Q1. Create a basic HTML document. (All basic tags along with required attributes, also mention the tags that goes in head element).

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
      Content goes here
</body>
</html>
```

Above mentioned lines is the basic syntax of the HTML document.The various tags that can be included in an HTML document are:

a) <html> - defines the rot of an HTML document.
b) <head> - It is used to store the metadata .
c) <title> - Used to define the title of the webpage.
d) <body> - All the content that will be displayed on the webpage is defined in the body of the HTML document.
e) <h1> to <h6> - Used to represent HTML headings with h1 having the largest font size and h6 having the lowest.
f) <a>- The anchor tag is used to mention links in an html document with the help of href attribute
Click here

g)<img> - Used to display an image in an html document.
<img src=”image.jpg” alt=” “>

h)<p>-USed to define a paragraph in the document.

i)<table>- Used to create a table in the document. By using the border attribute, you can define a border to the table. By using <th> we can mention the table headings and by using <td> you can mention table data

j) - Used to create ordered lists.

k) - Used to create unordered lists.

Q2. Mention 3 ways of styling a document, along with cascading order (mention priority using comments).

The three ways are:

-Inline CSS
-Internal CSS
-External CSS

Inline CSS

Inline CSS is used to uniquely style a an element.

Hello

Internal CSS

It used to uniquely style an HTML page
It is defined in the <style> element in the head of the HTML document.

```
selector {
        Property:value;
 }
```

External CSS
The same CSS file can be used to style many documents.This can be done by defining the link tag in the head of the HTML document.
<link rel="stylesheet" href="style.css">

Q3)Explain CSS selectors: simple, combinator, pseudo-class(links/form/table), pseudo-elements and attribute selectors.

HTML selectors style the elements that you select.

a)Simple selectors

1)Element selector
It selects based on element name

```
h1 {
text-align: center;
color: blue;
}
```

2) id selector
The id selector uses the id attribute of an element to select the specific element.

```
#id1 {
  text-align: center;
  color: blue;
```

```
}
```

3)Class selector
The class selector selects HTML elements with a unique class attribute.

```
.name {
  text-align: center;
  color: orange;
}
```

4)Universal selector

The universal selector * selects all HTML elements on the page.

```
* {
  text-align: center;
  color: orange;
}
```

5)Grouping selectors
The grouping selector selects all the HTML elements with the same style definitions.

```
p {
  text-align: center;
  color: orange;
}
h1 {
  text-align: center;
  color: orange;
}
```

The above code can be written as

```
h1,p {
  text-align: center;
  color: orange;
}
```

2)Descendant selector
The descendant selector matches all elements that are descendants of a specified element.

```
div p {
  background-color: red;
```

}
All the p elements inside div will be selected.

3)Child selector
The child selector selects all elements that are the children of a specified element.

```
div >  p {
  background-color: red;
}
```

4)Adjacent sibling selector

The adjacent sibling selector is used to select an element that is immediately after another specific element.

```
div +  p {
  background-color: red;
}
```

5) General sibling selector

The general sibling selector selects all elements that are next siblings of a particular element.

```
div ~  p {
  background-color: red;
}
```

3)Pseudo classes

A pseudo-class is used to define a special state of an element.

For example, it can be used to:

- Style an element when a user mouses over it
- Style visited and unvisited links differently
- Style an element when it gets focus

```
selector:pseudo-class {

  property: value;
```

```
}
```

Anchor pseudo class

```
/* unvisited link */
a:link {
  color:red;
}
/* mouse over link */
a:hover {
  color: pink;
}
/* visited link */
a:visited {
  color: blue;
}



/* selected link */
a:active {
  color: green;
}
```

Pseudo classes and HTML classes

```
a.highlight:hover {
```

```
  color: maroon;

}
```

Hover on div

```
div:hover {

  background-color: blue;

}
```

4)Pseudo elements

CSS pseudo-element is used to style specified parts of an element.

```
selector::pseudo-element {

  property: value;

}
```

The ::first-letter pseudo-element is used to add a special style to the first letter of a text.

```
p::first-letter {
  color: blue
  font-size: 17;
}
```

5)Attribute selector

The [attribute] selector is used to select elements with a specified attribute.

```
a[target="_blank"] {
  background-color: yellow;
}
```

The [attribute~="value"] selector is used to select elements with an attribute value containing a specified word.

```
[title~="fruit"] {
  border: 5px solid brown;
}
```

Q4. Mention ways to create an array, access and manipulate it. How to recognise if a variable is an array?

Creating an array

Basic syntax

```
const arrayname = [item1, item2, ...];
```

Example:

```
const fruit=[“mango”,”banana”,”cherry”];
```

2)You can also create an array, and then provide the elements by referring to the index:

```
const fruit=[];
fruit[0]=”orange”;
fruit[1]=”kiwi”;
fruit[2]=”mango”;
```

3)By using the keyword new

const fruit = new Array("mango", "kiwi", "papaya");

Accessing array elements

This can be done by referring to the index value.

const fruits=[“mango”,”banana”,”cherry”];

let fruit=fruits[1];

document.write(fruit);

The full array can be accessed by referring to the array name

const fruits=[“mango”,”banana”,”cherry”];

document.write(fruits);

Manipulating an array

const fruits=[“mango”,”banana”,”cherry”];

fruits[1]=”apple”;

We can also use push,pop functions to add and remove elements from an array.

By using the typeof keyword we can know if a particular variable is an array or not.

Q5. Find the size of an array, access the first and last element of array and add new element using appropriate array property.

Finding size of an array

```
const fruits=["mango","apple","banana","raspberry"];

    let ans= fruits.length;

    document.write(ans);
```

Accessing first and last element of an array

```
const fruits=["mango","apple","banana","raspberry"];
        let first= fruits[0];
        document.write("The first element is "+first);
        document.write("<br>");
        let last=fruits[fruits.length-1];
        document.write("The last element is "+last);
```

Adding a new element in an array

1)Using push method

```
const fruits=["mango","apple","banana","raspberry"];
        document.write(fruits);
        document.write("<br>"+"After adding new element");
        fruits.push("orange");
        document.write("<br>");
        document.write(fruits);
```

2)Using unshift method

```
      document.write(fruits);
        document.write("<br>"+"After adding new element");
        fruits.unshift("orange");
        document.write("<br>");
        document.write(fruits);
```

3)Using splice method

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(2, 0, "Plum", "Jamun");
```

Q6. Mention different methods to add new element to an array.

1)Using push method

```
const fruits=["mango","apple","banana","raspberry"];
        document.write(fruits);
        document.write("<br>"+"After adding new element");
        fruits.push("orange");
        document.write("<br>");
        document.write(fruits);
```

2)Using unshift method

```
      document.write(fruits);
        document.write("<br>"+"After adding new element");
        fruits.unshift("orange");
        document.write("<br>");
        document.write(fruits);
```

3)Using splice method

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(2, 0, "Plum", "Jamun");
```

Q7. Mention different methods to delete element from an array.

Using pop()

```
const fruits=["mango","apple","banana","raspberry"];
        document.write(fruits);
        document.write("<br>"+"After deleting an element");
        fruits.pop();
        document.write("<br>");
```

```
document.write(fruits);
```

Using shift()

```
const fruits=["mango","apple","banana","raspberry"];

document.write(fruits);

document.write("<br>"+"After deleting an element");

fruits.shift();

document.write("<br>");

document.write(fruits);
```

Using splice()

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];

fruits.splice(0, 1);
```

Q8. Mention iteration methods that loop through every elements in an array along with their return value.

Using forEach()

```
let txt = "";
numbers.forEach(myFunction);
document.getElementById("demo").innerHTML = txt;

function myFunction(value, index, array) {
  txt += value + "<br>";
}
```

Using every()

The every() method checks if all array values pass a test.

```
const numbers = [35, 1, 7, 17, 29];
let allOver18 = numbers.every(myFunction);
```

```
function myFunction(value, index, array) {
  return value > 15;
}
```

Using filter()

The filter() method creates a new array with array elements that pass a test.

```
const numbers = [45, 4, 9, 16, 25];
const over18 = numbers.filter(myFunction);

function myFunction(value, index, array) {
  return value > 18;
}
```

Using map()

The map() method creates a new array by performing a function on each array element.

```
const numbers1 = [45, 4, 9, 16, 25];
const numbers2 = numbers1.map(myFunction);

function myFunction(value, index, array) {
  return value * 2;
}
```

Q9. Mention methods that return index(es) of array element(s).

Using indexOf()

```
const fruits=["mango","apple","banana","raspberry"];
document.write(fruits);
document.write("<br>");
let index= fruits.indexOf("banana");
document.write(index);
```

Using findIndex()

```
const ages = [3, 10, 18, 20];
ages.findIndex(checkAge);
function checkAge(age)
{  return age > 17;
}
```

Using lastIndexOf()

```
const fruits=["mango","apple","apple","raspberry"];
      document.write(fruits);
      document.write("<br>");
      let index= fruits.lastIndexOf("apple");
      document.write(index);
```

Q10. Mention methods used to check if element is present in array along with their return value.

Using includes()

```
const fruits=["mango","apple","banana","raspberry"];
fruits.includes(“banana”);
```

Q11. Mention methods used to access last element, part of array along with their return value.

```
const fruits=["mango","apple","banana","raspberry"]

let last=fruits[fruits.length-1];

document.write("The last element is "+last);
```

Using pop()

```
const fruits=["mango","apple","banana","raspberry"]

let last=fruits.pop();

document.write(last);
```

Using slice

```
const fruits=["mango","apple","banana","raspberry"]
```

```
let last=fruits.slice(-1);

document.write(last);
```

Using splice

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(3, "Jamun", "Plum");
```

Q12. Mention methods used to combine arrays.

Using concat()

```
const num1=[1,3,5];
const num2=[7,9,11];
document.write(num1);
document.write("<br>"+num2);
const number=num1.concat(num2);
document.write("<br>");
document.write(number);
```

Using push()

```
const arr1=[1,2,3];
const arr2=[4,5,6];
arr1.push(arr2);
document.write(arr1);
```

Q13. Sort a number array, array of objects based on string (eg: [{name: 'Zayn', age: 23 }, {name: 'Ana', age: 35 },...]) and reverse it.

```
const arr = [
        { name: 'Zayn', age: 23 },
        { name: 'Ana', age: 35 },
        { name: 'Asher', age: 10}
        ];

const arr1 = arr.sort((p, q) => q.name.localeCompare(p.name));
const arr2 = arr1.reverse();
console.log(arr2);
```

Q14. In array: [1, 2, [3, 4], [5, [6, 7]]], mention methods used to get following output: [1, 2, 3, 4, 5, 6, 7].

Using flat()

```
const arr = [1, 2, [3, 4], [5, [6, 7]]];
const arr1=arr.flat(2);
document.write(arr1);
```

OUTPUT:

1,2,3,4,5,6,7

Q15. In array: [1, 2, 3, 4, 5], mention method to replace the last element with the first element.

```
const arr = [1,2,3,4,5];
document.write(arr);
arr[arr.length-1]=arr[0];
document.write("<br> After replacing " +arr);
```

OUTPUT:

1,2,3,4,1

Q16. In array: ["It's a sunny day", "I want ice-cream"] using appropriate method get this output: ["it's", "a", "sunny", "day", "I", "want", "ice-cream"] (Hint: "A boy".split(" ") => ["a", "boy"])

```
const arr=["It's a sunny day", "I want ice-cream"];
const arr1=arr.join(' ');
const arr2=arr1.split(' ');
document.write(arr2);
```

OUTPUT:

It's,a,sunny,day,I,want,ice-cream

Q.17 Use appropriate methods to convert string to array and vice versa.

```
const arr=["red","blue","purple"];
let str=arr.toString();
document.getElementById("d1").innerHTML=str;

const arr1=str.split(" ");
document.getElementById("d2").innerHTML=arr1;
```