Assignment Solution-3
HOF and Functional programming

1. You are building an e-commerce website. Write a function that calculates the total price of a customer's order. You're given an array of items, each with a price property. Use the forEach method to iterate through the array and sum up the prices to get the total order amount.

```
const ordersList = [
    { name: "Laptop", price: 120000 },
    { name: "Mobile", price: 70000 },
    { name: "Mobile Charger", price: 1500 },
    { name: "Laptop Charger", price: 10500 },
];

// OUTPUT: The total price is Rs.202000
```

Solution:

Q1.js

```
const ordersList = [
    { name: "Laptop", price: 120000 },
    { name: "Mobile", price: 70000 },
    { name: "Mobile Charger", price: 1500 },
    { name: "Laptop Charger", price: 10500 }
];

let totalOrderAmount = 0;

ordersList.forEach(item => {
    totalOrderAmount += item.price;
});

console.log("The total price is Rs." + totalOrderAmount);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q1.js
The total price is Rs.202000
```

2. In this challenge, your task is to create a function that generates a random number and prints it to the console every 2 seconds. The program should keep printing new random numbers indefinitely, with a 2-second delay between each number.

Solution:

Q2.js

```
setInterval(() => {
    const randomNumber = Math.random();
    console.log(randomNumber);
}, 2000);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q2.js
0.42120243565123294
0.3984172083917299
0.949765323222105
0.03122646414570407
```

3. You are given an array of expense objects representing monthly expenses. Each object has properties, amount and category. Use the map method to create a new array that includes the calculated tax for each expense. Assume a tax rate of 10%.

```
let expenses = [
  { amount: 100, category: "Utilities" },
  { amount: 200, category: "Groceries" },
  { amount: 50, category: "Entertainment" },
];
```

Solution:

Q3.js

```
let expenses = [
    { amount: 100, category: "Utilities" },
    { amount: 200, category: "Groceries" },
    { amount: 50, category: "Entertainment" }
];

let taxes = expenses.map(expense => {
    return {
        ...expense,
        tax: expense.amount * 0.1
    };
});

console.log(taxes);
```

```
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q3.js
[
  { amount: 100, category: 'Utilities', tax: 10 },
  { amount: 200, category: 'Groceries', tax: 20 },
  { amount: 50, category: 'Entertainment', tax: 5 }
]
```

4. Using the same array of expense objects, use the filter method to create a new array that includes only the expenses related to the category "Groceries."

Solution:

Q4.js

```
let expenses = [
    { amount: 100, category: "Utilities" },
    { amount: 200, category: "Groceries" },
    { amount: 50, category: "Entertainment" }
];
  const groceriesExpenses = expenses.filter(expense => expense.category === "Groceries");

  console.log(groceriesExpenses);
```

```
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q4.js
[ { amount: 200, category: 'Groceries' } ]
```

5. Using the same array of expense objects, use the reduce method to calculate the total amount of all expenses.

Solution:

Q5.js

```
let expenses = [
    { amount: 100, category: "Utilities" },
    { amount: 200, category: "Groceries" },
    { amount: 50, category: "Entertainment" }
];
  const totalAmount = expenses.reduce((accumulator, expense) => accumulator + expense.amount, 0);

  console.log(`The total amount of all expenses is Rs ${totalAmount}.`)
```

```
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q5.js
The total amount of all expenses is Rs 350.
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3>
```

6. You have a list of expenses, each with an amount and a category. Now, create a function named categorizeExpense that, based on the expense amount, returns either "High Expense" if it's more than 100, or "Low Expense" otherwise. Afterward, use this function along with the map method to generate a new array called categorizedExpenses, where each element represents the category for the corresponding expense in the original list. Finally, print out the categorizedExpenses array.

```
let expenses = [
  { amount: 100, category: "Utilities" },
  { amount: 200, category: "Groceries" },
  { amount: 50, category: "Entertainment" },
];
```

Solution:

Q6.js > categorizeExpense

```
let expenses = [
    { amount: 100, category: "Utilities" },
    { amount: 200, category: "Groceries" },
    { amount: 50, category: "Entertainment" }
];

function categorizeExpense(expense) {
    return expense.amount > 100 ? 'High Expense' : 'Low Expense';
}

let categorizedExpenses = expenses.map(expense => categorizeExpense(expense));

console.log('Categorized Expenses:', categorizedExpenses);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q6.js
Categorized Expenses: [ 'Low Expense', 'High Expense', 'Low Expense' ]
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3>
```

7. Consider an array of numbers named originalNumbers with the values [2, 5, 8, 10, 3]. Your task is to use the forEach method to iterate through each element in the array. During the iteration, double the value of each number. After completing the iteration, display the modified array.

```
let originalNumbers = [2, 5, 8, 10, 3];
```

Solution:

Q7.js > ...

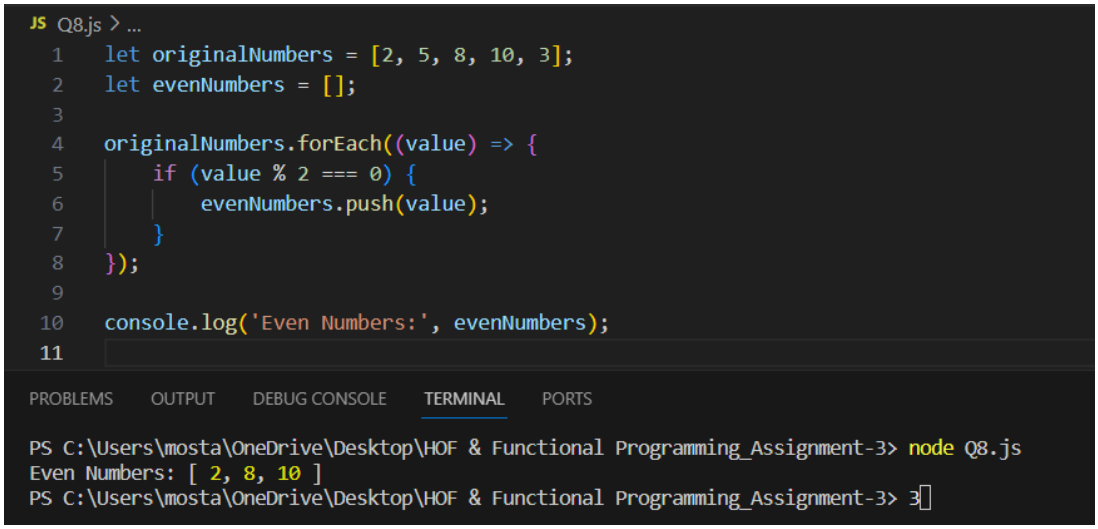
```
let originalNumbers = [2, 5, 8, 10, 3];
originalNumbers.forEach((value, index) => {
    originalNumbers[index] = value * 2;
});
console.log(originalNumbers);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q7.js
[ 4, 10, 16, 20, 6 ]
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3>
```

8. Using the same array of numbers, use the forEach method to collect and store only the even numbers in a new array.

Solution:



```
JS Q8.js > ...
1   let originalNumbers = [2, 5, 8, 10, 3];
2   let evenNumbers = [];
3
4   originalNumbers.forEach((value) => {
5       if (value % 2 === 0) {
6           evenNumbers.push(value);
7       }
8   });
9
10  console.log('Even Numbers:', evenNumbers);
11
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> node Q8.js
Even Numbers: [ 2, 8, 10 ]
PS C:\Users\mosta\OneDrive\Desktop\HOF & Functional Programming_Assignment-3> 3
```