2array.html

```
<!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>
   <h1>array question</h1>

   <script>
    //code step by step
//1    // cut array length
//     let data=[4,7,1,4,9,2,4,1];
// data.length=5;
// console.log(data)


//2 sum of array elements
//     let data=[4,7,1,4,9,2,4,1,1];
// let sum=data.reduce((x,y)=>x+y)
// console.log(sum)


3//remove duplicate value
//Sets ke under duplicate undifined empty value nahi a sakti
//     let data=[4,7,1,4,9,2,4,1,1];
// let unique=new Set(data);
// console.log([...unique])

//4 comma operator breaket ke bech mai ata isme 2 paramenter hota
//pahla calculation   //dusra value print
// let x=10;
// x=(x++,x);
// console.log(x)

//5 value swaping using destructuring
// let x=10; let y=2;
// [x,y]=[y,x];
// console.log("x: "+x+ " y: "+y)


//technical suneja
//adding element to the array
//level 0 push()exsist array ko bhi update kar deta hai
// let arr=[1,2];
// const append=(arr,ele)=>{
//     arr.push(ele);
//     return arr
// }
// // console.log(append(arr,3))
// newresult=append(arr,5);
// console.log(newresult) //125
// console.log(arr) //125
```

```
55
56
57
58 //level 1 original array ko modify nahi karega
59
60 // let arr=[1,2];
61 // const append=(arr,ele)=>{
62 //     return [...arr,ele]
63 // }
64 // console.log(append(arr,3));
65 // console.log(arr)
66
67 //important defination
68 //set Object
69          //it is a built in javascript data structure that allow us to store unique values of any type
70  //spread operator
71             //spread operator is used to expand the array into individual arguments
72 //includes method
73        //inclues method is used to check if it already exists in the uniqueElement arrat if it does not
74
75
76
77 //3 remove duplicate element in the array
78
79 //level 0
80 //  let arr=[1,1,2,2,3,4,4,5];
81
82 // function uniqueArray(){
83 //     let uniqueElement=[];
84 //     arr.forEach((elm)=>{
85 //         if(!uniqueElement.includes (elm)){
86 //             uniqueElement.push(elm)
87 //         }
88 //     })
89 //     return uniqueElement
90 // }
91
92 // console.log(uniqueArray(arr))
93
94
95
96 //level 1  data structure //it does not modify original array
97 // let arr=[1,1,2,2,3,3,4,5];
98 // function uniqueArray(arr){
99 //     return [...new Set(arr)]
100 // }
101 // console.log(uniqueArray(arr))
102 // console.log(arr)
103
104 //level 2
105 // let arr=[1,1,2,2,3,3,4,5];
106 // function uniqueArray(arr){
107 // return arr.reduce((acc,ele)=>{
108 // return acc.includes(ele)?acc:[...acc,ele]
109 // },[])
110
111 // }
112 // console.log(uniqueArray(arr))
```

```
113
114
115
116
117 //4 concatenating the array
118 // const mergeArray=(arr1,arr2)=>{
119 //     arr1.push(...arr2)
120 //     return arr1;
121 // }
122 // const arr1=[1,2];
123 // const arr2=[3,4];
124 // result=mergeArray(arr1,arr2);
125 // console.log(result)
126
127
128
129 //method 2
130 // const mergeArray=(arr1,arr2)=>{
131 //  return [...arr1,...arr2]
132 // }
133 // const arr1=[1,2];
134 // const arr2=[3,4];
135 // result=mergeArray(arr1,arr2);
136 // console.log(result)
137
138
139 //method 2
140 // const mergeArray=(arr1,arr2)=>{
141 //    return arr1.concat(...arr2)
142 // }
143 // const arr1=[1,2];
144 // const arr2=[3,4];
145 // result=mergeArray(arr1,arr2);
146 // console.log(result)
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
```

```
171
172
173     </script>
174 </body>
175 </html>
```