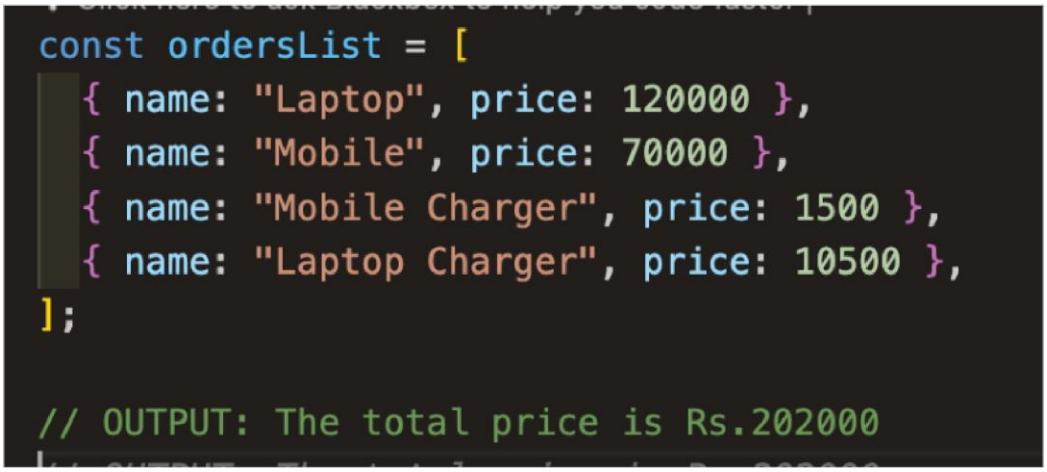
1 **You are building an e-commerce website. Write a function that calculates the total price of a customer's order. You're given an array of items, each with a price property. Use the forEach method to iterate through the array and sum up the prices to get the total order amount**



```
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:

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

let sum = 0;
ordersList.forEach((data) => (sum += data.price));

console.log('The total price is Rs.${sum}');
```

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:

```
function randomNumberGenerator() {
  const randomNumber = Math.floor(Math.random() * 100);
  console.log(randomNumber)


function printWithInterval() {
  randomNumberGenerator();
  setinterval(randomNumberGenerator, 2000);


printWithInterval();
```

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%.

Solution:

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

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

console.log("Expenses with Tax:", expensesWithTax);

output:
Expenses with Tax: [
   { 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:

JavaScript

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

```
let graceriesExpenses = expenses.filter(
  (expense) => expense.category === "Groceries"
);

console.log("Groceries Expenses:", groceriesExpenses);
```

Output:
Groceries Expenses: [{ amount: 200, category: 'Groceries' }

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

Solutior JavaScript .,.

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

let totalExpenses = expenses.reduce((acc, expense) => ace + expense.amount, 0);

console.log("Total Expenses:", totalExpenses);

//Output:
Total Expenses: 350
```

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.

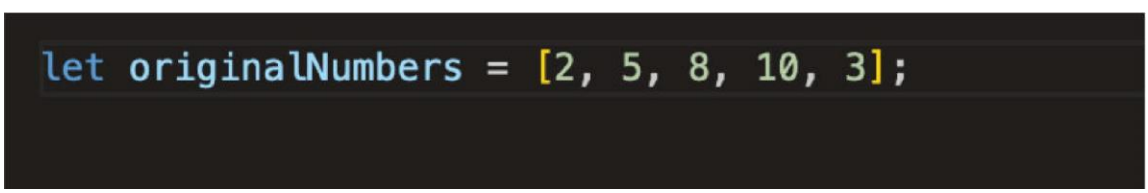
```
let expenses = [
  {amount: 100, category: "Utilities" },
  {amount: 200, category: "Groceries" },
  {amount: 50, category: "Entertainment" },
];
function categorizeExpense (expense) {
  if (expense.amount > 100) {
    return "High Expense";
    else {
    return "Low Expense";
  }
}

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

console.log ("Categorized Expenses:", categorizedExpenses);

//Output :
Categorized Expenses: [ 'Low Expense', 'High Expense', 'Low Expense' ]
```

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.



Solution

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

originalNumbers.forEach ((number, index, array) => {
  array[indexJ = number * 2;
});

console.log ("Doubled Numbers:", originalNumbers);

Output: Doubled Numbers: [4, 10, 16, 20, 6
```

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

Solution:

```
let evenNumbers = [];

originalNumbers.forEach(number => {
  if (number % 2 === 0) {
    evenNumbers.push(number);
  }
});

console.log("Even Numbers:", evenNumbers);
```

```
Output: Even Numbers: [ 2, 8, 10 ]
```