3 = 2 \times 2 \times 2$

$2^4 = 2 \times 2 \times 2 \times 2$

$3^4 = 3 \times 3 \times 3 \times 3$

Assignment Opreaters:-

= += -= *= %= **=

left = right

a+=1 → a = a+1

a+4 (compact) → a = a+4 → short syntax

```
let a = 5;
let b = 2;

a += 4; // a = a+4
console.log("a: ", a); // 9 → print. (9)

a -= 4; // a = a-5
console.log("a = ", a); // 1 → print. (1)

a *= 4; // a = a*5
console.log("a = ", a); // 20 → print (20)

a /= 4; // a = a/4
console.log("a = ", a); // 1.25 → print. (1.25)

a %= 4; // a = a % 4
console.log("a = ", a); // 1 → print (1)
```

Zainab Asif JS02(028)

```
a**=4; // a = a**4
console.log("a = ", a); // 1 -> print.
```

(1)

Comparison Opreators:

Equal to == Equal to & type ===
Not equal to != Not equal to & type !==

>, >=, <, <= (< lessthan) > (graiterthan)

```
let a = 5;
let b = 2;
console.log("5 == 2", a == b); // false. -> print.

let a = 6;
let b = 6;
console.log("6 == 6", a == b); // true -> print

let a = 5;
let b = 2;
console.log("5 != 2", a != b); // true -> print.

let a = 5; // number  <- convert.
let b = "5"; // string -> number
conde.log("a == b", a == b); // true -> print.
```

Zainab Asif JS-2 (028)

```
let a = 5;
let b = "5";
console.log("a === b", a !== b); // false → print.

let a = 5;
let b = 10;
console.log("5 <= 10", a <= b); // false.

let a = 5;
let b = 10;
console.log("5 <= 5", a <= b); // true.
```

* Logical operators.
  - logical and &&
  - logical OR ||
  - logical Not ! → opposite work

cond1	cond2	Result &&	Result ||
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

```
let a = 6;
let b = 5;
let cond1 = a > b; // true
let cond2 = a === 6; // true.
console.log("cond1 && cond2 =", cond1 && cond2);
```

Journal Entry JS 02 (028)

Conditional Statements.

To implement some condition in the code

If Statement:-

```
let color;
if (mode === "dark-mode") {
    color = "black";
}

let age = 30;
if (age >= 18) {
    console.log ("can vote"); → print
}
if (age < 18)
{
    console.log ("You CAN NOT vote"); → print.
}

let mode = "dark";
let color;
if (mode === "dark") {
    color = "black";
}
if (mode === "light") {
    color = "white";
}
console.log (color); black. → print.
```

Zainab Asif JS.02(028

if else:-

```
let mode = "light";
let color;
if (mode === "dark"){
        color = "black";
    } else {
        color = "white";
    }
console.log(color);

let age = 25;
if (age >= 18){
    console.log ("vote");
    } else {
        console.log ("not vote");
    }
//odd or even.

let num = 10;
if (num % 2 === 0){
    console.log (num, "is even");
    } else {
        console.log (num, "is odd");
    }
```

Zainab Asif JS-02 (.028)

• Syntax → means rules.

else - if Statement:-

```
if (age < 18){
    console.log ("junior");
} else if (age > 60) {
    console.log ("senior");
} else {
    console.log ("middle");
}
```

=

```
let mode = "dark";
let color;

if (mode === "dark") {
    color = "black";
} else if (mode === "blue") {
    color = "blue";
} else if (mode === "pink") {
    color: "pink";
} else {
    colo = "white";
} console.log (color); → black print.
```

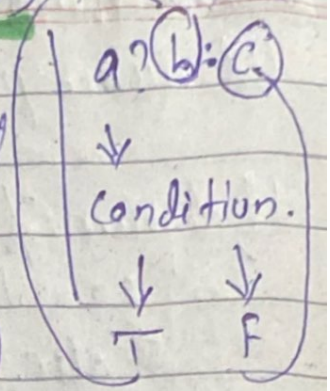
Journal 9: If else

JS02(028)

Ternary operator:-

condition (?) true output (:) false output

```
let age = 25;
let result = age >= 18 ? "adult" : "not adult";
console.log(result);
```




```
let age = 25
age >= 18 ? console.log("adult") : console.log("not adult");
```

(Generally not use in print). // simple.

* mdn developer web (for reading).

Practice:-

Q1. Get user to input a number using prompt ("Enter a number:"). Check if the number is a multiple of 5 or not

```
let name = prompt("XYZ !");
console.log(name);
```

Scanned with CamScanner

Decision Making — JS-02 (028)

```
let num = prompt("enter a number:");

if (num % 5 === 0){
    console.log(num, "is a multiple of 5");
}else{
    console.log(num "is Not a multiple of 5")
}
```

Q. Write a code which can give grades to students according to their scores:

90 - 100, A
70 - 89, B
60 - 69, C
50 - 59, D
0 - 49, F

```
if (score >= 90 && score <= 100) {
        grade = "A"
}else if (score >= 70 && socore <= 89) {
    =
}
```

Jainesh ji: if prompt("enter your score (0-100)"); JS-02(02)

```
let score = 75;
let grade;

if (score >= 90 && score <= 100){
    grade = "A";
} else if (score >= 70 && score <= 89){
    grade = "B";
} else if (score > 60 && score <= 69){
    grade = "C"
} else if (score >= 50 && score <= 59){
    grade = "D";
} else if (score >= 0 && score <= 49){
    grade = "F";
}
console.log("according to your scores,
your grade was A");
```

```
let age = 16;
let result = age >= 18 ? "adult" : "not adult";
console.log(result);
```

(25)

```
let age = 16;
age >= 18 ? console.log("adult"):
            console.log("not adult");
```

not prefer

Jainal Abid JS 02/02/28

Chapter: 3

Loops in JS

Loops are used to execute a piece of code again and again.

(for Loop) initilize stopping cond updation.

```
for (let i=1; i<=5; i++){
    console.log("  ");}
```

1. Initilize
2. Stopping condition
3. updation

} Three condition.

```
for (let count=1; count<=5; count++){
    console.log("Tech force."); //$ exec
}
console.log("loop has ended");

let sum=0;
for(let i=1; i<=10; i++){
    sum=sum+i;} // sum=
console.log("sum=", sum);
console.log("loop has ended");
```

```
let tabno=parseInt(prompt("Enter Table no of your choice:"
for (i=1; i<=12; i++)
```

Final Ajf JS-02-0228

```
{ document.write('${tabno} x ${i} = $
                 { tabno * i }');  // 3x1=
}
```

```
for (let i = 1; i <= 5; i++)
   { console.log("i = ", i); }
 // Calculate sum of 1 to 100.
   let sum = 0;
   let n = 100;
 for (let i = 1; i <= n; i++)
   { sum = sum + 1; // sum = 15
   }
     console.log("sum = ", sum);
     console.log("loop has ended");
```

Print 1 to 5

```
for (let i = 1; i <= 5; i++){
   console.log("i = ", i);}
```

{Recall}

Zainab Asif JS-02 (028

(2) Loop (infinite Loop): A Loop that never en
x never use in programing.

stopping ending condition → always True.
called infinite loop.

(3) While loop:- stop → F.
while (condition) { // do some work }.

let i = 1;
while (i <=5) { console.log ("i = " ,i);
i++; }.

(4) Do-while loop:-
do { // do some work } while (condition)

let i = 20; 1 for example.
do {
console.log ("Tech force");
i++; } while (i <= 10);

(5) for-of loop
(6) for-in loop

Zeinal Asif JS-02 (028)

for (let val of strVar) {String value}

// for of loop
let str = "Techforce"; //iterator → characters

for (let i of str) { console.log ("i =", i)

for of loop: » example:
let str = "JavaScript";

(for of):
only use for string and arrays
not use for objects

let size = 0;
for (let val of str) { console.log ("i =", i);
size++; }
console.log ("string size =", size); //10

for..in loop.
~~for (let key~~
~~for of key~~

(Use for objects)
arrays.

let student = {
name : "Zainal Asif";
age : 43,
cgpa : 7.5,
isPass: true };
for (let key in student)
console.log ("key =", key, "value =", student

Zainal Asif JS02 (028)

Strings in JS:-

Strings is a sequence of characters used to represent text.

Create String:-

let str = "Tech Force";
0123 45678 → (Individual position) pair Index

String length:-

str.length.

String Indices:-

str[0], str[1], str[2]

// Strings.
let str = "Tech Force";
console.log(str[2]; // C } → Normal string.

Template literals:-

A way to have embedded expressions in strings

"this is a template literal".

String Interpolation.

To create strings by doing substitution of

Zaina Q. Asif JS-02 (028)

placeholders.
"string text ${expression} string text".

// Template literals.
let specialstring = `This is a template literal`;
console.log (type of special string);

for example -
let ob = {
item: "beg",
price: 100, };
console.log ("the cost of", obj.item, "is", obj.price, "rupees");

↓

Single String // Template.

let output = `the cost of ${obj.item}
is ${obj.price} rupees`;
console.log (output);

let specialString = `This is a template literal
${1 + 2 + 3}`;
console.log (specialString); ↓ value create and then convert string

Zainab Arif JS02 (028)

// Template literals

console.log ("Tech force\nPakistan");

escape characters:-

\n for next line.

\t for tab space.

let str = "Tech \t force \n Pakistan"; //

console.log (str.length);

String Method:-

These are build in functions to manipulate a string.

- str.toUpperCase() → new string create with new value.
- str.toLowerCase()
- str.trim() // remove while spaces.

let str = "Tech forceofPakistan";

~~str.toUpperCase();~~

let newStr = str.toUpperCase();

console.log (str);

console.log (newStr);

string are immutable. ↓ (change X)

let str = "Tech force of Pakistan";

str = str.toUpperCase();

console.log(str);

change to upper case.

Zeinab Rafiq JS-02 (02)

- str.trim() // starting and ending spaces are removed

let str = " Quaide Azam ";
console.log(str.trim())

- str.slice(start, end?) // returns part of string.

let str = "0123456";
console.log(str.slice(1, 3)); → Not include (6)
Third index is inclusive that's why Third string is not print. not print 3 (string) Non-inclusive.

- str1.concat(str2) // join str2 with str1.

let str1 = " ";
let str2 = " ";

str1.concat(str2); let res = str1 + str2;
console.log(res);

let res = "I am learning from 'Tech force of
let res = "Hello 123";

• str.replace(searchVal, newVal)

```
let str = "[illegible]";
console.log(str.replace(" ", "_"));
```

• Str.charAt(idx):-

```
let str = "I      ";
console.log(str.charAt(2)); //
```

```
let str = "I Love JS";
[illegible] str = str.replace("I", "U");
console.log(str); //
```

Q. Prompt the user to enter their full name. Generate a user for them based on the input. Start username with @, followed by their full name and ending with the full name length.

eg. username= " ", username should be "@ 10"

username = "@" + fullName + fullName.length; (concat)

```
let fullName
prompt("enter your fullname without space");
let username = "@" + fullName + fullName.length;
console.log(username);
```

Zainab Ali

JS02(028

- Splice(): change orignal array (add, remove, rep

- starting indx
- del count
- New(E/1 ...)(element)

```
let arr = [1,2,3,4,5,6,7];
// arr.splice(2,2,101,102);
// Element Add Element.
   arr.splice(2,0,101);

// 1) delete elemement
   arr.slice(3,1);

// Replace Element
   arr.slice(3,1,101);

   arr.splice(4);
   (3) [5,6,7).
```

Q Create an array store companies ->

Remove the first company from the array

Remove ___ and ___ in its place.

- Add ___ at the end.

Zainab Asif JS-02 (028)

1(i) companies = [" ", " ", " ", " ", " ", " "]

//companies.shift ( );
// companies.splice (2, 1, " ");
companises.parush (" ");

Note: - PoP -> end delete.
shift -> Start delete.
splice -> Replace.
Push } -> add (Ending adding)
Unshift } -> store (Starting adding)

Zainal Asif JS-02(02

Array :-

Collections of items :-

let heroes = ["Hazrat Muhammad", "Hazrat Ali", "Hazrat Abu Bakar", "Hazrat Usman"];

let marks = [97, 89, 70, 69, 58];

let info = ["Zainal", 43, "Karachi"];

for example :-

- let marks = [97, 89, 70, 69, 58]; (key: Index, value)
  console.log(marks); -> print
  console.log(marks.length); //prop

- let heroes ["Hazrat Muhammad", "Hazrat Ali", "Hazrat Abu Bakar", "Hazrat Usman"]
  console.log(heroes); -> print.

- (Array is not a type of array is object means special type of object).

- (Turn on to object or behave like a obj

Zainab Asif JS-02 (028)

Array Indies:

arr [0], arr [1], arr [2]...

0	1	2	3	4
97	89	~~70~~ 60	69	58

→ lines means single line.

* [arr = [2]; for change of value it is possible only in array not strings.

Strings are immutable. } Noted:
array are mutable ✓ }

Looping over an Array:-
Print all elements of an array.
let heroes = [" ", " ", " ", " ", " "];

Loops → iterables (string, objects, array)
↓
(collection of items)
(for loop, do loop, while loop)

1	2	3	4	5
0	1	2	3	4

for example:-

for loop :- length (stoping condition).

```
for (let idex = 0; idx < arr.length; indx++)
{
}
```

& not equal to index.

Zaina Asif JS-02(o)

= let heroes [" ", " ", " ",
for (let idx = 0; idx < heroes.length; idx++
console.log (heroes [idx]); }

↓

(Standrad Style)

// for loop

// for of loop.

for (let hero of heroes) {
console.log(hero);

= let cities = [" ", " ", " ", " ", "
for let city of cites){
console.log(city); }.

↓

console.log (city.toUppesCase()); → for pri
uppercase.

= Practice:-

Q. for a given array with marks of stude
[85, 98, 44, 53, 68, 36]
0, 1, 2, 3, 4, 5

sum / total = 85, 98, 44, 53, 68, 36

sum / 6 = avg

Javinal Asif JS-02(028)

```
let mark: [85, 98, 44, 53, 68, 36];
let sum: 0;
```

↓

```
sum = mark[0] + mark[1] + mark[2];
```

its normal. ✗

Loop add ↓

```
for (let val of marks){
    console.log(val);
}
for (let val of mark){
    sum += val;
}
```

Logical programme.

```
let avg = sum / marks.length;
console.log(avg marks of the class = ${avg});
```

↓

We use here String, Veribles and data types arrays, loop, + equal to and divide opreaters.

Q. For a given array with prices of 5 items [300, 450, 250, 550, 680,] all items have an offer of 10% off on them. Change the array to store final price after applying offer.

300 → 30 off = 300 - 30 = 270

900 - 90 off = 900 - 90 → 810

JS-02(028

Zainab Asif

~~for~~ (let val of items).{
    offer: val */10;
val: val - offer}

for of loop se
we directly valu

let items: [300, 450, 250, 600, 750,];

for (let value of items){
    console.log(value);}.

let idex = 0;
for (let val of items){
console.log('value at index ${idx}');
let offer = val/10;
idx++;} items[idx] = items[idx] - offer;
console.log('value after offer = ${val}'; idx++
item s[i]}

// for let (let i = 0; i < items.length; i++)
}

let i = 0;
for (let val of items) {
let offer = val/10;
items[i] = items[i] - offer;
console.log('value after offer = $ items[i]
i++;}

Learning JS-02 (028)

```
for (let i = 0; i < items.length; i++) {
  let offer = items[i] / 10;
  items[i] -= offer; }
  console.log(items);
```

Array in JS. Methods :-

Push(): add to end.

Pop(): delete from end and return.

toString(): converts array to string.

for example.

```
let food items = ["Apple", "Orange", "Kiwi", "Banana"]
food items.push("Chips", "Pizza", "Burger");
console.log(food items); → print result add
                           items in end.
let food item = [" ", " ", " ", " "];
~~food items.pop();~~
console.log(food item);
let deleted item = food items.pop();
console.log(food items);
console.log("deleted"; deleted item);
```

Zainab Asif JS-02 (028)

let food item = [ ];
console.log (food Items);
console.log (food Items.to String ());
console.log (food Items);

↓

to string not change array it's return new array.

let marks = [98, 36, 28, 29,];
mark.toString ();

- concat ( ): join multiple arrays and return result.
- unshift ( ); add to start -> (push)
- shift ( ); delete from start and return (pop)

let marvel_heroes = ["thor", "Batman", "Iron
let dc Heroes: ["superman spiderman"
let heroes. concat (dc Heroes);
let heroes: marvel Heroes. conceil (dc Hero
console.log (heroes);

Zainab Asif JS02(028)

```
let marvelHeroes = ["Spiderman", "Ironman", "thor"
marvelHeroes.unshift( );
      "      .shift( );
console.log('deleted", val);
```

Slice( ): return a piece of the array.

Slice (start idx, end idx)

↓

original Array not changable.

```
marvelHeroes = ["Spiderman", "ironman", "thor"
                "          ", "      "];
console.log (marvelHeroes);
console.log (marvelHeroes.slice(1, 3));
```

- last index not include because (last-1) index ending non inclusive.

Splice( ): change orignal array (add, remove, replace).

start indx. }
del. indx. }
New Elt. }

- [1, 2, 3, 4, 5, 6, 7]
  0, 1, 2, 3, 4, 5, 6
- del.splice.(2, 2).
- add.splice.(2, 2, 101, 102)

[1, 2, 3, 4, 5, 6, 7]
101, 102, 5, 6, 7

Zainab Asif

JS02 (028)

```
let arr= [1,2,3,4,5,6,7];
arr.splice(2,2,101,102);
```

// Element Add

```
arr.splice(2,0,101);
```

go second item and delete 0 item and add 101.

// Delete Element

```
arr.splice.(3,1);
```

// Replace Element

```
arr.splice(3,1,101);
```

- if I pass index 4 it's start act like arry.
- if I pass 0 index then not deleting in array.

1. Create an array to store companies ->
" " , " " , " " , " " , " "
- Remove the ~~first~~ company from the array
- Remove "3rd" and Add "FFC" in the array.
- ~~Remove~~ Add "Nestle" at the end.

Zainab Asif

JS-02 (028)

let compaines = ["Netflex", "Google", "Microsoft", "Drive in", "Ubbro", "Abbol"];

- compaines.shift();
- compaines.splice(2, 1, "FF");
- compaines.push("");

pop - delet end
shift - start delet.
Splice - Replace.
push - end add }
Unshift - start add }

Zainab Asif JS02 (028)

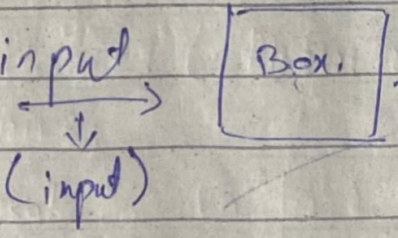
Functions:-

Block of code that performs a specific task, can be invoked whenever needed.
call.

console.log
.toUpperCase } Function.
.push

```
console.log(" ");
"String".toUpperCase()
"Array.push
```

( ) → parentisise. Function invoke mean call function.



- We made our own function.

Function Defintation (invoke) Function Call

```
function functionName() {Block of code
// do some work
}function functionName(param1, param2...){
// do work  calculation
return
```

```
functionName();
```

Zainalfsit JS02(628)

```
function myfunction( ){
    console.log ("Welcome to Tech Force Of Pakistan")
    console.log ("We are learning JS:)");}
```

// Function call

```
myfunction( );  } for Two Type call.
myfunction( );  }
```

// redundancy -> repeat (function safe as for (redundacy).

function help us to repeat.

```
function myfunction (msg) {        // input variable
    console.log (msg);}
                                   //parameter -> input.
myfunction ("I Love Pakistan");    // argument.
```

- function defination use veriables it called parameters
- In function call pass value its called arguments.

Same.

Seprate from ,

NaN (not anumber) error. not valid.

Zaina Asif JS-02 (02

// Function -> 2 numbers sum

= function (sum (x, y){
    console.log (x+y);}

for return value. (return value only return one v

function sum (x, y){ // local variables -> scope
    s= x+y;
    return s;}
=sum (3,4); let val = (3,4);
    console.log (val);

function sum (x,y){
s= x+y;
console.log ("before ~~return~~ return");
return s;
console.log ("after return");} Nai ~~execute~~ execute

- function parmetes like local variable of function and they have block scope.
- they live only function blocks.
- function work ups ~~to~~ to down.
- return is generaly last line.

Trained Artist JS02 (028)

Arrow Functions:-

```
//sum function
function sum(a,b){
  return a+b;}
//multiplication function
function mul(a,b){
  return a*b;}
```

→ These function write in different function means. compact style funtion like we write in Arrow function.

- How can we write Arrow function.
  Compact way of writing a function

```
const functionName = (param1, param2 ...) => { // same work }
```

input

= equal two, > greater then sign without space like ⇒

↓ arrow function

↓ arrow function sign.

```
function sum(a,b){
  return a+b;}
(a,b)=>{
  console.log(a+b);};
```

← arrow sum

but it's not excute because variable is not done.

```
function sum(a,b){
return a+b;}
const arrowSum=(a,b)=>{
console.log(a+b);
}
```

Modern Java functions

Coding Test JS02(028)

// multiplication function

```
function mul (a,b){
return a*b;}
```

↙ function variable.

```
const arrowMul = (a,b) => {
return a*b;};
```

- We call arrow function for sum and multiplication fuction.

```
const printHello = () => {
console.log("hello");};
```

Q. Create a function using the "function" keyword that takes a String as an argument and return the number of vowels in the string.

```
function countVowels (str) {
// "Tech force of Pakistan", count:
for (const char of str) {
    console.log(char);}}.
```

```
funtion countVowels (Str) {
let count = 0;
for (const char of str) {
```

Learning Asit JS02(028)

```
If (char === "a" || char === "e" ||
    char === "i" || char === "o" ||
    char === 'u") {
        count ++; } }
*return (count);
  console.log (count); }; };
```

Q. Create an arrow function to perform the same task.

```
const count Vow = (str) => {
let count = 0;
for (const char of str) {
  if {
      char === "a" ||
      char === "e" ||
      char === "i" ||
      char === "o" ||
      char === "u" ) {
        { count ++;
          }}
  return count; };
```

Zaimul Asif JS-02 (028)

For Each Loop in Arrays :-

arr. forEach (call Back Function)

Call back Function : Here, it is a function to execute for element in the array

* A call back is a function passed as an argument to another function.

for example-

```
let arr = [1, 2, 3, 4, 5]; // value of each il
arr.forEach (function print (val) (val) {
    console.log (val); });
```

=

```
let arr = [1, 2, 3, 4, 5]
arr.forEach ((val, idx, arr) => {
    console.log (val); });
```

val.toUpperCase(), idx, arr

- Three parameters use in for each
  - value . index . array.
  (val) . (idx) . (arr)
  not use for strings.

What is Higher Order Function/Methods.

s. For Each is a Higher Order function (HOF) Heigher order Methods (HOM).

Zainab Asif JS-02 (028)

Q For a given array of numbers, print the square of each value using the forEach loop. ✍

```
let num = [2,3,4,5,6];
num.forEach((num)=>{
  console.log(num * num);//num **2});
```

another way

```
let num = [89,74,25];
let calcSquare = (num)=>{
  console.log(num * num);};
num.forEach(calcSquare);
```

Some More Array Methods:-

Map ← (Simpler forEach Method)

Create a new array with the results of some operation. The value its callback returns are used to form new array.

Difference between Map and forEach method. Maps are create a new array and forEach calculate work for us.

Kainat Asif JS:02 (028)

arr.map(callbackFnx(value, index, arr

- Three parameter use. val / idx / arr generally use value in forEach and map(callback)

for example.

```
let nums=[12, 39, 43];
num.map((val)=>{
  console.log(val);});
```

```
let nums=[12,39,43];
let newArr= nums.map((val)=>{
  return val;        // val*val;
                     val x 2;
});
```

- Map is used to create new Array using some returned value based on each value which is used in the end.

= Simple forEach use for normally print calculation and Map use value for new array.

= Filter Method:-

Create a new array of elements that give true for a condition / filter.

eg. all even elements

(= filter value)

Zainab Asif JS-02 (028)

let newArr = arr.filter((val) => {
return val % 2 === 0;})

condition → true (new array safe)
↓ false (new array not store)

for example:-

let arr = [1,2,3,4,5,6,7];
let evenArr = arr.filter((val) => {
return val % 2 === 0;});
console.log(evenArr);

for even value
↓
val % 2 !== 0;
↓
for odd value

let arr = [1,2,3,4,6,7];
let evenArr = arr.filter((val) => {
return val > 3;});
console.log(evenArr);

Noted original Array is not change

Reduce Method:-

Performs some operations and reduces the array to a single value. It returns that single value.

Zainab Asif JS-02 (028)

```
const array1 = [1,2,3,4];
// 0+1+2+3+4
const initialValue = 0;
const sumWithInitial = array1.reduce(
(accumulator, currentValue) => accumulator
+ currentValue, initialValue,);
console.log(sumWithInital);
// Expected output: 10
```

(MDN) example:

```
let arr = [1,2,3,4]; // 10
const output
= arr.reduce((res, curr) => {
return res + curr; });
console.log(output);
```

→ This code + array means(add) (result and current add)

```
prev, current
arr.reduce((res, curr) => {
    return res + curr;
})
```

If i have an array [1, 2, 3, 4]

Previous, Current

```
const output = arr.reduce
(prev, curr) => {
return prev > curr ? prev : curr;
});
console.log(output);
```

rest → 1	Add 3
curr → 2	3

This code to find out largest number

Zainab Asif JS-02 (028)

Q. We are given array of marks of students. filter out of the marks of students that scored 90+.

```
let marks = [97, 64, 32, 49, 99, 96, 86];

let toppersmarks = filter((val) => {
    return val > 90;
})

console.log(toppers);
```

Q. Take a number n as input from user. Create an array of numbers from 1 to n. Use the reduce method to calculate sum of all numbers in the array. Use the reduce method to calculate product of all numbers in the array.

```
let n = prompt("enter a number");

let arr = [];
for (let i = 1; i <= n; i++) {
    arr[i-1] = i; // 1(0), 2(1), 3(2), 4(3)
}
console.log(arr);
```

Zainab Asif JS-02 (028

```
let sum = arr.reduce((res, curr) => {
  return res + curr; });
console.log (sum = ", sum);
```

factorial of 4 → 1 x 2 x 3 x 4 } factorial
fact of 5 → 2 x 2 x 3 x 4 x 5 } of N

1 x ..... n
factorial

```
let factorial = arr.reduce((res, curr) => {
  return res curr; });
console.log ("factorial = ", factorial());
```