Implementation of Abstract Class using Access Modifiers

Aim

To implement an abstract class using `public`, `private`, and `protected` access modifiers and demonstrate inheritance and method overriding in Java.

Problem Statement

Create an abstract class named **Person** for a university management system. The class must contain the following instance variables:

- `personId` (type `int`)
- `name` (type `String`)
- `role` (type `String`)

The variables `personId` and `name` should be declared as **private**, and the variable `role` should be declared as **protected**. Include a constructor that initializes the `personId` and `name`. Provide **public getter and setter methods** for all variables, but ensure that `role` can only be modified inside subclasses using **protected access**.

Declare an abstract method named **calculateMonthlyAllowance()** that returns a `double` value.

Create three subclasses named **StudentPerson**, **StaffPerson**, and **ResearchPerson** that extend the `Person` class. Each subclass must implement the `calculateMonthlyAllowance()` method with different logic:

- StudentPerson → 2000
- StaffPerson → 8000
- ResearchPerson → 12000

Each subclass should set its own `role` value using the protected variable from the parent class.

Write a test application named **PersonDemo** that creates an array of at least six `Person` objects containing different combinations of `StudentPerson`, `StaffPerson`, and `ResearchPerson`. Display the `personId`, `name`, `role`, and monthly allowance for each object using the appropriate methods.

Save the files as:

- Person.java
- StudentPerson.java
- StaffPerson.java
- ResearchPerson.java
- PersonDemo.java

Source Code

Person.java

```
public abstract class Person {

    private int personId;
    private String name;
    protected String role;

    public Person(int personId, String name) {
        this.personId = personId;
        this.name = name;
    }

    public int getPersonId() {
        return personId;
    }

    public void setPersonId(int personId) {
        this.personId = personId;
    }

    public String getName() {
        return name;
    }

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

```
    public String getRole() {
        return role;
    }

    protected void setRole(String role) {
        this.role = role;
    }

    public abstract double calculateMonthlyAllowance();
}
```

StudentPerson.java

```
public class StudentPerson extends Person {

    public StudentPerson(int personId, String name) {
        super(personId, name);
        setRole("Student");
    }

    @Override
    public double calculateMonthlyAllowance() {
        return 2000;
    }
}
```

StaffPerson.java

```
public class StaffPerson extends Person {

    public StaffPerson(int personId, String name) {
        super(personId, name);
        setRole("Staff");
    }

    @Override
    public double calculateMonthlyAllowance() {
        return 8000;
```

```
    }
}
```

ResearchPerson.java

```
public class ResearchPerson extends Person {

    public ResearchPerson(int personId, String name) {
        super(personId, name);
        setRole("Researcher");
    }

    @Override
    public double calculateMonthlyAllowance() {
        return 12000;
    }
}
```

PersonDemo.java

```
public class PersonDemo {

    public static void main(String[] args) {

        Person[] people = new Person[6];

        people[0] = new StudentPerson(101, "Ajay");
        people[1] = new StudentPerson(102, "Kiran");
        people[2] = new StaffPerson(201, "Meena");
        people[3] = new StaffPerson(202, "Ravi");
        people[4] = new ResearchPerson(301, "Anitha");
        people[5] = new ResearchPerson(302, "Suresh");

        for (Person p : people) {
            System.out.println("ID: " + p.getPersonId());
            System.out.println("Name: " + p.getName());
            System.out.println("Role: " + p.getRole());
            System.out.println("Monthly Allowance: ₹" + p.calculateMonthl
            System.out.println("---------------------------------");
```

```
        }
    }
}
```

Sample Output

```
ID: 101
Name: Ajay
Role: Student
Monthly Allowance: ₹2000
---------------------------------
ID: 102
Name: Kiran
Role: Student
Monthly Allowance: ₹2000
---------------------------------
ID: 201
Name: Meena
Role: Staff
Monthly Allowance: ₹8000
---------------------------------
ID: 202
Name: Ravi
Role: Staff
Monthly Allowance: ₹8000
---------------------------------
ID: 301
Name: Anitha
Role: Researcher
Monthly Allowance: ₹12000
---------------------------------
ID: 302
Name: Suresh
Role: Researcher
Monthly Allowance: ₹12000
---------------------------------
```

Result

Thus, the abstract class **Person** and its subclasses were successfully implemented using access modifiers, inheritance, and method overriding in Java.
