JavaScript Array & String Methods – Complete Professional Task Sheet

SECTION 1 – Array Higher Order Methods Tasks

Task 1 – forEach() Practice

let subjects = ["Maths", "Science", "English", "History"]

Requirements:

- Use forEach()
- Print Subject 1: Maths ... Subject 4: History
- Store result in a variable
- Question: Why does forEach() return undefined?

```
let subjects = ["Maths", "Science", "English", "History"];


let result = [];


subjects.forEach(function(subject, index) {
  result.push(`Subject ${index + 1}: ${subject}`);
});


console.log(result);
// forEach() returns undefined because it is designed for side effects  rather than returning a new array. It does not return anything, so the result of calling forEach() is always undefined.
```

Task 2 – map() Practice

let prices = [100, 200, 300, 400]

Requirements:

- Use map()
- Add 10% GST
- Return new array
- Expected Output: [110, 220, 330, 440]

```
let prices = [100, 200, 300, 400];

let newPrices = prices.map(function(price) {
    return price + (price * 0.10);  // Add 10% GST
});

console.log(newPrices);
```

output:

```
[110, 220, 330, 440]
```

Task 3 – filter() Practice

```
let students = [
{ name: "A", marks: 45 },
{ name: "B", marks: 75 },
{ name: "C", marks: 35 },
{ name: "D", marks: 85 }
]
```

Requirements:

- Use filter()
- Get students scoring above 50
- Print filtered array

```
let students = [
  { name: "A", marks: 45 },
  { name: "B", marks: 75 },
  { name: "C", marks: 35 },
  { name: "D", marks: 85 }
];

let passedStudents = students.filter(function(student) {
  return student.marks > 50;
});

console.log(passedStudents);
```

output:

```
[
  { name: "B", marks: 75 },
  { name: "D", marks: 85 }
]
```

Task 4 – find() Practice

Use same students array

Requirements:

- Use find()
- Find first student scoring above 50
- Print that object
- Question: Difference between filter() and find()?

```
let students = [
  { name: "A", marks: 45 },
  { name: "B", marks: 75 },
  { name: "C", marks: 35 },
  { name: "D", marks: 85 }
];

let firstPassed = students.find(function(student) {
  return student.marks > 50;
});

console.log(firstPassed);
```

output:

```
{ name: "B", marks: 75 }
```

Task 5 – reduce() Practice

```
let cart = [
{ item: "Shirt", price: 1000 },
{ item: "Shoes", price: 2000 },
{ item: "Watch", price: 3000 }
]
```

Requirements:

- Use reduce()
- Calculate total price
- Bonus: Add 5% tax inside reduce()

```
let cart = [
  { item: "Shirt", price: 1000 },
  { item: "Shoes", price: 2000 },
  { item: "Watch", price: 3000 }
];

// Calculate total price
let total = cart.reduce(function(accumulator, product) {
  return accumulator + product.price;
}, 0);

console.log("Total Price:", total);
```

output:

```
Total Price: 6000
```

```
// Bonus: Add 5% tax inside reduce()
let totalWithTax = cart.reduce(function(accumulator, product) {
  return accumulator + product.price * 1.05; // adding 5% tax
}, 0);

console.log("Total with 5% Tax:", totalWithTax);
```

output:

```
Total with 5% Tax: 6300
```

Task 6 – some() Practice

```
let numbers = [1, 3, 5, 7, 8]
```

Requirement:

- Check if at least one number is even
- Output: true

```
let numbers = [1, 3, 5, 7, 8];

let hasEven = numbers.some(function(num) {
  return num % 2 === 0;
});

console.log(hasEven);
```

output:

```
true
```

Task 7 – every() Practice

```
let ages = [22, 25, 19, 30]
```

Requirement:

- Check if all ages are above 18

```
let ages = [22, 25, 19, 30];

let allAdults = ages.every(function(age) {
  return age > 18;
});

console.log(allAdults);
```

output:

```
true
```

Task 8 – sort() Practice

let salaries = [50000, 10000, 70000, 30000]

Requirements:

- Sort ascending
- Sort descending
- Explain why normal sort() fails for numbers

```
let salaries = [50000, 10000, 70000, 30000];

// Ascending order
let ascending = [...salaries].sort((a, b) => a - b);
console.log("Ascending:", ascending);

// Descending order
let descending = [...salaries].sort((a, b) => b - a);
console.log("Descending:", descending);
```

output:

```
Ascending:  [10000, 30000, 50000, 70000]
Descending: [70000, 50000, 30000, 10000]
```

Task 9 – Array Conversion Methods

let arr = [10, 20, 30, 40]

Requirements:

- Convert using toString()
- Convert using join("-")
- Expected Output: 10-20-30-40

```
let arr = [10, 20, 30, 40];
// Convert using toString()
let str1 = arr.toString();
console.log(str1);
```

output:

```
10,20,30,40
```

```
// Convert using join("-")
let str2 = arr.join("-");
console.log(str2);
```

output:

```
10-20-30-40
```

SECTION 2 – String Methods Tasks

Task 10 – charAt() & charCodeAt()

let word = "Developer"

Requirements:

- Get character at index 4
- Get ASCII value of character at index 4

```
let word = "Developer";

// Character at index 4
let char = word.charAt(4);
console.log("Character at index 4:", char);

// ASCII value of character at index 4
let ascii = word.charCodeAt(4);
console.log("ASCII value:", ascii);
```

output:

```
Character at index 4: l
ASCII value: 108
```

Task 11 – slice()

let company = "StacklyCompany"

Extract: Company

```
let company = "StacklyCompany";
```

```
// Extract "Company"
let result = company.slice(7);

console.log(result);
```

output:

```
Company
```

Task 12 – Case Conversion

let userInput = "javaScript"

- Convert to uppercase
- Convert to lowercase

```
let userInput = "javaScript";

// Convert to Uppercase
let upper = userInput.toUpperCase();
console.log("Uppercase:", upper);

// Convert to Lowercase
let lower = userInput.toLowerCase();
console.log("Lowercase:", lower);
```

output:

```
Uppercase: JAVASCRIPT
Lowercase: javascript
```

Task 13 – trim()

let email = " naveen@gmail.com "

Remove unwanted spaces

```
let email = " naveen@gmail.com ";

// Remove unwanted spaces
let cleanEmail = email.trim();
```

```
console.log(cleanEmail);
```

output:

```
naveen@gmail.com
```

Task 14 – includes()

let message = "Welcome to JavaScript Training"

Check if "JavaScript" exists

```
let message = "Welcome to JavaScript Training";

// Check if "JavaScript" exists
let check = message.includes("JavaScript");

console.log(check);
```

output:

```
true
```

Task 15 – split()

let movie = "spider-man-no-way-home"

Convert into array

```
let movie = "spider-man-no-way-home";

// Convert into array
let result = movie.split("-");

console.log(result);
```

output:

```
["spider", "man", "no", "way", "home"]
```

Task 16 – indexOf() & lastIndexOf()

let text = "programming"

- Find first index of 'm'

- Find last index of 'm'

```
let text = "programming";

// First index of 'm'
let firstIndex = text.indexOf("m");
console.log("First index of 'm':", firstIndex);

// Last index of 'm'
let lastIndex = text.lastIndexOf("m");
console.log("Last index of 'm':", lastIndex);
```

output:

```
First index of 'm': 6
Last index of 'm': 9
```

Task 17 – replace()

```
let tech = "I love python"
```

Replace python with javascript

```
let tech = "I love python";

// Replace "python" with "javascript"
let result = tech.replace("python", "javascript");

console.log(result);
```

output:

```
I love javascript
```

Task 18 – startsWith() & endsWith()

```
let filename = "report.pdf"
```

- Check if starts with "report"
- Check if ends with ".pdf"

```
let filename = "report.pdf";

// Check if starts with "report"
let checkStart = filename.startsWith("report");
console.log("Starts with 'report':", checkStart);

// Check if ends with ".pdf"
let checkEnd = filename.endsWith(".pdf");
console.log("Ends with '.pdf':", checkEnd);
```

output:

```
Starts with 'report': true
Ends with '.pdf': true
```

Task 19 – repeat()

let star = "*"

Print 10 stars using repeat()

```
let star = "*";

// Print 10 stars
let result = star.repeat(10);

console.log(result);
```

```
output:**********
```

FINAL TEAM CHALLENGE – Employee Report System

```
let employees = [
{ name: "Naveen", salary: 50000 },
{ name: "Arun", salary: 30000 },
{ name: "Kiran", salary: 70000 }
]
```

Requirements:

• Convert names to uppercase using map()

- Filter salary > 40000
- Find first salary > 60000
- Calculate total salary using reduce()
- Sort salaries descending

```
let employees = [
  { name: "Naveen", salary: 50000 },
  { name: "Arun", salary: 30000 },
  { name: "Kiran", salary: 70000 }
];
// Convert names to uppercase using map()
let upperNames = employees.map(emp => ({
  ...emp,
  name: emp.name.toUpperCase()
}));

console.log("Uppercase Names:", upperNames);
output:
[
  { name: "NAVEEN", salary: 50000 },
  { name: "ARUN", salary: 30000 },
  { name: "KIRAN", salary: 70000 }
]
// Filter salary > 40000
let highSalary = employees.filter(emp => emp.salary > 40000);

console.log("Salary > 40000:", highSalary);
output:
[
  { name: "Naveen", salary: 50000 },
  { name: "Kiran", salary: 70000 }
]
// Find first salary > 60000
let firstHigh = employees.find(emp => emp.salary > 60000);

console.log("First Salary > 60000:", firstHigh);
output:
{ name: "Kiran", salary: 70000 }
let totalSalary = employees.reduce((total, emp) => {
  return total + emp.salary;
}, 0);
// Calculate total salary using reduce()
console.log("Total Salary:", totalSalary);
output:
Total Salary: 150000
let sortedDesc = [...employees].sort((a, b) => b.salary - a.salary);
// Sort salaries descending
console.log("Sorted Descending:", sortedDesc);
output:
[
  { name: "Kiran", salary: 70000 },
  { name: "Naveen", salary: 50000 },
  { name: "Arun", salary: 30000 }
]




</script>

<body>
</html>
```