Methods and Arrays in Java

Method Basics

Method Structure

```
public static returnType methodName(parameters) {
    // Method body
    return value; // if returnType is not void
}
```

Simple Method Examples

```
// Method with no parameters, no return value
public static void sayHello() {
    System.out.println("Hello!");
}

// Method with parameters, no return value
public static void greetPerson(String name) {
    System.out.println("Hello, " + name + "!");
}

// Method with parameters and return value
public static int addNumbers(int a, int b) {
    return a + b;
}

// Method that returns a value
public static double calculateArea(double length, double width) {
    return length * width;
}
```

Calling Methods

```
public static void main(String[] args) {
    sayHello();                    // Output: Hello!
    greetPerson("Alice");          // Output: Hello, Alice!
    int sum = addNumbers(5, 3);    // sum = 8
    double area = calculateArea(10.5, 4.2); // area = 44.1
}
```

Array Basics

Creating Arrays

```
// Declare and create array
int[] numbers = new int[5];        // Array of 5 integers
String[] names = new String[3];    // Array of 3 strings
```

```
// Declare and initialize array
int[] scores = {85, 92, 78, 96, 88};
String[] fruits = {"apple", "banana", "orange"};
```

Accessing Array Elements

```
int[] numbers = {10, 20, 30, 40, 50};

// Access elements (index starts at 0)
int first = numbers[0];      // 10
int third = numbers[2];      // 30
int last = numbers[4];       // 50

// Modify elements
numbers[1] = 25;            // Array becomes {10, 25, 30, 40, 50}

// Array length
int size = numbers.length;   // 5
```

Array Loops

```
int[] scores = {85, 92, 78, 96, 88};

// Traditional for loop
for (int i = 0; i < scores.length; i++) {
    System.out.println("Score " + i + ": " + scores[i]);
}

// Enhanced for loop (for-each)
for (int score : scores) {
    System.out.println("Score: " + score);
}
```

Practical Examples

Example 1: Array Processing Method

```
public static int findMaximum(int[] numbers) {
    int max = numbers[0];
    for (int i = 1; i < numbers.length; i++) {
        if (numbers[i] > max) {
            max = numbers[i];
        }
    }
    return max;
}

public static double calculateAverage(int[] numbers) {
    int sum = 0;
    for (int number : numbers) {
        sum += number;
    }
    return (double) sum / numbers.length;
```

```
}
```

Example 2: String Array Methods

```
public static void printNames(String[] names) {
    for (String name : names) {
        System.out.println("Hello, " + name);
    }
}

public static int countLongNames(String[] names, int minLength) {
    int count = 0;
    for (String name : names) {
        if (name.length() >= minLength) {
            count++;
        }
    }
    return count;
}
```

Common Patterns

Pattern 1: Sum Array Elements

```
public static int sumArray(int[] numbers) {
    int sum = 0;
    for (int number : numbers) {
        sum += number;
    }
    return sum;
}
```

Pattern 2: Search Array

```
public static boolean contains(int[] numbers, int target) {
    for (int number : numbers) {
        if (number == target) {
            return true;
        }
    }
    return false;
}
```

Pattern 3: Count Elements

```
public static int countEven(int[] numbers) {
    int count = 0;
    for (int number : numbers) {
        if (number % 2 == 0) {
            count++;
        }
    }
```

```
    return count;
}
```