**Name : Yash**

**Q1. Array Methods**

**Ans -**

var a = [1,2,3,4,5]

Undefined

var b = [4,5,6]

Undefined

var c = a.concat(b) //concat

Undefined

c

1. [1, 2, 3, 4, 5, 4, 5, 6]

b.every((val) => val > 3) //every

True

b.every((val) => val > 4)

False

b

1. [4, 5, 6]

var d = c.filter((val) => val%2 == 0); //filter

Undefined

d //contains even elemnts of c where c is concatenation of a and b

1. [2, 4, 4, 6]

c.forEach((val) => console.log(val)) //forEach

VM1179:1 1

VM1179:1 2

VM1179:1 3

VM1179:1 4

VM1179:1 5

VM1179:1 4

VM1179:1 5

VM1179:1 6

Undefined

c.indexOf(5) //indexOf\_example

4

c.lastIndexOf(5) //lastindexOf\_example

6

var e = c.map((val) => val + 2) //map\_example

Undefined

e //mapped c

1. [3, 4, 5, 6, 7, 6, 7, 8]

console.log(e.join()) //join\_example

VM1490:1 3,4,5,6,7,6,7,8

Undefined

e.pop() //pop\_example

8

e.push(689) //push\_example

8

e //e\_after\_pushing\_689

1. [3, 4, 5, 6, 7, 6, 7, 689]

var x = c.reduce((total, val) => total + val) //reduce

Undefined

x

30

var x = c.reduceRight((total, val) => total + val) //reduceRight

Undefined

x

30

c.reverse() //reverse

(8) [6, 5, 4, 5, 4, 3, 2, 1] //c after reverse

c.shift() //shift

6

c.slice(1, 5) //slice

(4) [4, 5, 4, 3]

c

1. [5, 4, 5, 4, 3, 2, 1] //c after slice

c.some((val) => val > 6) //some

false

c.some((val) => val >= 5)

True

c.sort((a,b) => a-b) //sort

(7) [1, 2, 3, 4, 4, 5, 5]

c.splice(1, 3) //splice

(3) [2, 3, 4]

c.toString() //to\_string

"1,4,5,5"

c.unshift(0) //unshift

5

c //c after unshift

1. [0, 1, 4, 5, 5]

// to source is not supported in my browser but here is the code along with the error…..

function Car(brand, type, cost, color) {

this.brand = brand;

this.type = type;

this.color = color;

this.cost = cost;

}

Undefined

acar = new Car('BMW', 'sedan', 3000000, 'Black');

Car {brand: "BMW", type: "sedan", color: "Black", cost: 3000000}

acar.toSource()

VM423:1 Uncaught TypeError: acar.toSource is not a function

at <anonymous>:1:6

**Q2. Output of following code**:

var add = (function() {

var counter = 0;

return function() {return counter += 1;}

})();

add();

add();

add();

**Ans** - The output for this code snippet will be 1 2 3 ... and so on because the function add is returning another function which increases the value of counter by 1.

However since the function which is being returned does not have any counter declared locally (as the counter being declared in the add function is private to the add function) it adds one to counter variable which is hoisted instead of adding 1 to the local value of counter. Hence we get 1, 2, 3 instead of 1, 1, 1.

**Q3. /n vs /r**

**Ans -** /n and /r both move the cursor to the next line, however they are different conceptually. /r is carriage return whereas /n is called line feed. On old devices like the typewriters /r is responsible for

shifting the typewriter back to the starting of the next line when the previous line gets completely filled whereas /n starts a new line regardless of the space left in the previous line.

For example when you write a line until the line has no space left, your pointer automatically moves to the next line. This is when carriage return is functioning. Whereas when you press 'Enter' to move to the next line \n works. Modern day OS still have these conventions, Unix uses \n to mean new line, Macs prior to OS9 used \r, and Windows uses \r\n.

**Q4. regex and array are in the html file**