```
// //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



function calculateTotalPrice(items) {
    let total = 0;
    items.forEach(item => {
        total += item.price;
    });

    return total;
}

// Example usage:
const orderItems = [
    { name: 'Item 1', price: 29.99 },
    { name: 'Item 2', price: 15.49 },
    { name: 'Item 3', price: 42.00 }
];

 const totalPrice = calculateTotalPrice(orderItems);
 console.log('Total Price:', totalPrice);


// 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.
function printRandomNumbers() {

    setInterval(() => {

        const randomNumber = Math.floor(Math.random() * 101);

        console.log(randomNumber);
    }, 2000);
}


printRandomNumbers();


// 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%.
// Array of expense objects
const expenses = [
    { amount: 100, category: 'Groceries' },
    { amount: 50, category: 'Utilities' },
    { amount: 200, category: 'Rent' }
];


const taxRate = 0.10;


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


console.log(expensesWithTax);

// 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."
// Array of expense objects
const expenses = [
    { amount: 100, category: 'Groceries' },
    { amount: 50, category: 'Utilities' },
    { amount: 200, category: 'Rent' },
    { amount: 75, category: 'Groceries' }
];


const groceriesExpenses = expenses.filter(expense => expense.category === 'Groceries');


console.log(groceriesExpenses);


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

const expenses = [
    { amount: 100, category: 'Groceries' },
    { amount: 50, category: 'Utilities' },
    { amount: 200, category: 'Rent' },
    { amount: 75, category: 'Groceries' }
];


const totalAmount = expenses.reduce((accumulator, expense) => {
    return accumulator + expense.amount;
}, 0);


console.log('Total Amount:', totalAmount);


// 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.


const expenses = [
    { amount: 100, category: 'Groceries' },
    { amount: 50, category: 'Utilities' },
    { amount: 200, category: 'Rent' },
    { amount: 75, category: 'Groceries' }
];


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


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

// Output the categorizedExpenses array
console.log(categorizedExpenses);


// 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];


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


console.log(originalNumbers);
```

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

const originalNumbers = [2, 5, 8, 10, 3];


const evenNumbers = [];


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


console.log(evenNumbers);
```