JS CHEATSHEET

→ Javascript Basics
Set of Javascript basic syntax to add, execute and write basic programming paradigms in Javascript.

1. On Page Script
→ Adding internal javascript to HTML.
<script type = "text/javascript"> // JS code </script>

2. External JS File
→ Adding external Javascript to HTML
<script src = "script.js"></script>

3. Functions
→ Javascript Function syntax
function nameofFunction() {
    // function body
}

4. DOM Element
→ Changing content of a DOM Element.
document.getElementById("elementID").innerHTML = "Hello";

5. Output
→ This will print the of value of a in Javascript console.
console.log(a);

→ Conditional Statements
Conditional Statements are used to perform operations based on some conditions.

1. If statement

→ The block of code to be executed, when the condition specified is true.

```
if (condition) {
    // block of code to be executed if condition is True.
}
```

2. If-else Statement

→ If the condition for the if block is false, then the else block will be executed.

```
if (condition) {
    // block of code to be executed if the condition is true.
}
else {
    // block of code to be executed if the condition is false.
}
```

3. Else-if Statement

→ A basic if-else ladder

```
if (condition1) {
    // block of code to be executed if the condition1 is true.
}
else if (condition2) {
    // block of code to be executed if condition1 is false but condition2 is true.
}
else {
    // block of code to be executed if previous conditions are false.
}
```

4. Switch Statement

→ Switch case statement is Javascript

```
switch (expression) {
    case x: // code
            break;
    case y: // code
            break;
    default : // code
}
```

→ Iterative Statements (Loops)

Iterative statement facilitates programmer to execute any block of code lines repeatedly and can be controlled as per conditions added by the programmer.

1. For loop

→ For loop syntax in javascript

```
for (initialization; condition; updation;)
{
    // code
}
```

2. While loop

→ Runs the code till the specified condition is true.

```
while (condition) {
    // code
}
```

3. Do While loop

→ A do-while loop is executed at least once despite the condition being true or false.

```
do
{
    // code
    updation;
} while (condition);
```

→ Strings

The string is a sequence of characters that is used for storing and managing text-data.

1. charAt method

→ Returns the character from the specified index.

```
str.charAt(3);
```

2. concat method

→ Joins two or more strings together.

```
str1.concat(str2);
```

3. IndexOf method

→ Returns the index of the first occurence of the specified character from the string else -1 if not fwound.

```
str.indexOf('substr');
```

4. match method

→ Searches a string for a match against a regular expression.

```
str.match(/chapter \d+(\.\d)*/i;);
```

5. replace method

→ Searches a string for a match against a specified string or char and returns a new string by replacing the specified values.

```
str1.replace(str2);
```

6. search method

→ Searches a string against a specified value.

str.search('term');

7. split method

→ Splits a string into an array consisting of substrings.
str.split('\n');

8. substring method

→ Returns a substring of a string containing characters from the specified indices.
str.substring(0, 5);

→ Arrays

The array is a collection of data items of the same type. In simple terms, it is a variable that contains multiple values.

1. Variable

→ Containers for storing data.
var fruits = ["element1", "element2", "element3"];

2. concat method

→ Joins two or more arrays together.
concat();

3. indexOf method

→ Returns the index of the specified item from the array.
indexOf();

4. join method

→ Converts the array elements to a string.
join();

5. pop method

→ Deletes the last element of the array.
pop();

6. reverse method

→ This method reverses the order of the array elements.
reverse();

7. Sort method

→ Sorts the array elements in a specified manner: sort();

8. toString method

→ Converts the array elements to a String.
toString();

9. valueOf method

→ returns the relevent Number Object holding the value of the argument passed.
valueOf();

→ Number Methods.
JS math and number objects provide several constant and methods to perform mathematical operations.

1. toExponential method

→ Converts a number to its exponential form.
toExponential();

2. toPrecision method

→ Formats a number into a specified length.
toPrecision();

3. toString method

→ Converts an object to a string.
toString();

4. valueOf method

→ Returns the primitive value of a number.
valueOf();

→ Maths method

1. ceil method
→ Rounds a num to nearest up int.
ceil(x);

2. exp method
→ Returns value of e^x.
exp(x);

3. log method
→ Returns logarithmic value of x.
log(x);

4. pow method
→ Return the value of x to power y
pow(x, y);

5. random method
→ Return random number
random() // b/w 0 & 1.

6. sqrt method
→ Returns the square root of x.
Sqrt(x);

→ Dates

Date object is used to get the year, month and day. It has methods to get and set day, month, year, hour, minute, and seconds.

1. Pulling Date
→ getDate();

2. Pulling Day
→ getDay();

3. Pulling Hours
→ getHours();

4. Pulling Minutes
→ getMinutes();

5. Pulling ~~Minutes~~ Seconds
→ getSeconds();

6. Pulling Time
→ getTime();

★ For Events

→
```
element.addEventListener('event_name', () => {
    // code
});
```

→ Mouse Events

Any change in the state of an object is referred to as an Event. With the help of JS, you can handle events, i.e

how any specific HTML tag will work when the user does something.

1. click : Fired when an element is clicked
2. oncontextmenu : Fired when an element is right-clicked
3. dblclick : Fired when an element is double-clicked.
4. mouseenter : Fired when an element is entered by the mouse arrow.
5. mouseleave : Fired when an element is exited by the mouse arrow.
6. mousemove : Fired when the mouse is moved inside the element.

→ Keyboard Events

1. Keydown : Fired when the user is pressing a key on the keyboard.
2. keypress : Fired when the user presses the key on the keyboard.
3. keyup : Fired when the user releases a key on the keyboard.

→ Errors

Errors are thrown by the compiler or interpreter whenever they find any fault in the code, and it can be of any type like syntax error, run-time error, logical error, etc. JS provides some functions to handle the errors.

1. Try & catch

→ Try the code block and execute catch when error is thrown.

```
try {
    // code
}
catch (err)
{
    // code
}
```

→ Window methods
Methods that are available from the window object.

1. alert method
→ Used to alert something on screen.
```
alert();
```

2. blur method
→ The blur() method removes focus from the current window.
```
blur();
```

3. setInterval
→ Keeps executing code at a certain interval.
```
setInterval(() => {
    // code
}, 100);
```

4. setTimeout
→ Executes the code after a certain interval of time.
```
setTimeout(() => {
    // code
}, 100);
```

5. close
→ The window.close() method closes the current window
```
window.close();
```

6. confirm

→ The window.confirm() instructs the browser to display a dialog with an optional message, and to wait until the user either confirms or cancels.
window.confirm('Are you sure?');

7. Open
→ Opens a new window
window.open("https://atharvarajsinha.netlify.app");

8. prompt
→ Prompts the user with a text and takes a value. Second parameter is the default value.
var name = prompt("Enter the Name", "Atharva");

9. scrollBy
→ window.scrollBy(100, 0);

10. scrollTo
→ Scrolls to the document to specified coordinates.
window.scrollTo(500, 0);

11. clearInterval
→ Clears the setInterval. var is the value returned by setInterval call.
clearInterval(var);

12. clearTimeout()
→ Clears the setTimeout. var is the value returned by setTimeout call.
clearTimeout(var);

13. stop
→ Stops the further resource loading.
stop();

→ Query / Get Elements
The browser creates a DOM (Document Object Model) whenever a webpage is loaded, and with the help of

HTML Dom, one can access and modify all the elements of the HTML document.

1. querySelector
→ Selector to select first matching element.
document.querySelector("css-selector");

2. querySelectorAll
→ A selector to select all matching elements.
document.querySelectorAll("ess-selectors", ...);

3. getElementByTagName
→ Selects elements by tag name
document.getElementByTagName("element");

4. getElementByClassName
→ Select Elements by class name.
document.getElementByClassName("class");

5. getElementById
→ Select Elements by its id.
document.getElementById("id");

→ Creating Elemerds
Creates a new elements in the DOM.

1. Create a new element
→ Create a new element
document.createElement('spom');

2. createTextNode
→ Create a new TextNode
document.createTextNode('Text');