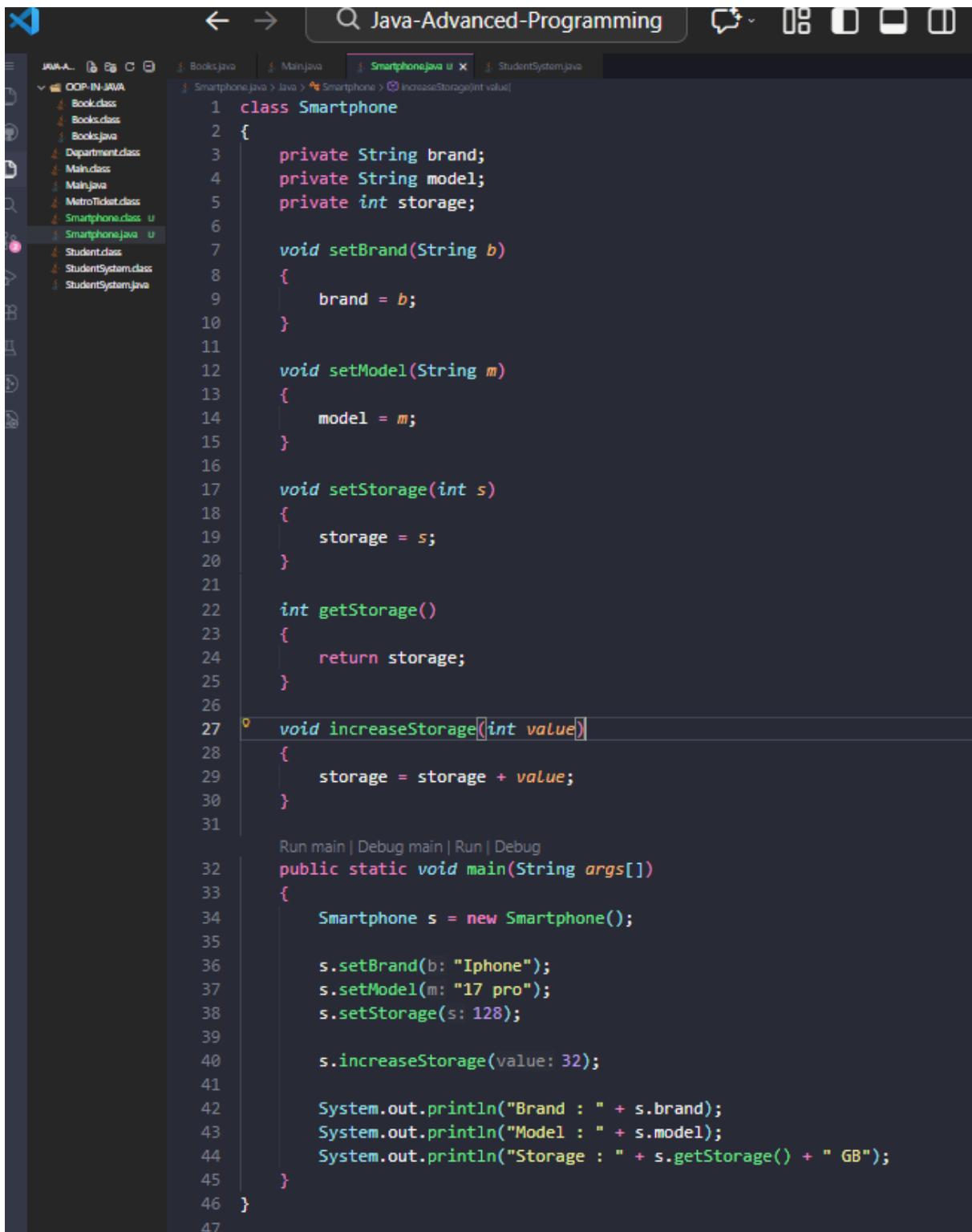


Problem: 01. Write a Java program to create a class called Smartphone with private instance variables brand, model, and storageCapacity. Provide public getter and setter methods to access and modify these variables. Add a method called increaseStorage() that takes an integer value and increases the storageCapacity by that value.



The screenshot shows a Java IDE interface with the following details:

- Title Bar:** Java-Advanced-Programming
- Project Explorer:** Shows a package named "OOP-IN-JAVA" containing several classes: Book.class, Books.class, Books.java, Department.class, Main.class, MainJava, MetroTicket.class, Smartphone.class, Student.class, StudentSystem.class, and StudentSystem.java. The "Smartphone.java" file is currently selected.
- Code Editor:** Displays the code for the "Smartphone.java" file. The code defines a class "Smartphone" with private instance variables "brand", "model", and "storage". It includes four public methods: "setBrand", "setModel", "setStorage", and "increaseStorage". The "increaseStorage" method adds an integer value to the current storage capacity. A main method is also present, creating an instance of "Smartphone" and demonstrating the use of the class methods.

```
1 class Smartphone
2 {
3     private String brand;
4     private String model;
5     private int storage;
6
7     void setBrand(String b)
8     {
9         brand = b;
10    }
11
12    void setModel(String m)
13    {
14        model = m;
15    }
16
17    void setStorage(int s)
18    {
19        storage = s;
20    }
21
22    int getStorage()
23    {
24        return storage;
25    }
26
27    void increaseStorage(int value)
28    {
29        storage = storage + value;
30    }
31
32    public static void main(String args[])
33    {
34        Smartphone s = new Smartphone();
35
36        s.setBrand(b: "Iphone");
37        s.setModel(m: "17 pro");
38        s.setStorage(s: 128);
39
40        s.increaseStorage(value: 32);
41
42        System.out.println("Brand : " + s.brand);
43        System.out.println("Model : " + s.model);
44        System.out.println("Storage : " + s.getStorage() + " GB");
45    }
46 }
```

OutPut:



A screenshot of a terminal window from a code editor. The window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, SPELL CHECKER, and GITLENS. The TERMINAL tab is active. The output shows the following text:

```
Storage : 160 GB
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac Smartphone.java } ; if ($?) { java Smartphone }
Brand : Iphone
Model : 17 pro
Storage : 160 GB
PS D:\Java-Advanced-Programming>
```

Problem: 02. Create a class named Complex that must have two integer data members (real, and imaginary). Create two constructors, one Read function to take keyboard input, one Add (return object), function, and one Display function to print results. The Add function must take one object as an argument. Watch the input and output section for better understanding.

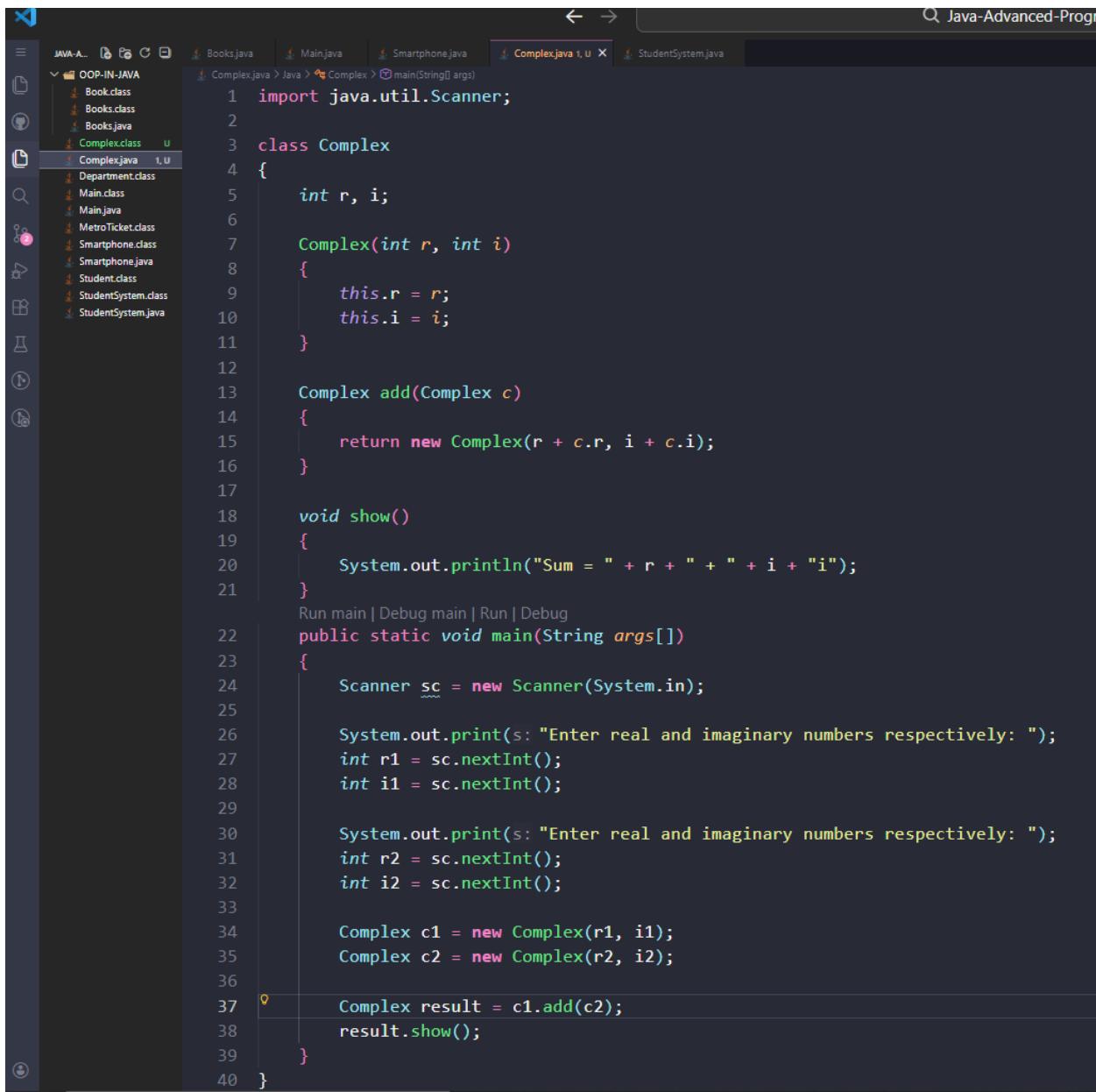
Input:

Enter real and imaginary numbers respectively: 16 7

Enter real and imaginary numbers respectively: 5 8

Output:

Sum = 21 + 15i



```
import java.util.Scanner;

class Complex
{
    int r, i;

    Complex(int r, int i)
    {
        this.r = r;
        this.i = i;
    }

    Complex add(Complex c)
    {
        return new Complex(r + c.r, i + c.i);
    }

    void show()
    {
        System.out.println("Sum = " + r + " + " + i + "i");
    }
}

public static void main(String args[])
{
    Scanner sc = new Scanner(System.in);

    System.out.print("Enter real and imaginary numbers respectively: ");
    int r1 = sc.nextInt();
    int i1 = sc.nextInt();

    System.out.print("Enter real and imaginary numbers respectively: ");
    int r2 = sc.nextInt();
    int i2 = sc.nextInt();

    Complex c1 = new Complex(r1, i1);
    Complex c2 = new Complex(r2, i2);

    Complex result = c1.add(c2);
    result.show();
}
```

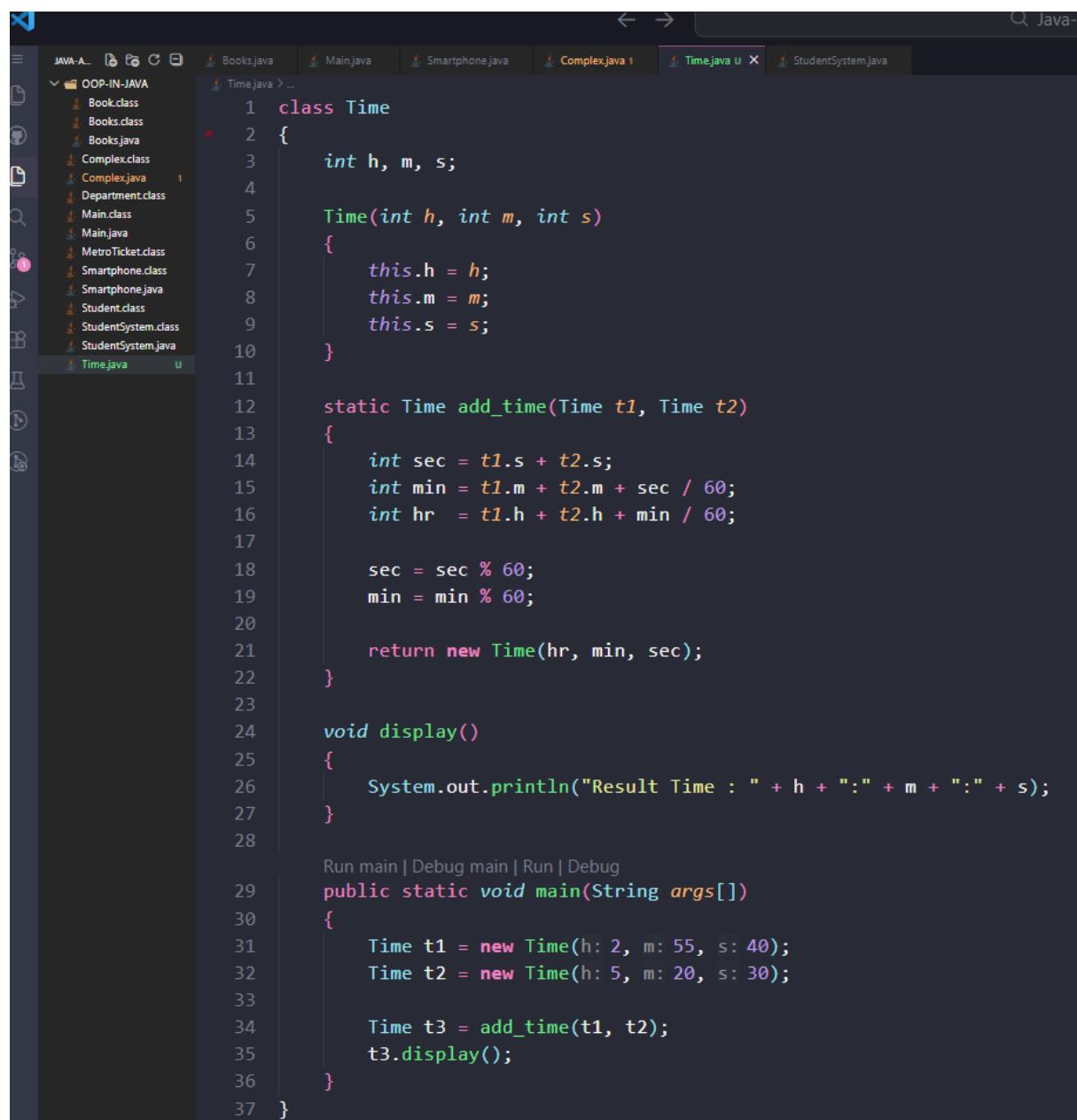
Output:

```
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if (?) { javac Complex.java } ; if (?) { java Complex }
Enter real and imaginary numbers respectively: 2
2
Enter real and imaginary numbers respectively: 45
4
Sum = 47 + 6i
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if (?) { javac Complex.java } ; if (?) { java Complex }
Enter real and imaginary numbers respectively: 16 7
Enter real and imaginary numbers respectively: 5 8
Sum = 21 + 15i
PS D:\Java-Advanced-Programming>
```

Problem:03. Create a class named Time that must have three integer data members (hours, minutes, and seconds). Create two constructors, one member function named add_time (return object), and one display function to print the time in 11:59:59 format. The add_time function must take two objects as arguments. The main function calls the add_time function to add two-time objects and store the result in a third object. Use the display function to print the result on the console.

Input: 2 55 40, 5 20 30

Output: 8:16:10

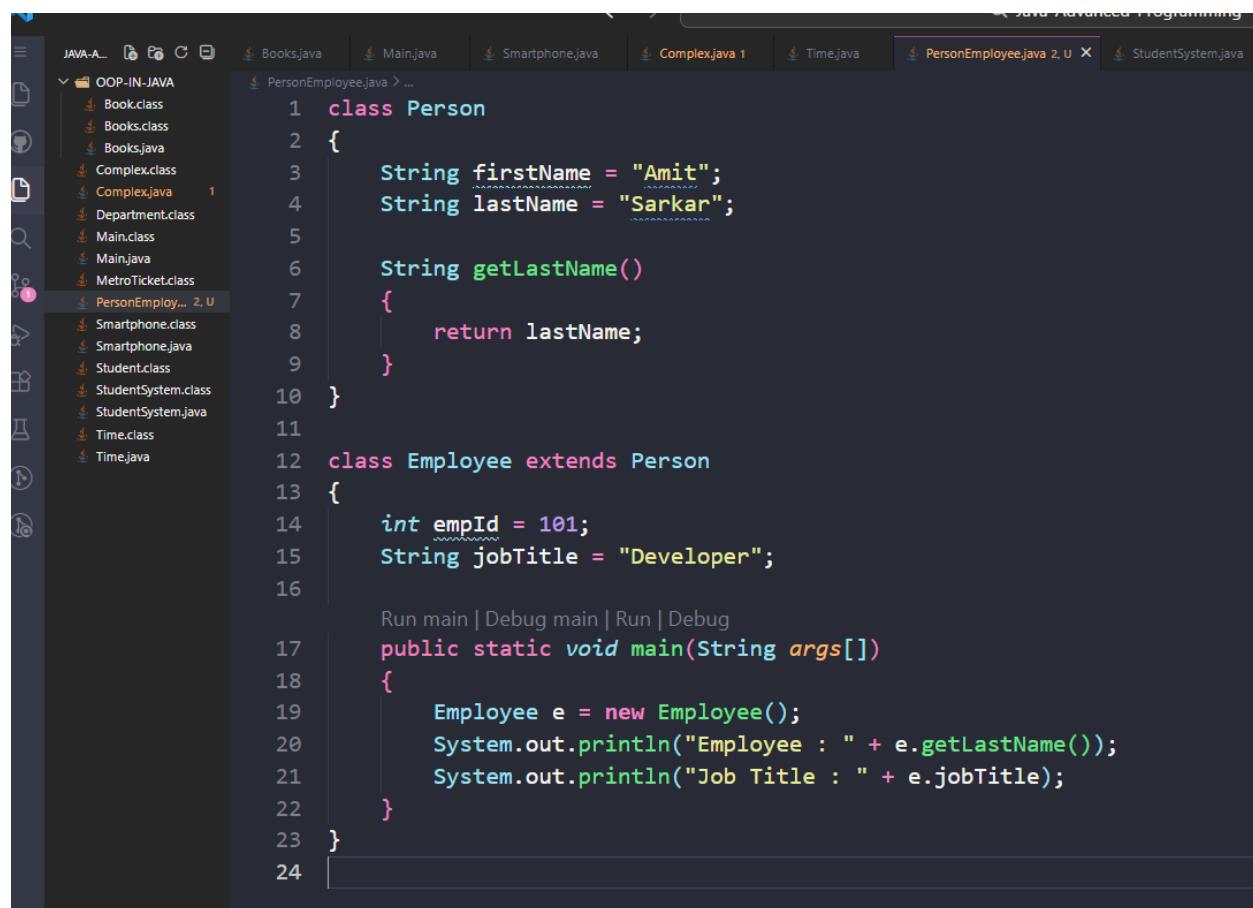


```
1  class Time
2  {
3      int h, m, s;
4
5      Time(int h, int m, int s)
6      {
7          this.h = h;
8          this.m = m;
9          this.s = s;
10     }
11
12     static Time add_time(Time t1, Time t2)
13     {
14         int sec = t1.s + t2.s;
15         int min = t1.m + t2.m + sec / 60;
16         int hr = t1.h + t2.h + min / 60;
17
18         sec = sec % 60;
19         min = min % 60;
20
21         return new Time(hr, min, sec);
22     }
23
24     void display()
25     {
26         System.out.println("Result Time : " + h + ":" + m + ":" + s);
27     }
28
29     Run main | Debug main | Run | Debug
30     public static void main(String args[])
31     {
32         Time t1 = new Time(h: 2, m: 55, s: 40);
33         Time t2 = new Time(h: 5, m: 20, s: 30);
34
35         Time t3 = add_time(t1, t2);
36         t3.display();
37     }
}
```

Output:

```
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Amitkumarsarkar/Java-Advanced-Programming-2026.git
  4e3adb1..364b41d  main -> main
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac Time.java } ; if ($?) { java Time }
Result Time : 8:16:10
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac Time.java } ; if ($?) { java Time }
Result Time : 8:16:10
```

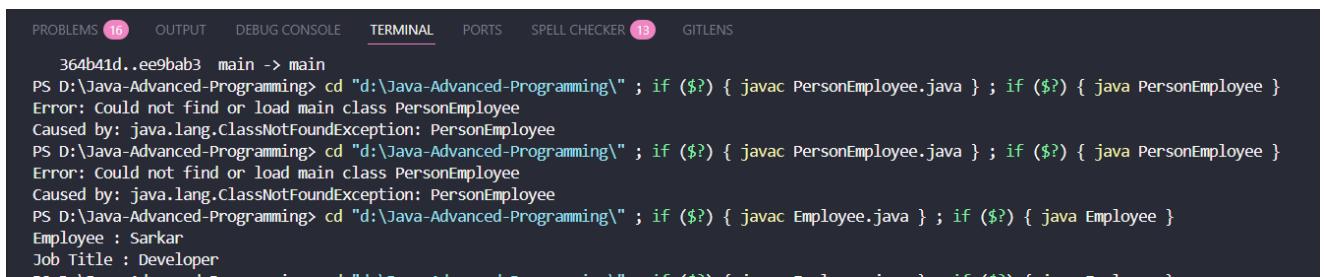
Problem:04. Write a Java program to create a class known as Person with methods called getFirstName() and getLastName(). Create a child class called Employee that adds a new method named getEmployeeId() and accesses the getLastName() method to include the employee's job title.



The screenshot shows an IDE interface with a dark theme. On the left is a file explorer window titled 'JAVA-A...' containing several Java files under a folder 'OOP-IN-JAVA'. The central area displays the code for 'PersonEmployee.java' (version 2, U). The code defines a 'Person' class with attributes for first and last names, and a 'getLastName()' method. It then defines an 'Employee' class that extends 'Person' and adds an 'empId' attribute and a 'jobTitle' attribute. The 'main' method creates an 'Employee' object and prints its last name and job title.

```
1  class Person
2  {
3      String firstName = "Amit";
4      String lastName = "Sarkar";
5
6      String getLastName()
7      {
8          return lastName;
9      }
10 }
11
12 class Employee extends Person
13 {
14     int empId = 101;
15     String jobTitle = "Developer";
16
17     Run main | Debug main | Run | Debug
18     public static void main(String args[])
19     {
20         Employee e = new Employee();
21         System.out.println("Employee : " + e.getLastName());
22         System.out.println("Job Title : " + e.jobTitle);
23     }
24 }
```

Output:



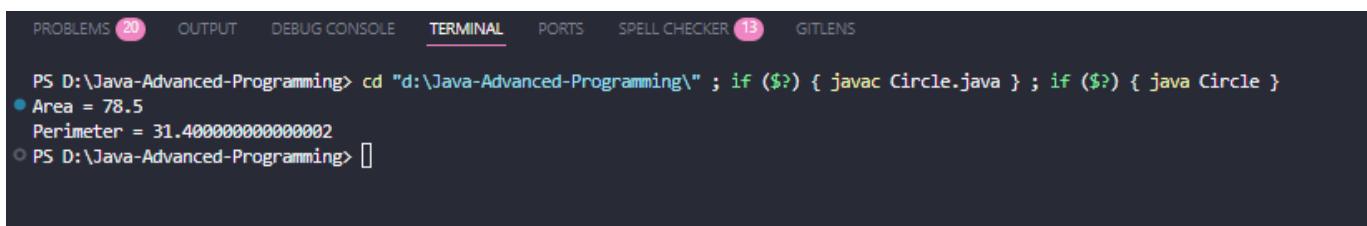
The screenshot shows a terminal window within a code editor interface. The tabs at the top are PROBLEMS (16), OUTPUT, DEBUG CONSOLE, TERMINAL (selected), PORTS, SPELL CHECKER (1), and GITLENS. The terminal output is as follows:

```
364b41d..ee9bab3 main -> main
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac PersonEmployee.java } ; if (?) { java PersonEmployee }
Error: Could not find or load main class PersonEmployee
Caused by: java.lang.ClassNotFoundException: PersonEmployee
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac PersonEmployee.java } ; if (?) { java PersonEmployee }
Error: Could not find or load main class PersonEmployee
Caused by: java.lang.ClassNotFoundException: PersonEmployee
PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if (?) { javac Employee.java } ; if (?) { java Employee }
Employee : Sarkar
Job Title : Developer
```

Problem:05. Write a Java program to create a class called Shape with methods called getPerimeter() and getArea(). Create a child class called Circle that uses the getPerimeter() and getArea() methods to calculate the area and perimeter of a circle.

```
1  class Shape
2  {
3      double getArea()
4      {
5          return 0;
6      }
7
8      double getPerimeter()
9      {
10         return 0;
11     }
12 }
13
14 class Circle extends Shape
15 {
16     double r = 5;
17
18     double getArea()
19     {
20         return 3.14 * r * r;
21     }
22
23     double getPerimeter()
24     {
25         return 2 * 3.14 * r;
26     }
27
28     Run main | Debug main | Run | Debug
29     public static void main(String args[])
30     {
31         Circle c = new Circle();
32         System.out.println("Area = " + c.getArea());
33         System.out.println("Perimeter = " + c.getPerimeter());
34     }
}
```

OutPut:

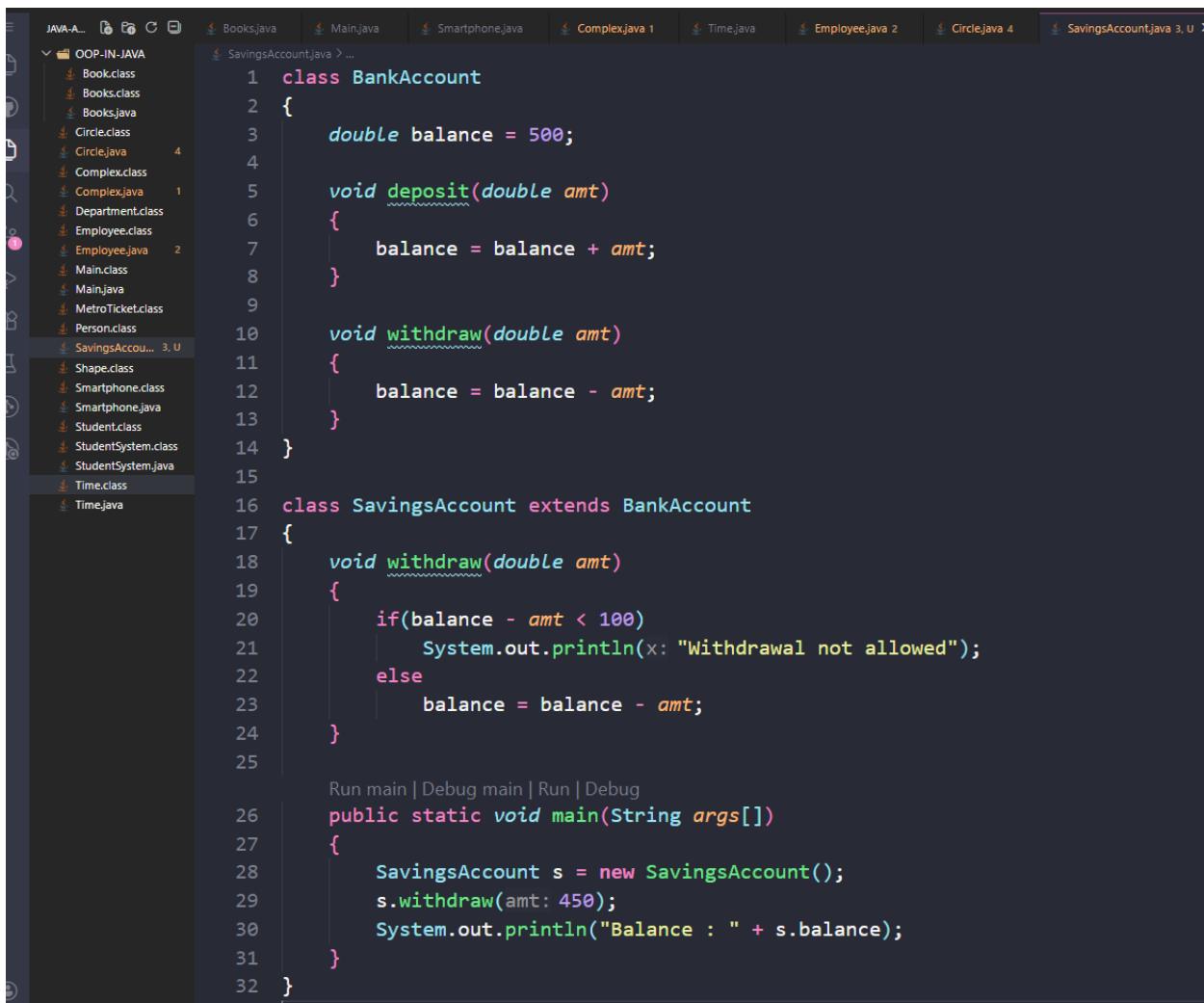


```
PROBLEMS 20 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 13 GITLENS

PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac Circle.java } ; if ($?) { java Circle }

● Area = 78.5
Perimeter = 31.400000000000002
○ PS D:\Java-Advanced-Programming> []
```

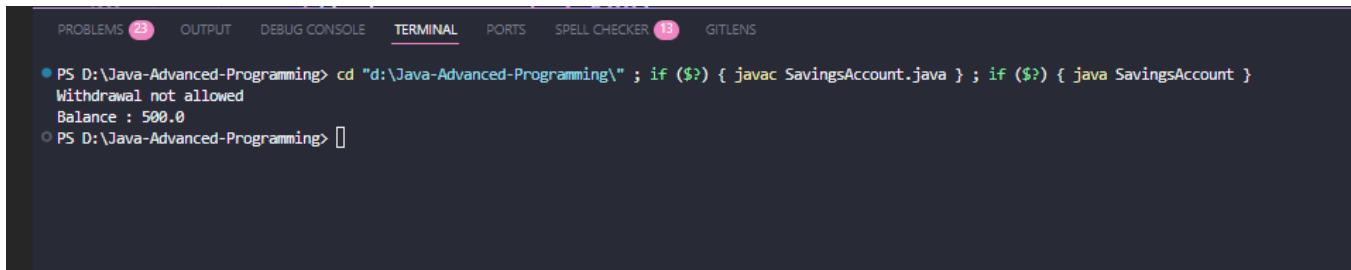
Problem:06. Write a Java program to create a class known as BankAccount with methods called deposit() and withdraw(). Create a child class called SavingsAccount that accesses the withdraw() method to prevent withdrawals if the account balance falls below one hundred.



```
JAVA-A... Books.java Main.java Smartphone.java Complex.java 1 Time.java Employee.java 2 Circle.java 4 SavingsAccount.java 3, U

1 class BankAccount
2 {
3     double balance = 500;
4
5     void deposit(double amt)
6     {
7         balance = balance + amt;
8     }
9
10    void withdraw(double amt)
11    {
12        balance = balance - amt;
13    }
14 }
15
16 class SavingsAccount extends BankAccount
17 {
18     void withdraw(double amt)
19     {
20         if(balance - amt < 100)
21             System.out.println("Withdrawal not allowed");
22         else
23             balance = balance - amt;
24     }
25
26     Run main | Debug main | Run | Debug
27     public static void main(String args[])
28     {
29         SavingsAccount s = new SavingsAccount();
30         s.withdraw(amt: 450);
31         System.out.println("Balance : " + s.balance);
32     }
}
```

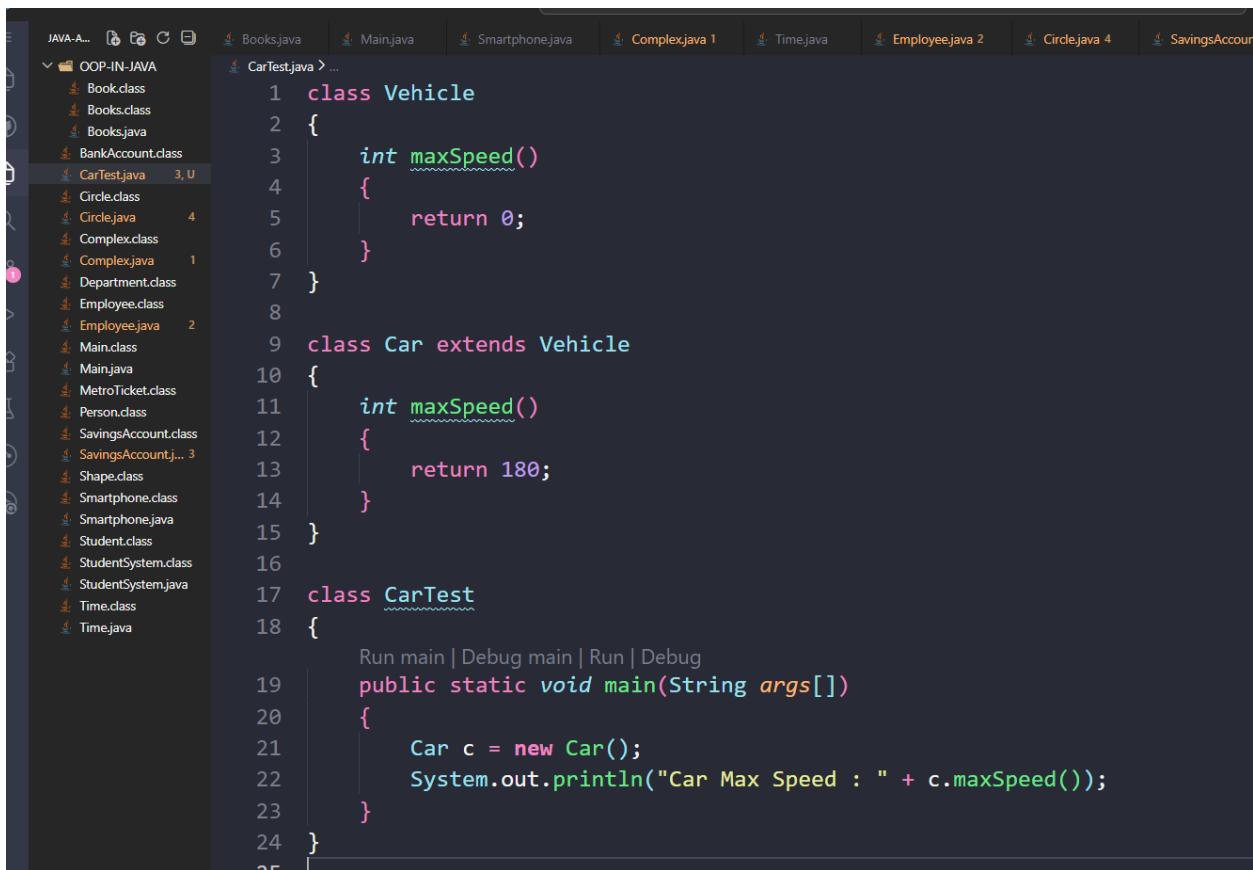
OutPut:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER GITLENS

PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac SavingsAccount.java } ; if ($?) { java SavingsAccount }
Withdrawal not allowed
Balance : 500.0
PS D:\Java-Advanced-Programming> []
```

Problem:07. Write a Java program to create a vehicle class hierarchy. The base class should be Vehicle, with child classes Truck, Car and Motorcycle. Each child class should have properties such as make, model, year, and fuel type. Implement methods for calculating fuel efficiency, distance traveled, and maximum speed.



```
JAVA-A... Booksjava Main.java Smartphone.java Complex.java 1 Time.java Employee.java 2 Circle.java 4 SavingsAccount.java

OOP-IN-JAVA
  Book.class
  Books.class
  Books.java
  BankAccount.class
  CarTest.java 3, U
  Circle.class
  Circle.java 4
  Complex.class
  Complex.java 1
  Department.class
  Employee.class
  Employee.java 2
  Main.class
  Main.java
  MetroTicket.class
  Person.class
  SavingsAccount.class
  SavingsAccount.j... 3
  Shape.class
  Smartphone.class
  Smartphone.java
  Student.class
  StudentSystem.class
  StudentSystem.java
  Time.class
  Time.java

1  class Vehicle
2  {
3      int maxSpeed()
4      {
5          return 0;
6      }
7  }
8
9  class Car extends Vehicle
10 {
11     int maxSpeed()
12     {
13         return 180;
14     }
15 }
16
17 class CarTest
18 {
    Run main | Debug main | Run | Debug
19     public static void main(String args[])
20     {
21         Car c = new Car();
22         System.out.println("Car Max Speed : " + c.maxSpeed());
23     }
24 }
```

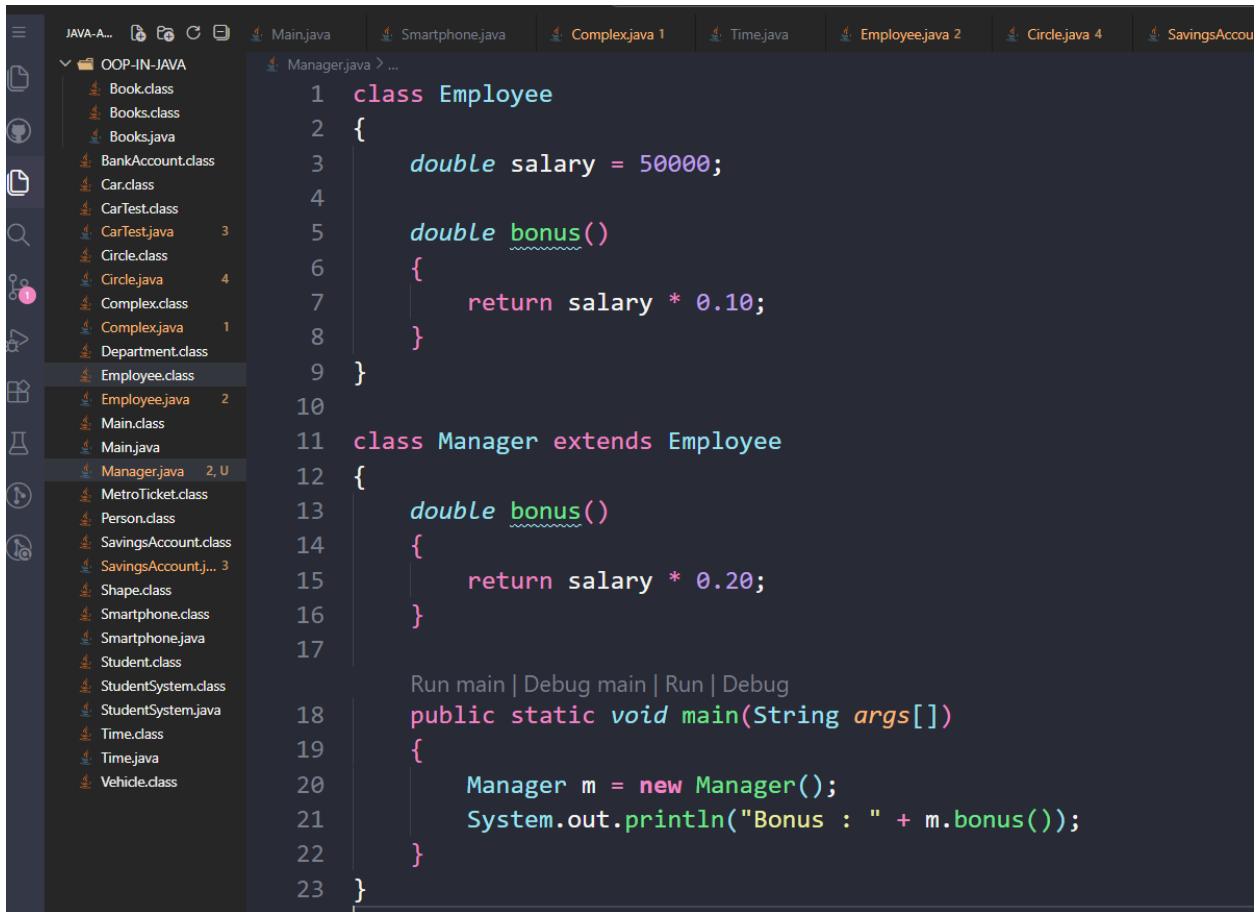
OutPut:



```
PROBLEMS 26 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 13 GITLENS

● PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac CarTest.java } ; if ($?) { java CarTest }
Car Max Speed : 180
○ PS D:\Java-Advanced-Programming>
```

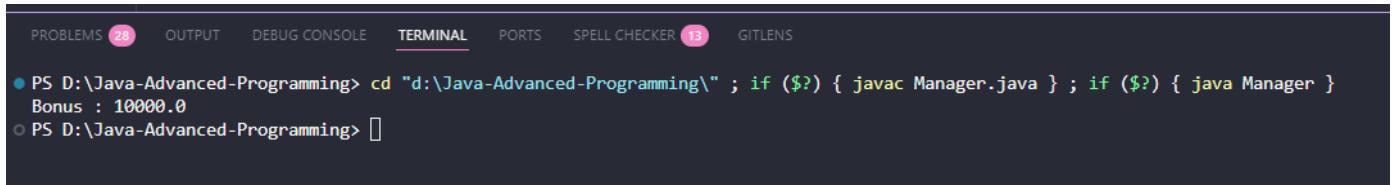
Problem:08. Write a Java program that creates a class hierarchy for employees of a company. The base class should be Employee, with child classes Manager, Developer, and Programmer. Each child class should have properties such as name, address, salary, and job title. Implement methods for calculating bonuses, generating performance reports, and managing projects.



```
JAVA-A... Main.java Smartphone.java Complex.java 1 Time.java Employee.java 2 Circle.java 4 SavingsAccou...
OOP-IN-JAVA Book.class Books.class Books.java BankAccount.class Car.class CarTest.class CarTest.java 3 Circle.class Circle.java 4 Complex.class Complex.java 1 Department.class Employee.class Employee.java 2 Main.class Main.java Manager.java 2, U MetroTicket.class Person.class SavingsAccount.class SavingsAccount.j... 3 Shape.class Smartphone.class Smartphone.java Student.class StudentSystem.class StudentSystem.java Time.class Time.java Vehicle.class

Manager.java > ...
1 class Employee
2 {
3     double salary = 50000;
4
5     double bonus()
6     {
7         return salary * 0.10;
8     }
9 }
10
11 class Manager extends Employee
12 {
13     double bonus()
14     {
15         return salary * 0.20;
16     }
17
18     Run main | Debug main | Run | Debug
19     public static void main(String args[])
20     {
21         Manager m = new Manager();
22         System.out.println("Bonus : " + m.bonus());
23     }
}
```

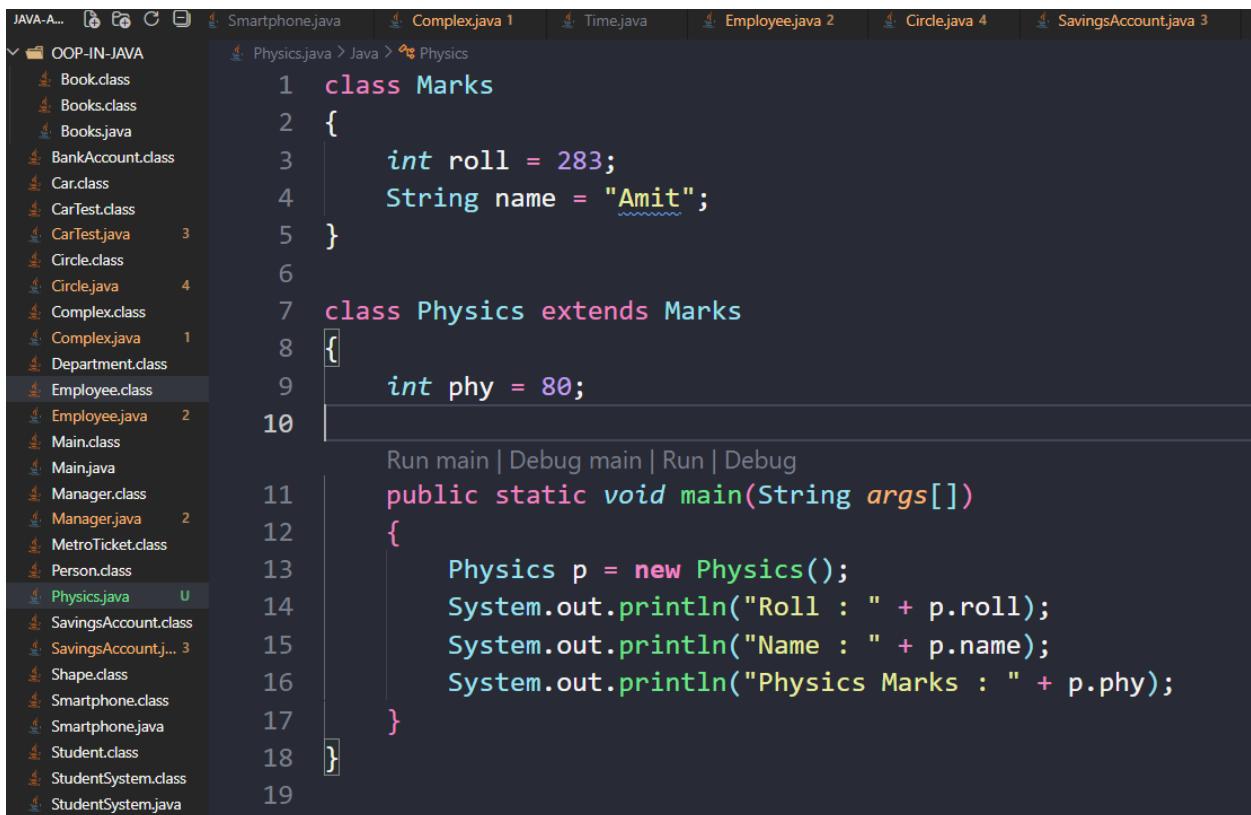
OutPut:



```
PROBLEMS 28 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 13 GITLENS

● PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if ($?) { javac Manager.java } ; if (?) { java Manager }
Bonus : 10000.0
○ PS D:\Java-Advanced-Programming> [ ]
```

Problem:09. We want to calculate the total marks of each student of a class in Physics, Chemistry and Mathematics and the average marks of the class. The number of students in the class are entered by the user. Create a class named Marks with data members for roll number, name and marks. Create three other classes inheriting the Marks class, namely Physics, Chemistry and Mathematics, which are used to define marks in individual subjects of each student.



The code editor shows a file named Physics.java with the following content:

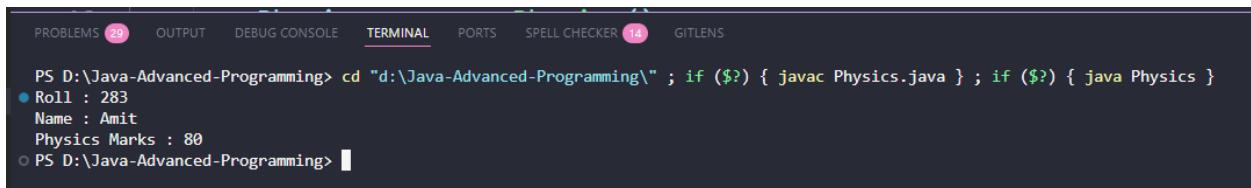
```
class Marks
{
    int roll = 283;
    String name = "Amit";
}

class Physics extends Marks
{
    int phy = 80;

    public static void main(String args[])
    {
        Physics p = new Physics();
        System.out.println("Roll : " + p.roll);
        System.out.println("Name : " + p.name);
        System.out.println("Physics Marks : " + p.phy);
    }
}
```

The sidebar on the left lists several Java files in the OOP-IN-JAVA folder, including Book.class, Books.class, Booksjava, BankAccount.class, Car.class, CarTest.class, CarTest.java, Circle.class, Circle.java, Complex.class, Complex.java, Department.class, Employee.class, Employee.java, Main.class, Main.java, Manager.class, Manager.java, MetroTicket.class, Person.class, Physics.java (which is currently selected), SavingsAccount.class, SavingsAccount.java, Shape.class, Smartphone.class, Smartphone.java, Student.class, StudentSystem.class, and StudentSystem.java.

OutPut:



```
PROBLEMS 29 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 14 GITLENS

PS D:\Java-Advanced-Programming> cd "d:\Java-Advanced-Programming\" ; if (?) { javac Physics.java } ; if (?) { java Physics }
● Roll : 283
Name : Amit
Physics Marks : 80
○ PS D:\Java-Advanced-Programming> [ ]
```

