Array – That store a type of data type

📌 JavaScript Concepts for Expense Tracker

1️⃣ Arrays (To store multiple expenses)

- **Definition**: An array is a list that holds multiple values.

- **Example**: Storing expenses in an array.

```javascript
let expenses = [100, 200, 300];
console.log(expenses[0]); // Output: 100 (first expense)
```

2️⃣ Objects (To store detailed expense information)

- **Definition**: An object holds multiple properties inside `{}`.

- **Example**: Store amount, category, and date inside an object.

```javascript
let expense = { amount: 100, category: "Food", date: "2025-02-13" };
console.log(expense.category); // Output: "Food"
```

Foreach – array me looop

3 forEach (To loop over expenses)
- **Definition**: Runs a function on every array element.
- **Example**: Show all expenses in the console.

```javascript
expenses.forEach(function(expense) {
  console.log(expense);
});
```

Output:

```
100
200
300
```

Map – array me har index me jake koi operation karna works like loop

4 map (To modify all expenses)
- **Definition**: Creates a new array by changing each element.
- **Example**: Increase all expenses by 10%.

```javascript
let updatedExpenses = expenses.map(expense => expense * 1.1);
console.log(updatedExpenses);
```

Output:

```csharp
[110, 220, 330]
```

Filter – it filter works like if

5 filter (To keep only certain expenses)

- **Definition**: Returns a new array with specific elements.
- **Example**: Get only expenses greater than 150.

```javascript
let highExpenses = expenses.filter(expense => expense > 150);
console.log(highExpenses);
```

Output:

```csharp
[200, 300]
```

Reduce-find sum of the array

6 reduce (To calculate total expenses)

- **Definition**: Reduces an array to a single value.
- **Example**: Calculate total expenses.

```javascript
let total = expenses.reduce((sum, expense) => sum + expense, 0);
console.log(total);
```

Output:

```
600
```