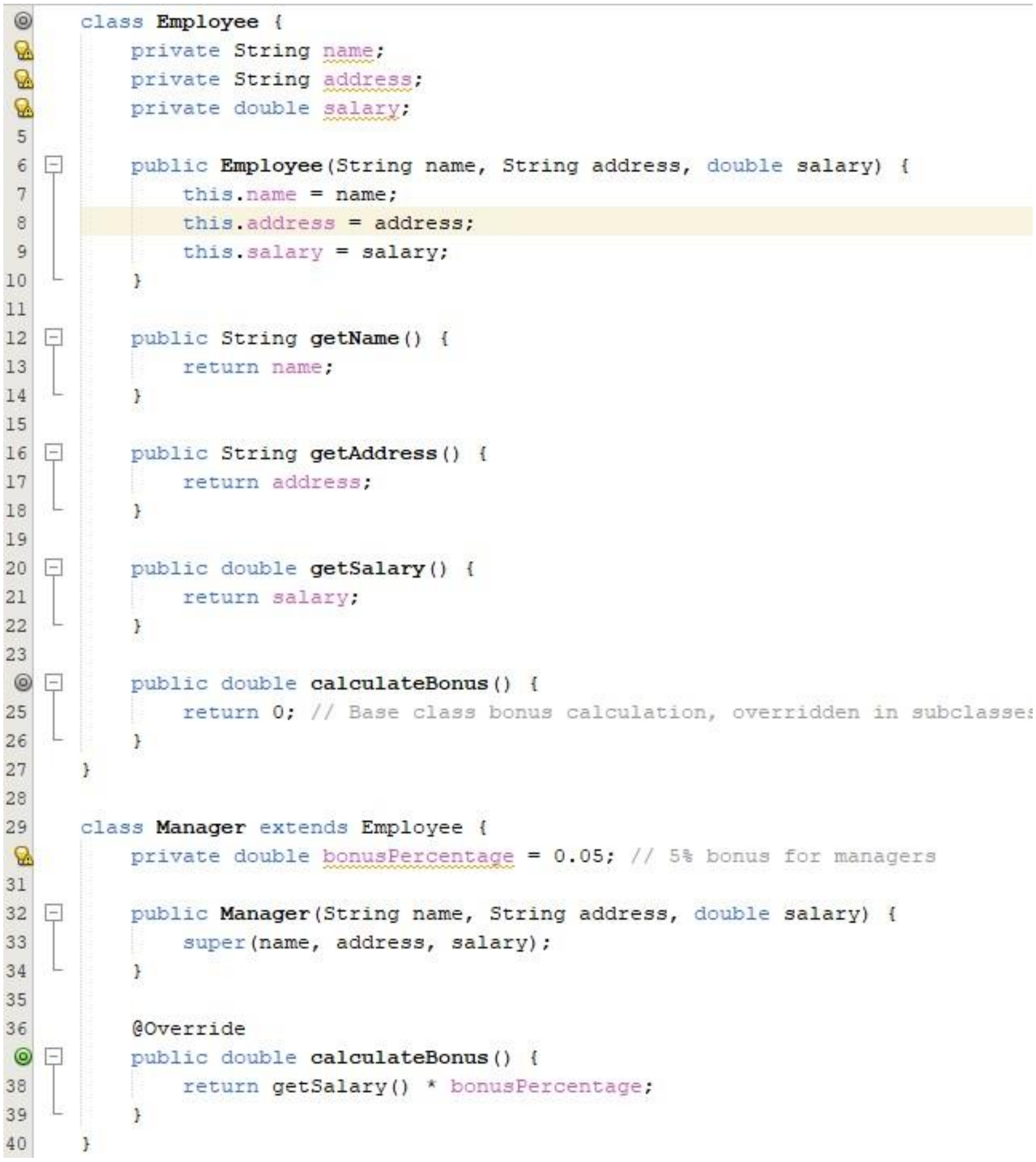
Moiz Ahmed

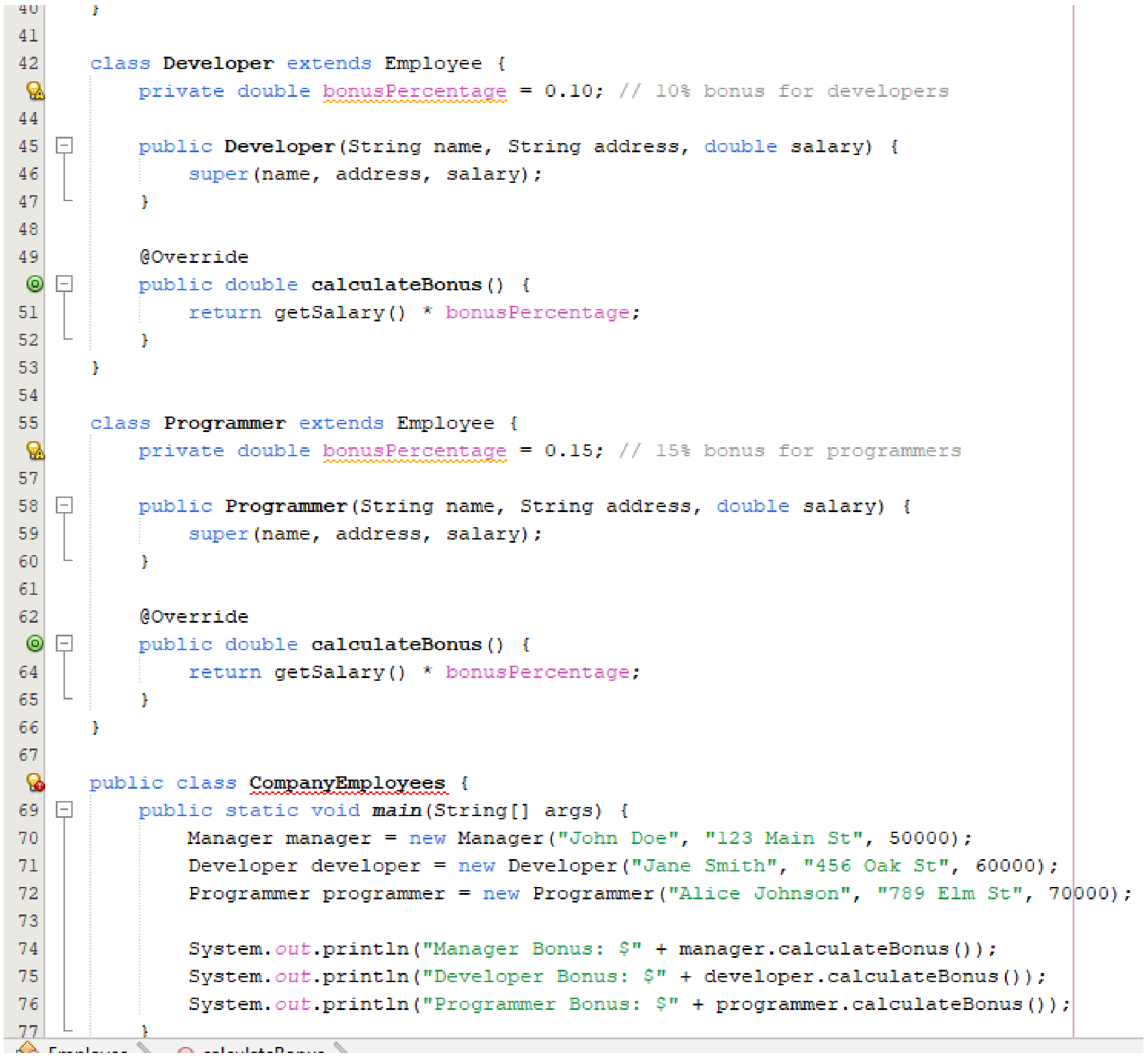
Oop

Bel 4

Task 1

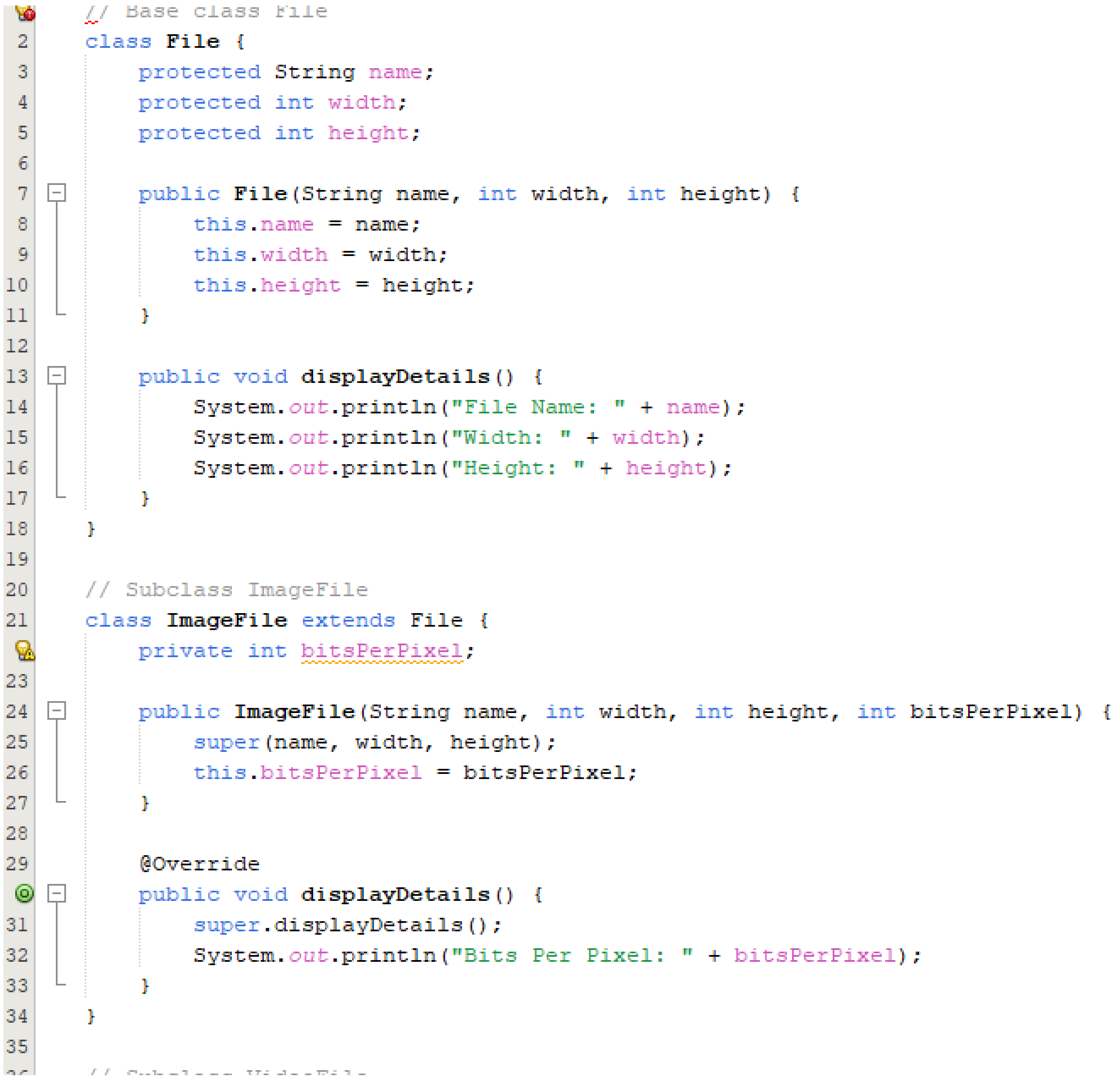


```java
class Employee {
    private String name;
    private String address;
    private double salary;

    public Employee(String name, String address, double salary) {
        this.name = name;
        this.address = address;
        this.salary = salary;
    }

    public String getName() {
        return name;
    }

    public String getAddress() {
        return address;
    }

    public double getSalary() {
        return salary;
    }

    public double calculateBonus() {
        return 0; // Base class bonus calculation, overridden in subclasse
    }
}

class Manager extends Employee {
    private double bonusPercentage = 0.05; // 5% bonus for managers

    public Manager(String name, String address, double salary) {
        super(name, address, salary);
    }

    @Override
    public double calculateBonus() {
        return getSalary() * bonusPercentage;
    }
}
```

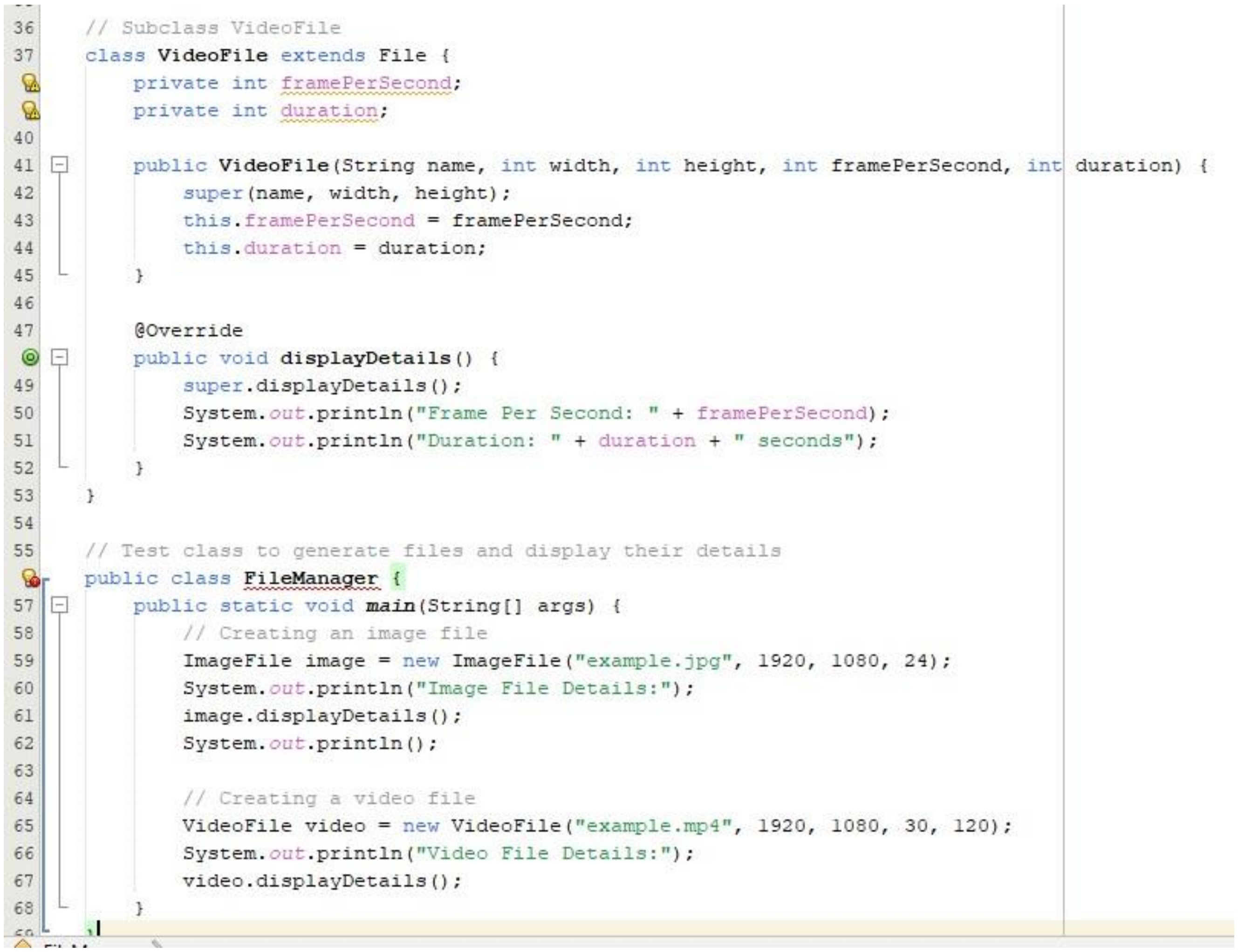


```java
}

class Developer extends Employee {
    private double bonusPercentage = 0.10; // 10% bonus for developers

    public Developer(String name, String address, double salary) {
        super(name, address, salary);
    }

    @Override
    public double calculateBonus() {
        return getSalary() * bonusPercentage;
    }
}

class Programmer extends Employee {
    private double bonusPercentage = 0.15; // 15% bonus for programmers

    public Programmer(String name, String address, double salary) {
        super(name, address, salary);
    }

    @Override
    public double calculateBonus() {
        return getSalary() * bonusPercentage;
    }
}

public class CompanyEmployees {
    public static void main(String[] args) {
        Manager manager = new Manager("John Doe", "123 Main St", 50000);
        Developer developer = new Developer("Jane Smith", "456 Oak St", 60000);
        Programmer programmer = new Programmer("Alice Johnson", "789 Elm St", 70000);

        System.out.println("Manager Bonus: $" + manager.calculateBonus());
        System.out.println("Developer Bonus: $" + developer.calculateBonus());
        System.out.println("Programmer Bonus: $" + programmer.calculateBonus());
    }
```

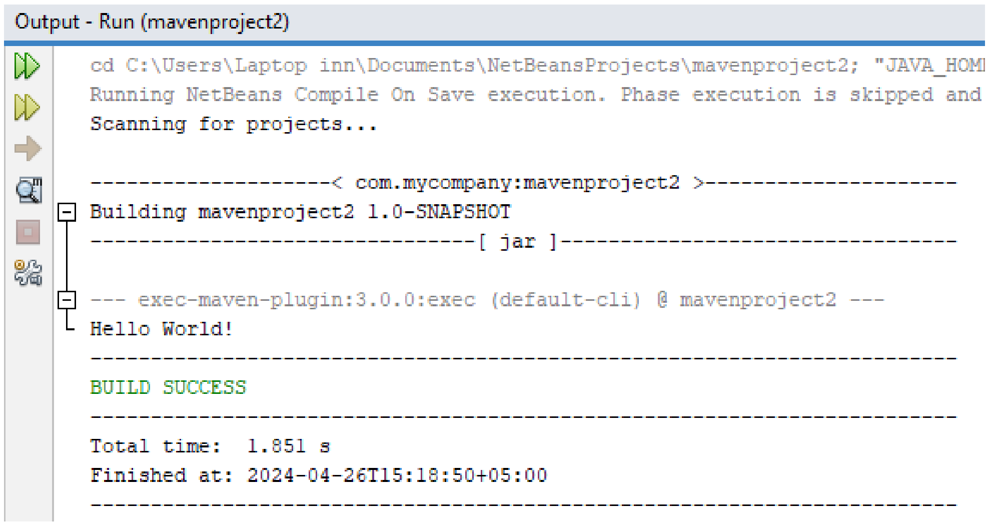
Task 2



```java
// Base class File
class File {
    protected String name;
    protected int width;
    protected int height;

    public File(String name, int width, int height) {
        this.name = name;
        this.width = width;
        this.height = height;
    }

    public void displayDetails() {
        System.out.println("File Name: " + name);
        System.out.println("Width: " + width);
        System.out.println("Height: " + height);
    }
}

// Subclass ImageFile
class ImageFile extends File {
    private int bitsPerPixel;

    public ImageFile(String name, int width, int height, int bitsPerPixel) {
        super(name, width, height);
        this.bitsPerPixel = bitsPerPixel;
    }

    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Bits Per Pixel: " + bitsPerPixel);
    }
}

```



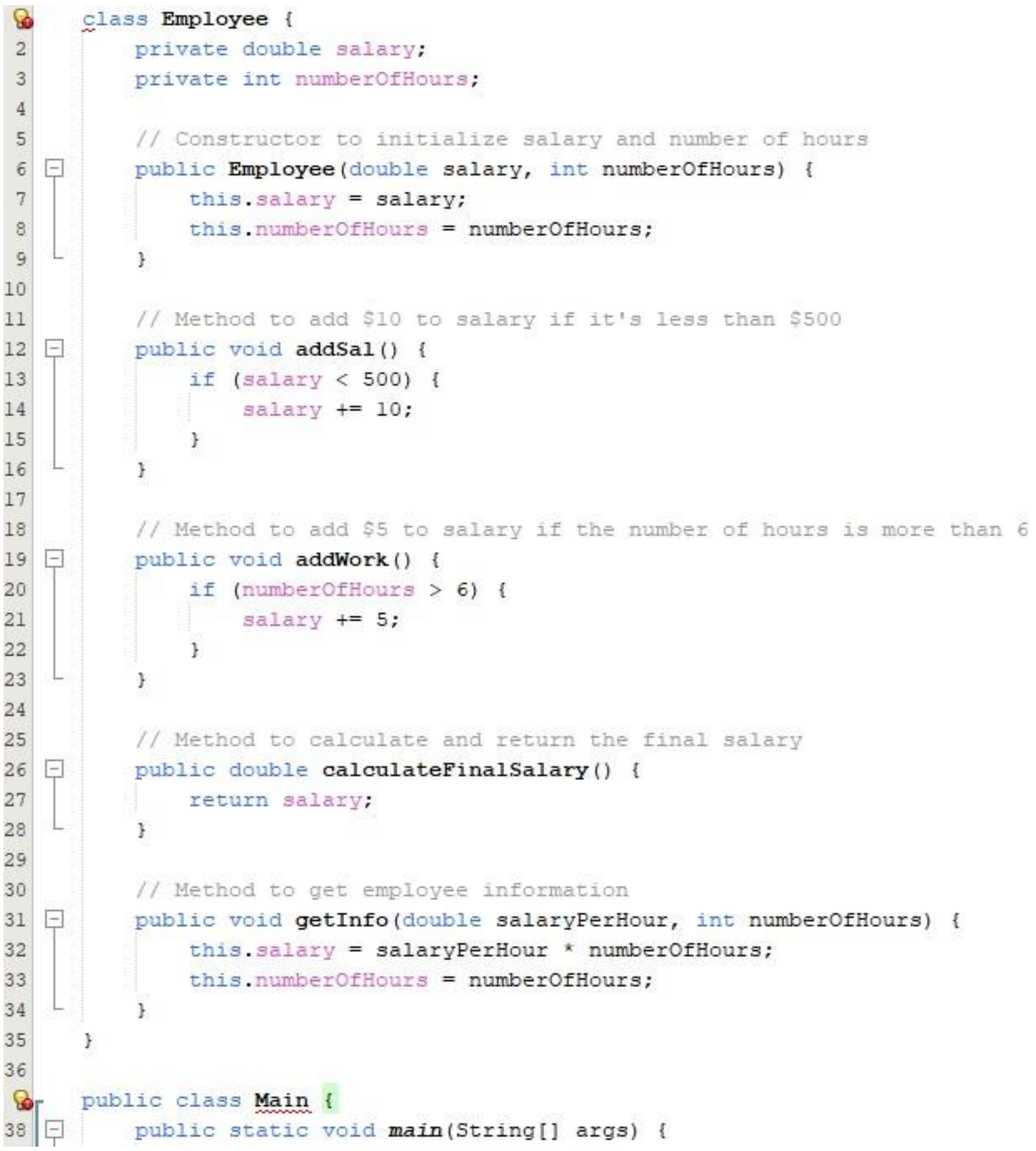
```
// Subclass VideoFile
class VideoFile extends File {
    private int framePerSecond;
    private int duration;

    public VideoFile(String name, int width, int height, int framePerSecond, int duration) {
        super(name, width, height);
        this.framePerSecond = framePerSecond;
        this.duration = duration;
    }

    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Frame Per Second: " + framePerSecond);
        System.out.println("Duration: " + duration + " seconds");
    }
}

// Test class to generate files and display their details
public class FileManager {
    public static void main(String[] args) {
        // Creating an image file
        ImageFile image = new ImageFile("example.jpg", 1920, 1080, 24);
        System.out.println("Image File Details:");
        image.displayDetails();
        System.out.println();

        // Creating a video file
        VideoFile video = new VideoFile("example.mp4", 1920, 1080, 30, 120);
        System.out.println("Video File Details:");
        video.displayDetails();
    }
}
```



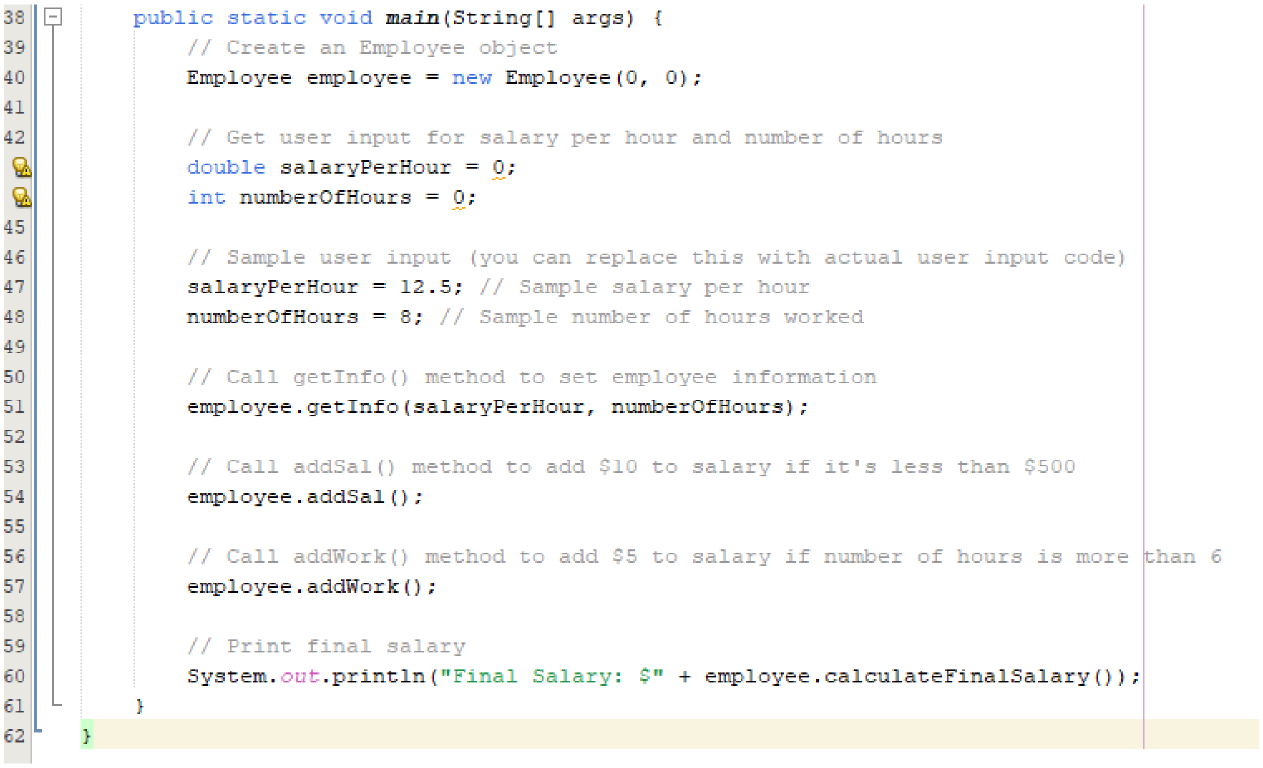
Output - Run (mavenproject2)

```
cd C:\Users\Laptop inn\Documents\NetBeansProjects\mavenproject2; "JAVA_HOM
Running NetBeans Compile On Save execution. Phase execution is skipped and
Scanning for projects...

--------------------< com.mycompany:mavenproject2 >---------------------
Building mavenproject2 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ mavenproject2 ---
Hello World!
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.851 s
Finished at: 2024-04-26T15:18:50+05:00
------------------------------------------------------------------------
```

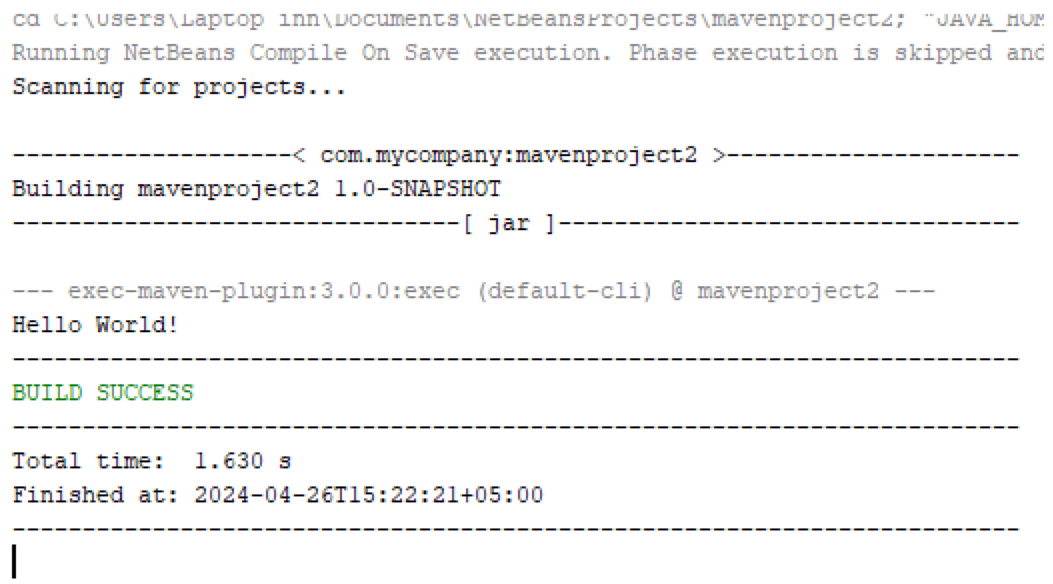
Task 3



```java
class Employee {
    private double salary;
    private int numberOfHours;

    // Constructor to initialize salary and number of hours
    public Employee(double salary, int numberOfHours) {
        this.salary = salary;
        this.numberOfHours = numberOfHours;
    }

    // Method to add $10 to salary if it's less than $500
    public void addSal() {
        if (salary < 500) {
            salary += 10;
        }
    }

    // Method to add $5 to salary if the number of hours is more than 6
    public void addWork() {
        if (numberOfHours > 6) {
            salary += 5;
        }
    }

    // Method to calculate and return the final salary
    public double calculateFinalSalary() {
        return salary;
    }

    // Method to get employee information
    public void getInfo(double salaryPerHour, int numberOfHours) {
        this.salary = salaryPerHour * numberOfHours;
        this.numberOfHours = numberOfHours;
    }
}

public class Main {
    public static void main(String[] args) {
```

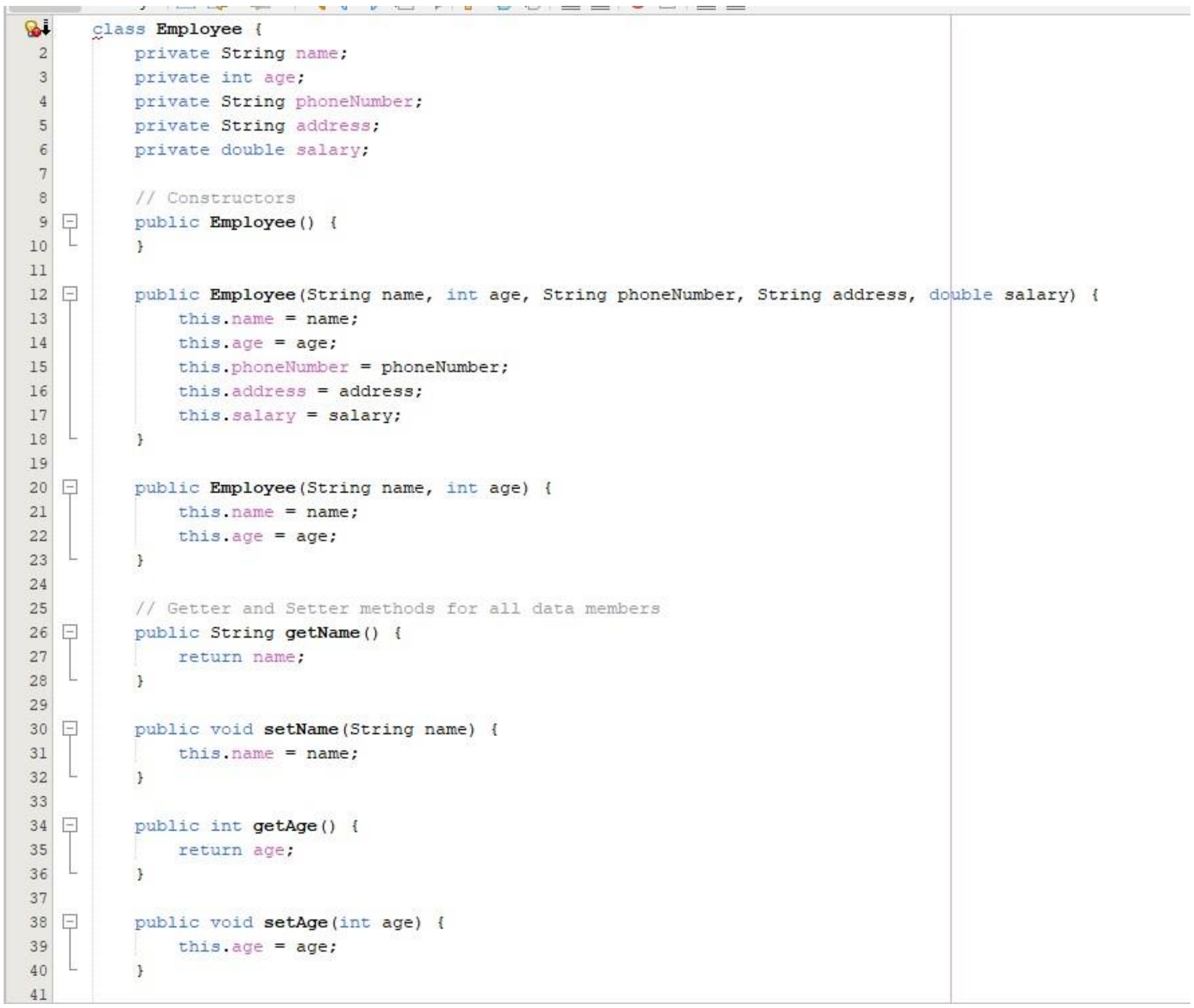


```java
    public static void main(String[] args) {
        // Create an Employee object
        Employee employee = new Employee(0, 0);

        // Get user input for salary per hour and number of hours
        double salaryPerHour = 0;
        int numberOfHours = 0;

        // Sample user input (you can replace this with actual user input code)
        salaryPerHour = 12.5; // Sample salary per hour
        numberOfHours = 8; // Sample number of hours worked

        // Call getInfo() method to set employee information
        employee.getInfo(salaryPerHour, numberOfHours);

        // Call addSal() method to add $10 to salary if it's less than $500
        employee.addSal();

        // Call addWork() method to add $5 to salary if number of hours is more than 6
        employee.addWork();

        // Print final salary
        System.out.println("Final Salary: $" + employee.calculateFinalSalary());
    }
}
```

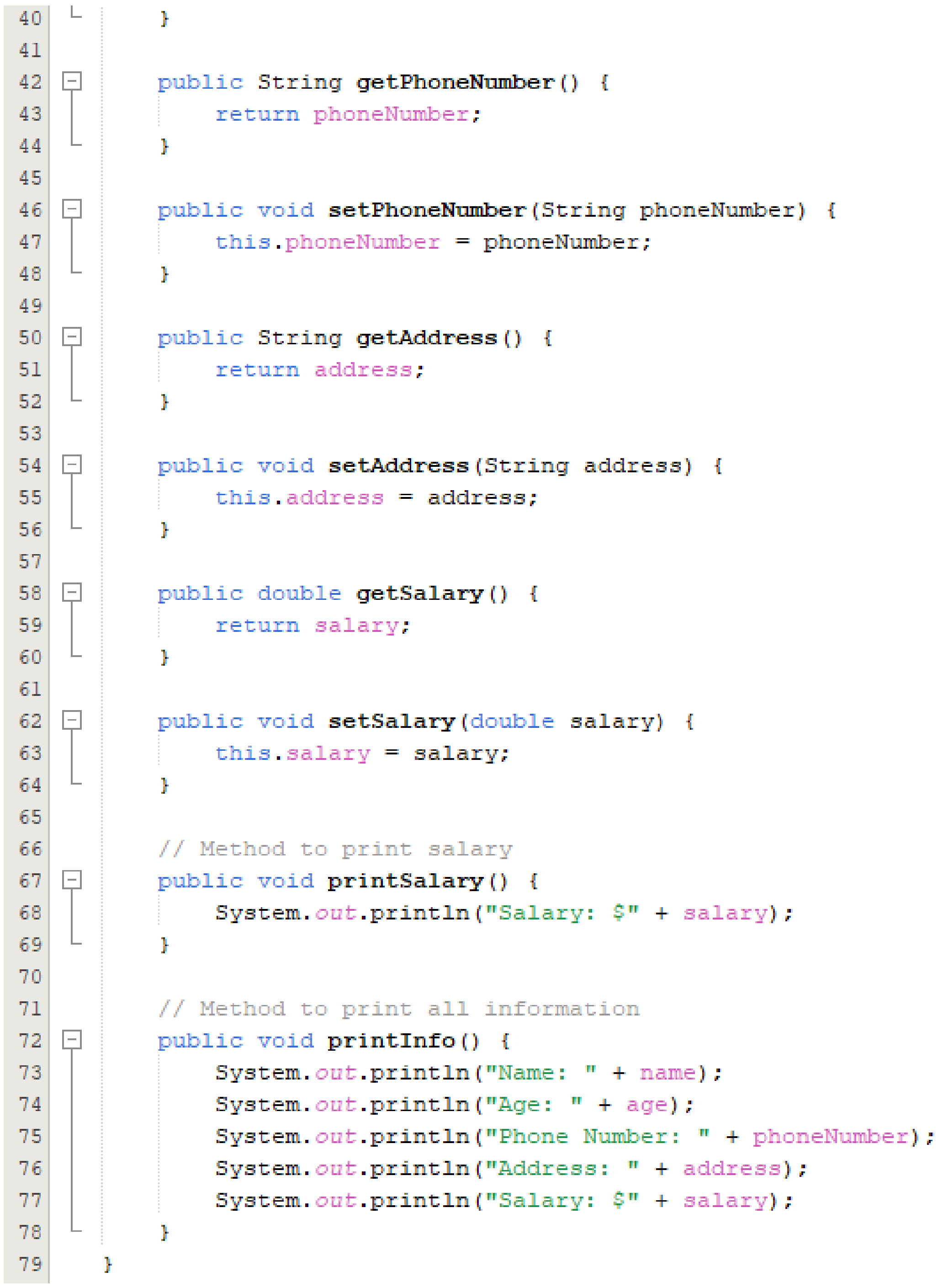


```
cd C:\Users\Laptop inn\Documents\NetBeansProjects\mavenproject2; "JAVA_HOM
Running NetBeans Compile On Save execution. Phase execution is skipped and
Scanning for projects...

--------------------< com.mycompany:mavenproject2 >--------------------
Building mavenproject2 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ mavenproject2 ---
Hello World!
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.630 s
Finished at: 2024-04-26T15:22:21+05:00
------------------------------------------------------------------------
```

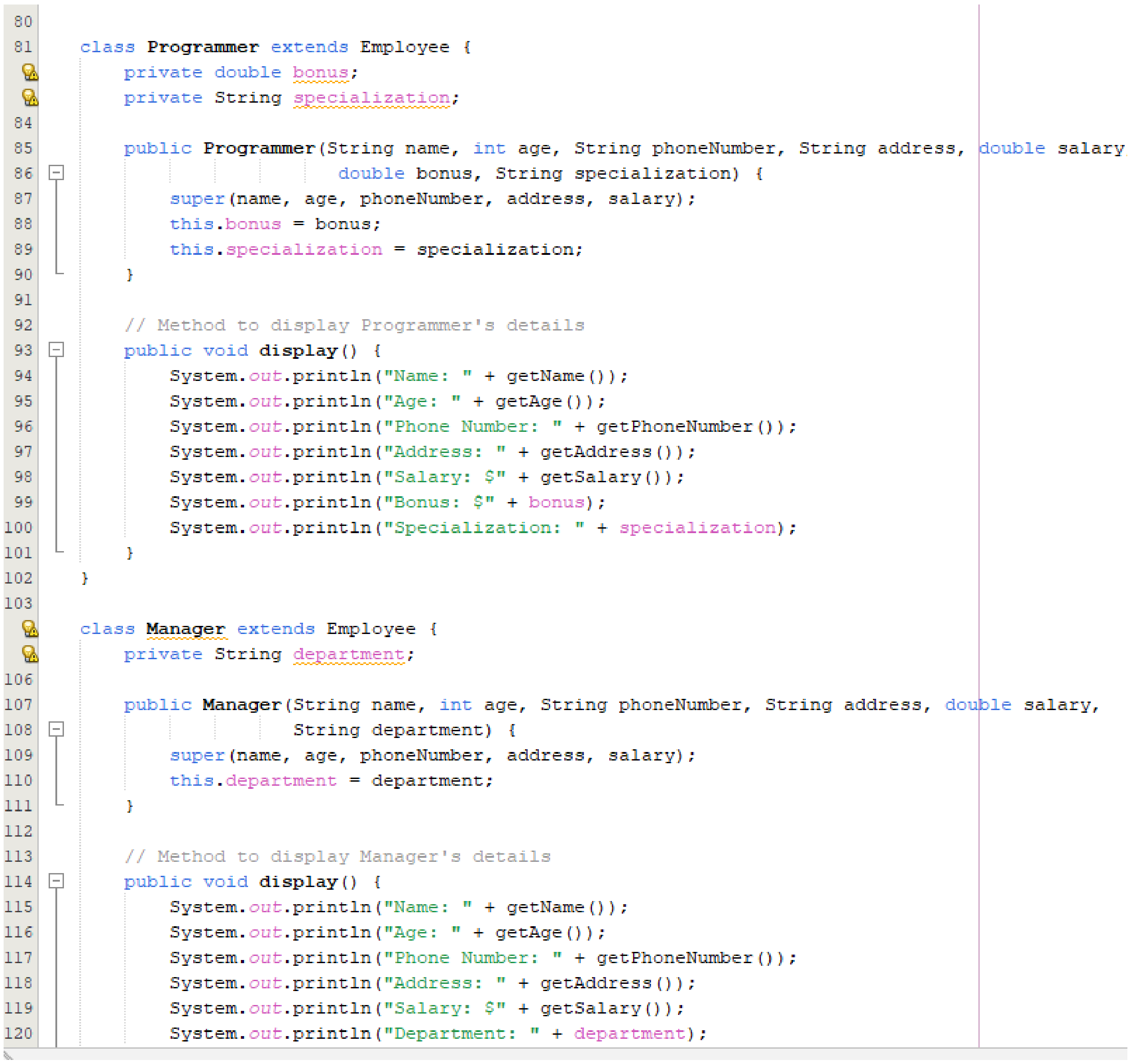
Task 4



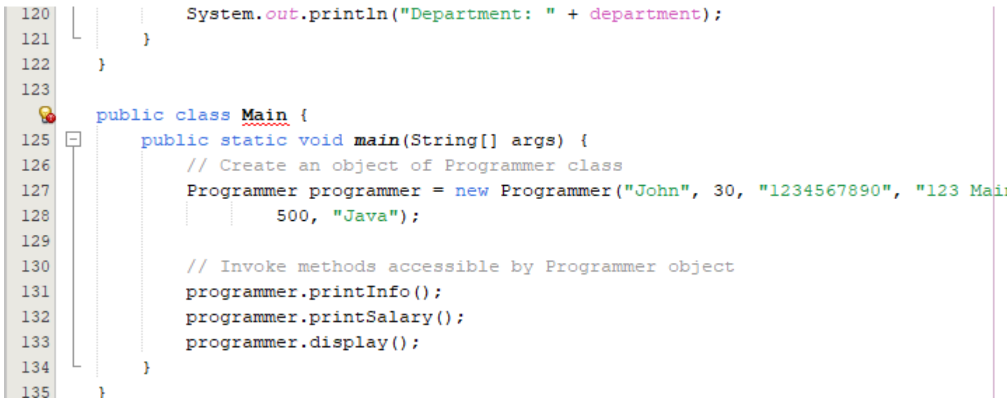
```java
class Employee {
    private String name;
    private int age;
    private String phoneNumber;
    private String address;
    private double salary;

    // Constructors
    public Employee() {
    }

    public Employee(String name, int age, String phoneNumber, String address, double salary) {
        this.name = name;
        this.age = age;
        this.phoneNumber = phoneNumber;
        this.address = address;
        this.salary = salary;
    }

    public Employee(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Getter and Setter methods for all data members
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

```



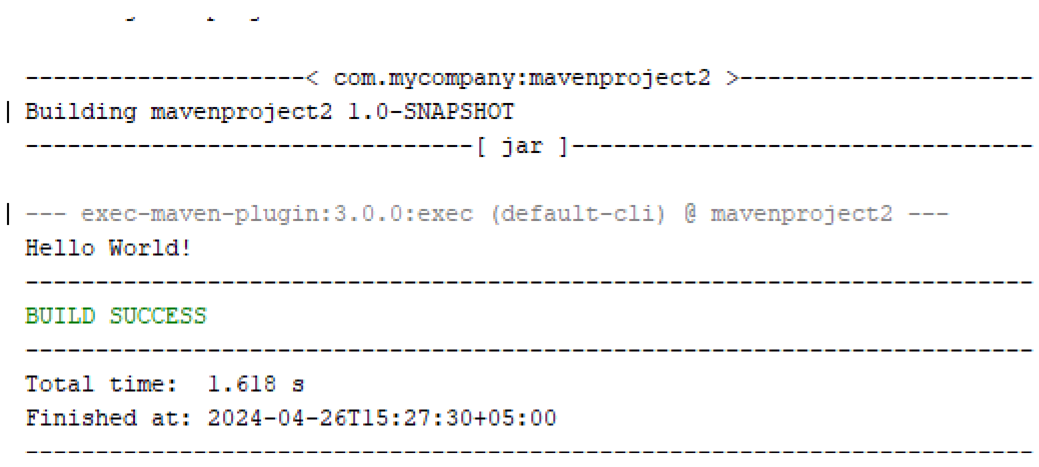
```java
    }

    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    // Method to print salary
    public void printSalary() {
        System.out.println("Salary: $" + salary);
    }

    // Method to print all information
    public void printInfo() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " + phoneNumber);
        System.out.println("Address: " + address);
        System.out.println("Salary: $" + salary);
    }
}
```



```java

class Programmer extends Employee {
    private double bonus;
    private String specialization;

    public Programmer(String name, int age, String phoneNumber, String address, double salary
                      double bonus, String specialization) {
        super(name, age, phoneNumber, address, salary);
        this.bonus = bonus;
        this.specialization = specialization;
    }

    // Method to display Programmer's details
    public void display() {
        System.out.println("Name: " + getName());
        System.out.println("Age: " + getAge());
        System.out.println("Phone Number: " + getPhoneNumber());
        System.out.println("Address: " + getAddress());
        System.out.println("Salary: $" + getSalary());
        System.out.println("Bonus: $" + bonus);
        System.out.println("Specialization: " + specialization);
    }
}

class Manager extends Employee {
    private String department;

    public Manager(String name, int age, String phoneNumber, String address, double salary,
                   String department) {
        super(name, age, phoneNumber, address, salary);
        this.department = department;
    }

    // Method to display Manager's details
    public void display() {
        System.out.println("Name: " + getName());
        System.out.println("Age: " + getAge());
        System.out.println("Phone Number: " + getPhoneNumber());
        System.out.println("Address: " + getAddress());
        System.out.println("Salary: $" + getSalary());
        System.out.println("Department: " + department);
```

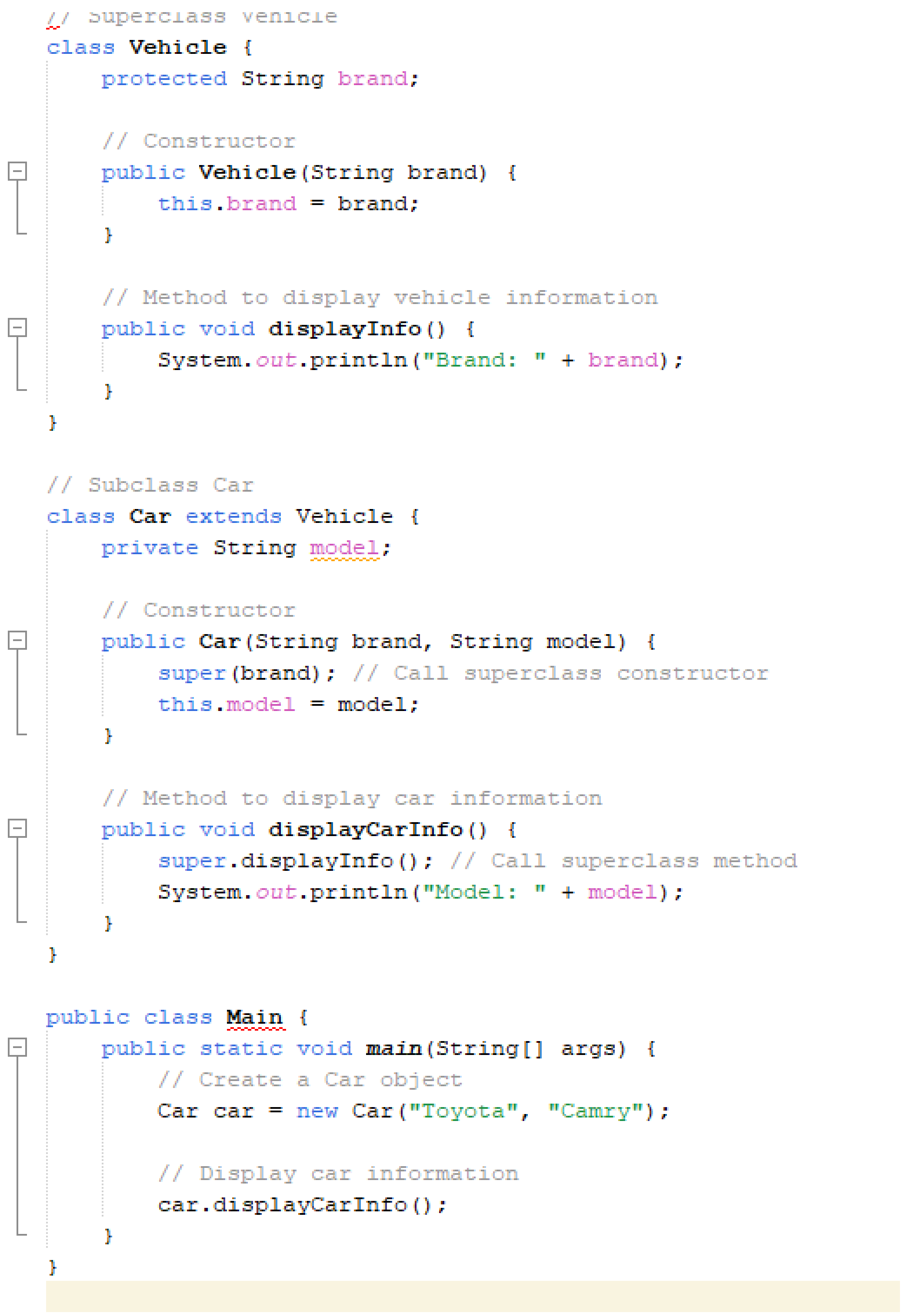


```java
        System.out.println("Department: " + department);
    }
}

public class Main {
    public static void main(String[] args) {
        // Create an object of Programmer class
        Programmer programmer = new Programmer("John", 30, "1234567890", "123 Main St", 5000,
                500, "Java");

        // Invoke methods accessible by Programmer object
        programmer.printInfo();
        programmer.printSalary();
        programmer.display();
    }
}
```



```
--------------------< com.mycompany:mavenproject2 >---------------------
Building mavenproject2 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ mavenproject2 ---
Hello World!
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.618 s
Finished at: 2024-04-26T15:27:30+05:00
------------------------------------------------------------------------
```

Task 5



```java
// Superclass Vehicle
class Vehicle {
    protected String brand;

    // Constructor
    public Vehicle(String brand) {
        this.brand = brand;
    }

    // Method to display vehicle information
    public void displayInfo() {
        System.out.println("Brand: " + brand);
    }
}

// Subclass Car
class Car extends Vehicle {
    private String model;

    // Constructor
    public Car(String brand, String model) {
        super(brand); // Call superclass constructor
        this.model = model;
    }

    // Method to display car information
    public void displayCarInfo() {
        super.displayInfo(); // Call superclass method
        System.out.println("Model: " + model);
    }
}

public class Main {
    public static void main(String[] args) {
        // Create a Car object
        Car car = new Car("Toyota", "Camry");

        // Display car information
        car.displayCarInfo();
    }
}
```