Q6

```
class SimpleObject {
    SimpleObject() {
        System.out.println("Hello from the SimpleObject constructor!");
    }

    public static void main(String[] args) {
        SimpleObject obj = new SimpleObject();  // Constructor runs here
    }
}
```

Q7

```
class Number {
    private double num;

    Number(double num) {
        this.num = num;
    }

    boolean isZero() {
        return num == 0;
    }

    boolean isPositive() {
        return num > 0;
    }

    boolean isNegative() {
        return num < 0;
    }

    boolean isOdd() {
        // Odd or even only makes sense for integers
        if (num % 1 != 0) return false;
        return ((int) num) % 2 != 0;
    }

    boolean isEven() {
        if (num % 1 != 0) return false;
        return ((int) num) % 2 == 0;
    }

    boolean isPrime() {
        if (num <= 1 || num % 1 != 0) return false; // prime only for integers > 1
        int n = (int) num;
        for (int i = 2; i <= n / 2; i++) {
            if (n % i == 0) return false;
        }
        return true;
    }

    boolean isAmstrong() {
```

```
        if (num % 1 != 0 || num < 0) return false;  // Armstrong only for non-negative integers
        int n = (int) num;
        int original = n;
        int digits = String.valueOf(n).length();
        int sum = 0;

        while (n > 0) {
            int digit = n % 10;
            sum += Math.pow(digit, digits);
            n /= 10;
        }
        return sum == original;
    }

    // For easy testing
    public static void main(String[] args) {
        Number number = new Number(153);
        System.out.println("Zero = " + number.isZero());
        System.out.println("Positive = " + number.isPositive());
        System.out.println("Negative = " + number.isNegative());
        System.out.println("Odd = " + number.isOdd());
        System.out.println("Even = " + number.isEven());
        System.out.println("Prime = " + number.isPrime());
        System.out.println("Amstrong = " + number.isAmstrong());
    }
}
```

Q8

```
class Room {
    int roomno;
    String roomtype;
    double roomarea;
    boolean ACmachine;

    void setData(int roomno, String roomtype, double roomarea, boolean ACmachine) {
        this.roomno = roomno;
        this.roomtype = roomtype;
        this.roomarea = roomarea;
        this.ACmachine = ACmachine;
    }

    void displayData() {
        System.out.println("Room No: " + roomno);
        System.out.println("Room Type: " + roomtype);
        System.out.println("Room Area: " + roomarea);
        System.out.println("AC Machine Present: " + ACmachine);
    }

    public static void main(String[] args) {
        Room r = new Room();
        r.setData(101, "Deluxe", 350.5, true);
        r.displayData();
```

```
    }
}
```

Q9

```
class Employee {
    double basicSalary;

    Employee(double basicSalary) {
        this.basicSalary = basicSalary;
    }

    double calculateSalary() {
        // Base salary - can be overridden
        return basicSalary;
    }
}

class Manager extends Employee {
    Manager(double basicSalary) {
        super(basicSalary);
    }

    @Override
    double calculateSalary() {
        // Let's say manager gets basic + 20% bonus + 10% allowance
        return basicSalary + (basicSalary * 0.2) + (basicSalary * 0.1);
    }
}

class Programmer extends Employee {
    Programmer(double basicSalary) {
        super(basicSalary);
    }

    @Override
    double calculateSalary() {
        // Programmer gets basic + 15% bonus
        return basicSalary + (basicSalary * 0.15);
    }
}

class TestEmployee {
    public static void main(String[] args) {
        Manager m = new Manager(50000);
        Programmer p = new Programmer(40000);

        System.out.println("Manager Salary: " + m.calculateSalary());
        System.out.println("Programmer Salary: " + p.calculateSalary());
    }
}
```