Practical 1: Spread and Rest Operators in ES6

Aim

To write ES6 code demonstrating the use of rest and spread operators for:

a) Accepting multiple numbers and returning their sum using rest operator

b) Merging two arrays using the spread operator

c) Copying and updating an object using the spread operator

d) Passing array elements as function arguments using spread

(a) Sum of Numbers using Rest Operator

The rest parameter (...) allows a function to accept an indefinite number of arguments as an array. Here we use it with reduce() to sum all arguments.

```
function add(...numbers) {
  return numbers.reduce((sum, number) => sum + number, 0);
}

console.log(add(1, 2, 3));                    // 6
console.log(add(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)); // 55

// Mixing regular + rest parameters
function addMixed(a, b, c, ...rest) {
  return a + b + c + rest.reduce((sum, n) => sum + n, 0);
}

console.log(addMixed(1, 2, 3, 4, 5));         // 15
```

Output:

```
6
55
15
```

(b) Merging Two Arrays using Spread Operator

The spread operator (...) expands an array into individual elements. We can use it to merge two arrays into a new one.

```
const csStudents = ["Akash", "Ashish", "Abhi"];
const itStudents = ["Amit", "Raj", "Sanjay"];

const allStudents = [...csStudents, ...itStudents];
console.log(allStudents);
```

Output:

```
["Akash", "Ashish", "Abhi", "Amit", "Raj", "Sanjay"]
```

(c) Copy and Update an Object using Spread Operator

The spread operator can copy all properties of an object into a new one. Properties listed after the spread override the original values.

Practical 1: Spread and Rest Operators in ES6

```
const user = {
  name: "Akash",
  age: 20,
  city: "Delhi",
  country: "India"
};

const updatedUser = {
  ...user,
  city: "Mumbai"    // overrides "Delhi"
};

console.log(updatedUser);
```

Output:

```
{ name: "Akash", age: 20, city: "Mumbai", country: "India" }
```

(d) Passing Array Elements as Function Arguments using Spread

The spread operator can expand an array into individual arguments when calling a function.

```
const sum = (...numbers) => {
  return numbers.reduce((sum, number) => sum + number, 0);
};

const numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
console.log(sum(...numbers));
// ...numbers spreads the array into individual arguments
```

Output:

```
55
```

Conclusion

The rest operator collects multiple arguments into an array, while the spread operator expands arrays/objects into individual elements. Together, they provide a clean and flexible way to handle variable-length data in ES6.