JavaScript Assignment - 2

1. Given an array of objects representing transactions, write a function to group transactions by type.

```
let transactions = [
   { type: "deposit", amount: 100 },
   { type: "withdrawal", amount: 50 },
   { type: "deposit", amount: 200 },
   { type: "withdrawal", amount: 30 }
];

function groupByType(transactions) {
   let grouped = {};

   // your code here...

   return grouped;
}

console.log(groupByType(transactions));
// Output: { deposit: [{ type: "deposit", amount: 100 }, { type: "deposit",
amount: 200 }], withdrawal: [{ type: "withdrawal", amount: 50 }, { type:
"withdrawal", amount: 30 }] }
```

2. You have an array of objects representing employees. Write a function to filter out employees who are older than a given age.

```
let employees = [
   { name: "Alice", age: 25 },
   { name: "Bob", age: 35 },
   { name: "Charlie", age: 30 }
];
let ageLimit = 30;

function filterEmployees(employees, ageLimit) {
   // your code here
```

```
}

console.log(filterEmployees(employees, ageLimit));
// Output: [{ name: "Alice", age: 25 }, { name: "Charlie", age: 30 }]
```

3. Given a 2D array representing a list of student names in different classes, write a function to check if there are any duplicate names across all classes.
   (Specifically use JavaScript **Set** Objects)

```
let classes = [
   ["Alice", "Bob", "Charlie"],
   ["David", "Alice", "Eve"],
   ["Frank", "George", "Bob"]
];

function checkDuplicates(classes) {
   let nameSet = new Set();
   // your code here
}

console.log(checkDuplicates(classes));
// Output: true
```

4. Write a function to get the current date and time in the format `YYYY-MM-DD HH:MM:SS`.
   (Hint: Use Date object in JavaScript)

5. Write a function that takes a birthdate as input and calculates the person's age.

```
function calculateAge(birthdate) {

}

console.log(calculateAge("1990-06-15"));
```

6. Write a function that takes a date and a number of days as input and returns the new date after adding the specified number of days.

```
function addDays(date, days) {
   // YOUR CODE HERE
}

console.log(addDays("2022-06-15", 10));
```

7. Write a function that calculates the number of days between two given dates.
(Hint: use Math function in JavaScrpt & Date object to write the function)

```
function dateDifference(date1, date2) {
   // Your code here
}

console.log(dateDifference("2022-06-15", "2023-06-15")); // Example dates
```

8. Suppose you are developing a web application that allows users to set and retrieve their preferences. Use a `Map` to store user preferences, and write functions to set a preference and get a preference by key.

```
let userPreferences = new Map();

function setPreference(key, value) {
   // USE SET METHOD IN MAP OBJECT
}

function getPreference(key) {
   // use get method in Map oBJECT
}

setPreference("theme", "dark");
setPreference("fontSize", "16px");
console.log(getPreference("theme")); // Output: dark
console.log(getPreference("fontSize")); // Output: 16px
```

9. Write a function that takes an array of strings and returns a `Map` where the keys are the unique strings and the values are the number of times each string appears in the array. (Use Map Object)

```
function countOccurrences(arr) {
   // your code here
}

// Example usage:
let strings = ["apple", "banana", "apple", "orange", "banana", "apple"];
console.log(countOccurrences(strings));
// Output: Map { 'apple' => 3, 'banana' => 2, 'orange' => 1 };
```

10. You are given an array of employee objects with `id` and `name` properties. Write a function to create a `Map` that maps employee IDs to their names.

```
let employees = [
   { id: 1, name: "Alice" },
   { id: 2, name: "Bob" },
   { id: 3, name: "Charlie" }
];

function mapEmployeeIdsToNames(employees) {
   // use Map Object here
}

let employeeMap = mapEmployeeIdsToNames(employees);
console.log(employeeMap);
// Output: Map { 1 => 'Alice', 2 => 'Bob', 3 => 'Charlie' }
```

11. Write a function to manage inventory for a store using a `Map`. The `Map` should store item names as keys and their quantities as values. Provide functions to add new items, update item quantities, and check the quantity of an item.

```
let inventory = new Map();

function addItem(itemName, quantity) {
   // use Map Object's set method to add the item to the inventory
}

function updateItemQuantity(itemName, quantity) {
   // use Map Object's has, set, get methods to update the quantity

   // check if the item exists in the inventory
   // if exists, update the quantity
   // if not, console log the message "Item not found"
}

function checkItemQuantity(itemName) {
   // check item quantity in the inventory
}

addItem("apple", 100);
addItem("banana", 150);
updateItemQuantity("apple", 50);
console.log(checkItemQuantity("apple")); // Output: 150
console.log(checkItemQuantity("banana")); // Output: 150
```

12. Write a function that takes an array of numbers and returns a new array with all duplicate items removed. (Use Set Object)

```
function removeDuplicates(arr) {
   // your code here
}

let numbers = [1, 2, 2, 3, 4, 4, 5];
console.log(removeDuplicates(numbers));
// Output: [1, 2, 3, 4, 5]
```

13. Write a function that takes two arrays and returns a new array containing only the elements that are present in both arrays.
(use Set object)

```
function intersection(arr1, arr2) {
   let set1 = new Set(arr1);
   let set2 = new Set(arr2);

   // your code here
   // understand the above 2 variable declarations
}

let array1 = [1, 2, 3, 4];
let array2 = [3, 4, 5, 6];
console.log(intersection(array1, array2));
// Output: [3, 4]
```

14. Write a function that takes multiple arrays and returns a new array containing all unique elements from all arrays.

```
function union(...arrays) {
   let resultSet = new Set();

   // your code here
}

let array1 = [1, 2, 3];
let array2 = [3, 4, 5];
let array3 = [5, 6, 7];
console.log(union(array1, array2, array3));
// Output: [1, 2, 3, 4, 5, 6, 7]
```

15. Write a function that takes an array of items and a set, and removes all items in the array from the set.

```
function removeItemsFromSet(items, set) {
   // your code here
}



let mySet = new Set([1, 2, 3, 4, 5]);
let itemsToRemove = [2, 3];
removeItemsFromSet(itemsToRemove, mySet);
console.log([...mySet]); // Output: [1, 4, 5]
```

16. Write a function that takes an array of student names representing attendance on a particular day and a `Set` representing the overall attendance. Update the `Set` with the new attendance and return the updated `Set`.

```
function updateAttendance(newAttendance, overallAttendance) {
   // your code here
}

let overallAttendance = new Set(["Alice", "Bob"]);
let todayAttendance = ["Charlie", "Alice"];
overallAttendance = updateAttendance(todayAttendance, overallAttendance);
console.log([...overallAttendance]);
// Output: ["Alice", "Bob", "Charlie"]
```