Week 6 Task

Question 1: Reverse an Array

Write a function that takes an array and returns a new array with the elements in

reverse order.

Input: [1, 2, 3, 4, 5]

Sol:

Using reverse() method:

let a = [1, 2, 3, 4, 5]; a.reverse(); console.log(a);

Using map method:

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

var newarr = a.map((item,idx)=> a[a.length -1 - idx])
console.log(newarr)

Output: [5, 4, 3, 2, 1]

Use Case: This function can be used in a web application where user reviews need to be displayed in reverse chronological order.

Question 2: Flatten an Array

Write a function that takes a nested array and flattens it to a single-level array.

```
Input:[1, [2, 3], [4, [5]]]
Sol:
let a = [1, [2, 3], [4, [5]]];
let b = flattenArray(a);
console.log(b);
function flattenArray(arr) {
  return arr.flat(Infinity);
}
Output: [1, 2, 3, 4, 5]
```

Use Case: Useful for aggregating user-selected items from multiple categories into a single list for checkout.

Question 3: Check for Duplicates

Write a function that checks if an array contains duplicates.

Input: [1, 2, 3, 4, 5, 1]

Output: true

Input: [1, 2, 3, 4, 5]

Output: false

```
Sol:
```

```
let set1=[1, 2, 3, 4, 5, 1]
let set2=[1, 2, 3, 4, 5]
function hasDuplicates(arr) {
 const seenElements = {};
 for (let i = 0; i < arr.length; i++) {
  if (seenElements[arr[i]]) {
   return true;
  seenElements[arr[i]] = true;
 return false;
}
console.log(hasDuplicates(set1));
console.log(hasDuplicates(set2));
Use Case: Can be used to validate user inputs in forms, such as
ensuring usernames are
unique during registration.
```

Question 4: Merge Two Objects

Write a function that merges two objects into one.

```
Input: { a: 1, b: 2 }, { b: 2, c: 4 }
Sol:
```

Using Object.assign() method:

```
var obj1={ a: 1, b: 2 };
var obj2={ b: 2, c: 4 };
var obj3 = Object.assign({},obj1,obj2);
console.log(obj3);
```

Using spread method:

```
var obj1={ a: 1, b: 2 };
var obj2={ b: 2, c: 4 };
var obj3 = mergeobject(obj1,obj2);
function mergeobject(obj1,obj2){
  return {...obj1, ...obj2}
}
console.log(obj3);
```

Output: { a: 1, b: 2, c: 4 }

Use Case: This can be used in a web application to combine user profile settings from different sources.

Question 5: Find the Maximum Number in an Array Write a

function that finds the maximum number in an array.

```
Input: [1, 3, 2, 8, 5] Sol:
```

}

Using Math.max() method:

```
const a = [1, 3, 2, 8, 5];
const b = findmax(a);
function findmax(high){
  return Math.max(...high);
}
console.log(b)
Using filter method
const arr = [1, 3, 2, 8, 5];
function largestElementWithFilter(arr) {
  if (arr.length === 0) {
     return;
  }
  let max = arr[0];
  arr.filter(num => {
     if (num > max) {
       max = num;
```

```
});
  return max;
}
console.log( largestElementWithFilter(arr));
Output: 8
Use Case: This function can help in analytics dashboards to find
the highest sales figure or user
activity.
Question 6: Group Array of Objects by Property
Write a function that groups an array of objects by a specific
property.
Input: [ { id: 1, category: 'fruit' }, { id: 2, category: 'vegetable' },
{ id: 3, category: 'fruit' } ]
Sol:
var product=[
  { id: 1, category: 'fruit' },
  { id: 2, category: 'vegetable' },
  { id: 3, category: 'fruit' }
  ];
  function groupBy(array, property) {
 return array.reduce((acc, obj) => {
```

```
const key = obj[property];
  if (!acc[key]) {
   acc[key] = [];
  }
  acc[key].push(obj);
  return acc;
 }, {});
}
const groupedResult = groupBy(product, 'category');
console.log(groupedResult);
Output: {
fruit: [ { id: 1, category: 'fruit' }, { id: 3, category: 'fruit' } ],
vegetable: [ { id: 2, category: 'vegetable' } ]
Use Case: Useful for organizing products by category in an e-
commerce application.
```

Question 7: Find the Intersection of Two Arrays

Write a function that returns the intersection of two arrays.

```
Input: [1, 2, 3], [2, 3, 4]
Sol:
const num1 = [1, 2, 3];
```

```
const num2 = [2, 3, 4];
function intersection(num1, num2) {
 return num1.filter(element => num2.includes(element));
}
const result = intersection(num1,num2);
console.log(result);
```

Output: [2, 3]

Use Case: This can be used in social media applications to find mutual friends between users.

Question 8: Calculate the Sum of Array Elements

Write a function that calculates the sum of all numbers in an array.

Input: [1, 2, 3, 4, 5]

Sol:

Using forLoop method:

```
var a= [1, 2, 3, 4, 5]
var sum = 0
for(count=0; count < a.length; count++){</pre>
  sum += a[count];
}
console.log(sum)
```

Using reduce() method:

```
const a = [1, 2, 3, 4, 5];
const sum = sumArray(a);
function sumArray(arr) {
 return arr.reduce((acc, num) => acc + num, 0);
}
console.log(sum);
Output: 15
Use Case: Useful in financial applications to calculate the total
expenses or revenue.
Question 9: Remove Falsy Values from an Array Write a
function that removes all falsy values from an array.
Input: [0, 1, false, 2, ", 3]
Sol:
const num = [0, 1, false, 2, ", 3];
function removeFalsyValues(arr) {
 return arr.filter(Boolean);
const result = removeFalsyValues(num);
console.log(result);
Output: [1, 2, 3]
```

Use Case: This function can be used to clean up user inputs or configuration arrays.

Question 10: Calculate Average of an Array

Write a function that calculates the average of all numbers in an array.

```
Input: [1, 2, 3, 4, 5] Sol:
```

Using forLoop method:

```
var a= [1, 2, 3, 4, 5]
var sum = 0
for(count=0; count < a.length; count++){
    sum += a[count]/a.length;
}
console.log(sum)</pre>
```

Using forEach method:

```
let a = [1, 2, 3, 4, 5];
function calculateAverage(array) {
    let sum = 0;
    array.forEach(function(element) {
        sum += element;
    });
```

```
return sum / a.length;
}
console.log(calculateAverage(a));
Output: 3
```

Use Case: This function is useful in educational applications where you need to compute the average score of students from an array of their grades.