23CSE101

OBJECT ORIENTED PROBLEM SOLVING

LAB MANUAL



**Department of computer and communication Engineering Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

# Name: Astha Kiran

**Verified By: Roll No: AV.SC.U4CSE24010**

**Date of submission: Sem: 1st Year**

**Class: CSE/A**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Title | Date | Page No | Signature |
| WEEK 1 |  |  |  |  |
| 1 | How to download and install JAVA |  |  |  |
| 2 | Write a java program to print the message ‘Welcome to Java Programing’ |  |  |  |
| 3 | Write a Java Program to print name Roll and Section of the student |  |  |  |
| WEEK 2 |  |  |  |  |
| 1 | Write a java program to calculate area of rectangle |  |  |  |
| 2 | Write a java program to convert celsius into fahrenhiet to celsious and vice versa |  |  |  |
| 3 | Write a java program to calculate the simple interest |  |  |  |
| 4 | Write a java program to find the largest of the number using ternatary operatop |  |  |  |
| 5 | Write a java program to find the factorial of the number |  |  |  |
| WEEK 3 |  |  |  |  |
| 1 | Create a java program with the following instructions:   * Create a class with name Car. * Create attributes named Car\_color, Car\_Brand,Fuel\_type,mileage. * Create three methods named start(), stop(), service(). * Create three objects car1, car2, car3. * Create one constructor which should print “welcome to my garage”. |  |  |  |
| 2. | Write a JAVA Program to create a class named bank account with two methods deposit() and withdraw():   * In deposit()- whenever an amount is deposited, it has to be updated with the current amount. * Withdraw()- whenever an amount is being withdrawn it has to be less than the current balance otherwise print insufficient balance. |  |  |  |

# 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Title | Date | Page No | Signature |
| WEEK 4 |  |  |  |  |
| 1 | Write a JAVA Program with class named Book:   * The class should contain various attributes such as “title\_of\_book, Author, year\_of\_publication”. * It should also contain a constructor with parameters which initializes “title\_of\_book, Author, year\_of\_publication”. * Create a method which displays the details of the book “title\_of\_book, Author, year\_of\_publication”. * Display the details of the two books by creating two objects. |  |  |  |
| 2 | To create a JAVA program with class named Myclass:   * with “static variable-count” of int type, initialize to zero and a constant variable “pi-double” to initialize to 3.1415 as attributes of that class. * Now define a constructor for Myclass that increments the count variable each time object for Myclass is created. Finally print values of “count” and “pi” variables. |  |  |  |
| WEEK 5 |  |  |  |  |
| 1. | Create a calculator using the opertations including addition , subtraction multiplication and division using multilevel inheritence and desire output. |  |  |  |
| 2. | 1. A vehicle rental company wants to develop a system that maintains information about different types of vehicles available for rent out cars and bikes and they need a program to store details about each vehicle such as brand and speed.  * Cars should have an additional property/attributes no. of doors , sitting capacities. * Bikes should have a property indicating whether they have gears or not. * The system should also include a function to display details about each vehicle and indicate when a vehicle is starting * Each class should have a constructor   Which obj oriented programming concept is used in the above program? Explain why it is useful in this scenario. |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Title | Date | Page No | Signature |
|  | 1. If the company decides to add a new type of vehicle truck, how would you modify the above program:  * Truck should include an additional property-capacity(in tons). * Create a show truckdetails() to display the trucks capacity. * Write a constructor for the truck that initializes all properties.   Implement truck class and update main to create a truck object and also create an object for car and bike subclass.  Finally display its details |  |  |  |
| WEEK 6 |  |  |  |  |
| 1 | Write a java program to create a vehicle class with a method displayinfo(). Override this in the Car subclass to provide specific information about a car [ carCompany, carModel, carPrize, seatingCapacity,petrol\_or\_not(Boolean)] |  |  |  |
| 2 | A college is developing an automated admission system that veifies students eligibility for undergraduate(UG) and post-graduate(PG) programs. Each program has different eligibility criteria based on the students percentage in their previous qualification.   * UG qualification require : min 60% * PG qualification require : min 70% |  |  |  |
| 3. | Create a calculator class with overloaded methods to perform addition:   * Add 2 int * Add 3 int * Add 2 doubles |  |  |  |
| 4. | Create a shape class with a method calculateArea() that is overloaded for different shapes(e.g. square,rectangle) then, create a subclass circle that overrides the calculateArea() method for circle. |  |  |  |
| WEEK 7 |  |  |  |  |
| 1. | Write a JAVA program to create an abstract class animal with an abstract method called Sound(). Create subclasses Lion and Tiger and implement the Sound() method to make a specific sound for each animal. |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Title | Date | Page No | Signature |
| 2. | Write a JAVA program to create an abstract class shape3D with abstract method calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend the shape3D class and implement the respective methods to calculate the volume and surface area of each shape. |  |  |  |
| 3. | Write a JAVA program using an abstract class to define a method for pattern printing.   * Create an abstract class PatternPrinter with an abstract method printPattern(int rows) and a concrete method to display the pattern title. * Implement two subclasses: * StarPattern: prints a right angled triangle of stars(\*). * NumberPattern: prints a right angled triangle of increasing numbers.   In the main main method , create objects of both subclasses and print the pattern for given numbers of rows. |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# WEEK-1

# Program1)

# AIM – To download and install JAVA

# PROCEDURE

# 1)Search “Java download” in the search bar (e.g. Google)

# 2) Go to the website of Oracle

# 

# 3)Download the LTS(Long-Term Support) version of jdk.

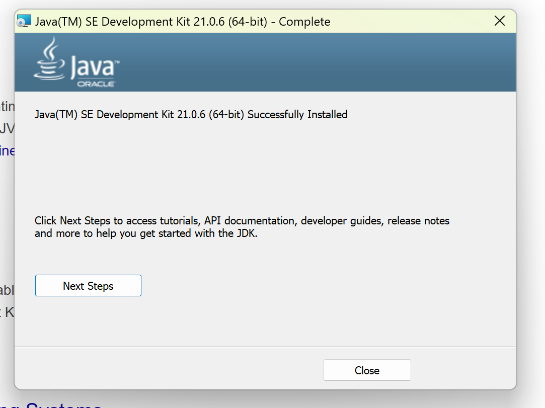
# Here it is “jdk21”

4) Select your operating System i.e. for me it is windows so I m selecting

windows option. Then select x64 installer

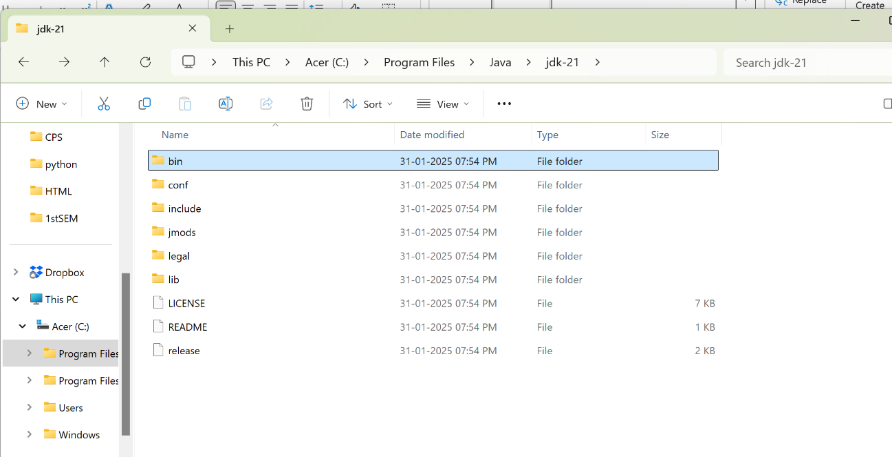
# 

5) Download and installation.



**6**) C Drive 🡪 Program files 🡪 Java 🡪 jdk21 🡪 libraries + modules 🡪bin

Now select and copy the path**.**

****

7) Press Windows + R, type sysdm.cpl, and click Ok.

8) The System Properties window will open.

9) Navigate to the Advanced tab.

10) Click on Environment Variables at the bottom.



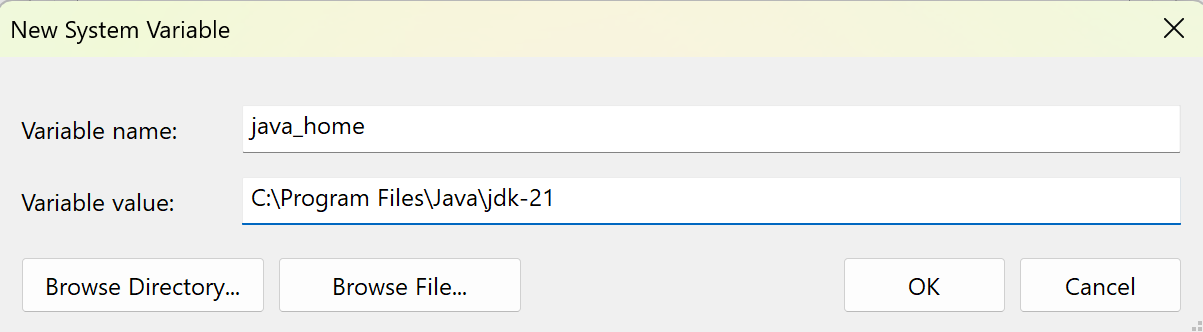
11) Under System Variables, click New.

12)Set the Variable name as Java\_home.

13) Set Variable value as C:\Program Files\Java\jdk-21

(or your installation path).

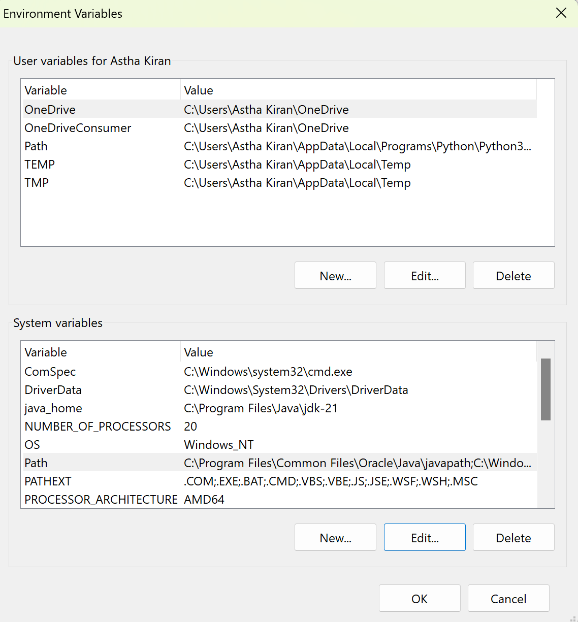
Click OK.



14) In System Variables, find Path and double click on it.

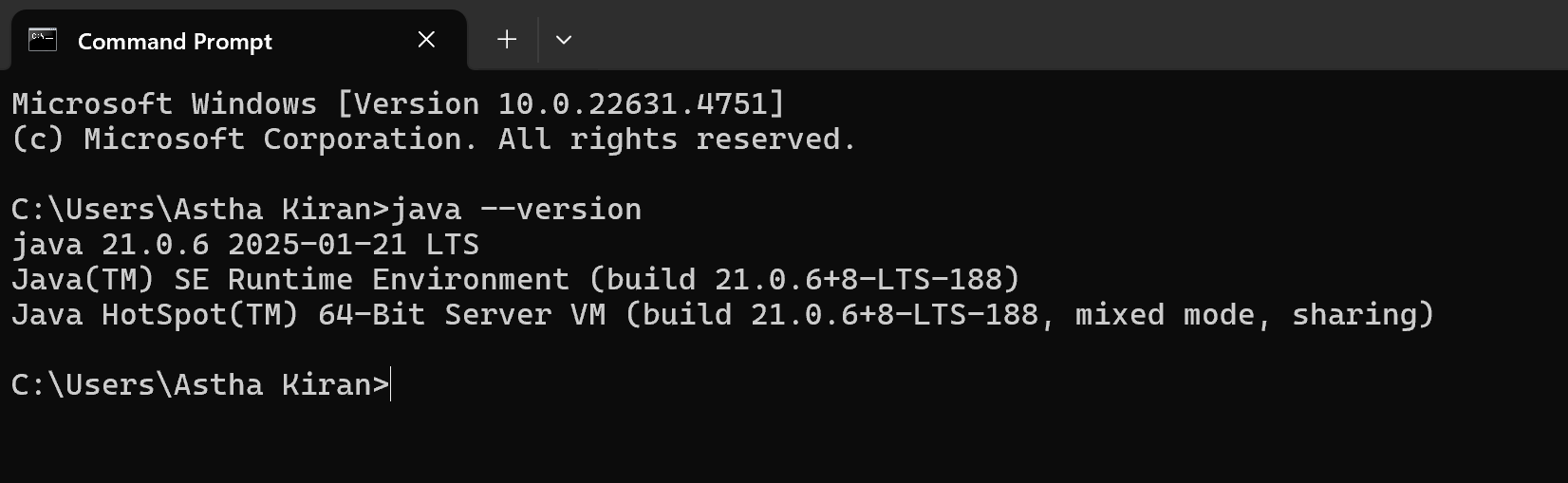
15)Click New and add: C:\Program Files\Java\jdk-21\bin

16)Click OK to save.



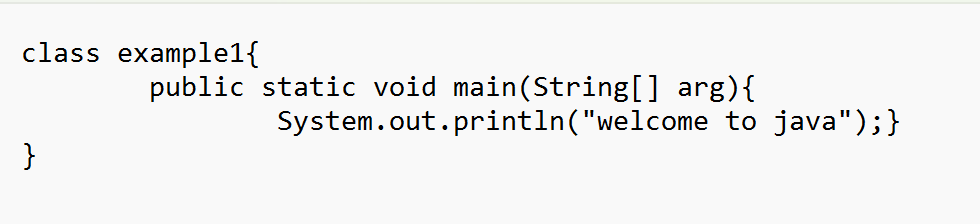
**Step 17:Verify Installation**

1. Open Command Prompt.
2. Type the following command: **java --version** and press Enter.



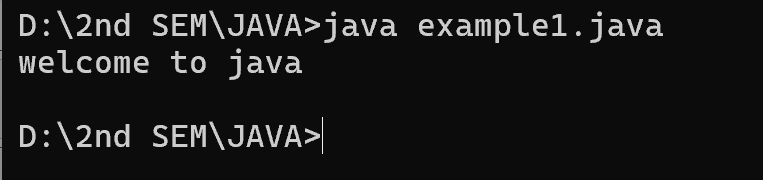
**Program 2)**

# AIM – To print the statement “Welcome to Java” using JAVA

****

# 

**Output**

****

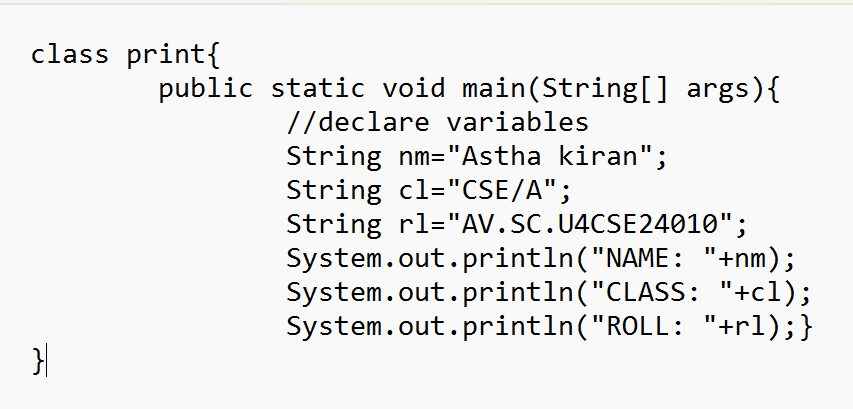
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected System.out.println("welcome to java")} | Adding ‘;’ at the end of the statement  System.out.println("welcome to java"); |
| 2. | error: cannot find symbol  public static void main(string[] arg){ | **String** symbol instead of string |

**Concepts to be known:**

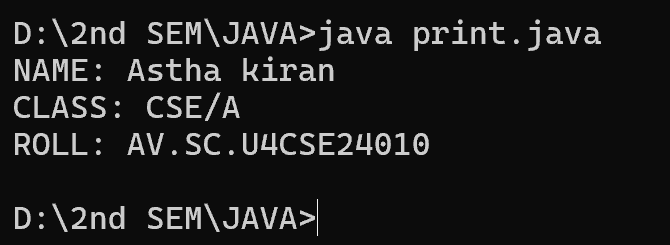
* System.out.println(" ")-to print the statement

**Program 3)**

**AIM-To print Name Roll and Section of the student using JAVA**

****

**Output**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: cannot find symbol  string cl="CSE/A"; | **String** symbol instead of string |

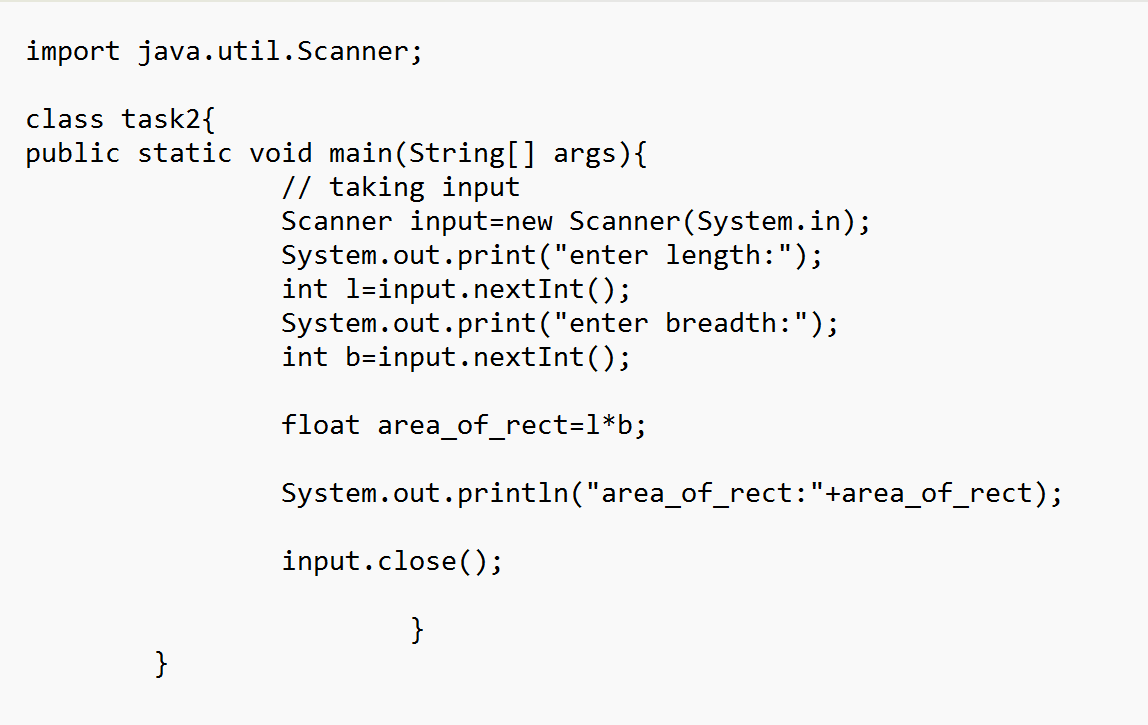
**Concepts to be known:**

* System.out.println(" ")-to print the statement
* String – to declare the data type as string
* // - used to write comments

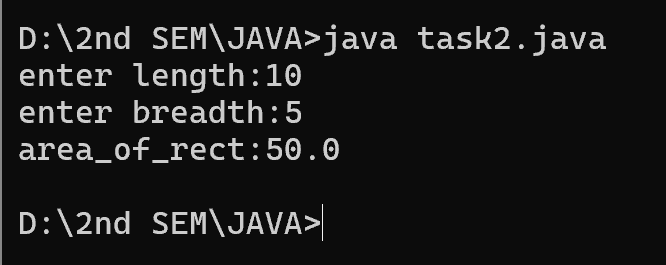
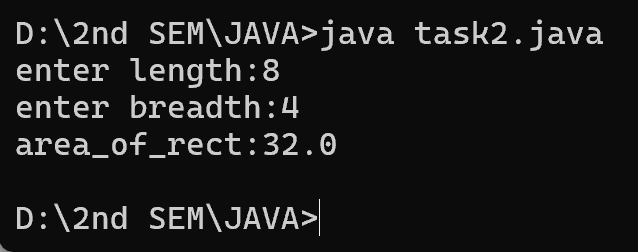
**WEEK-2**

**Program 1)**

**AIM- to calculate area of rectangle using JAVA**



**Output**

****

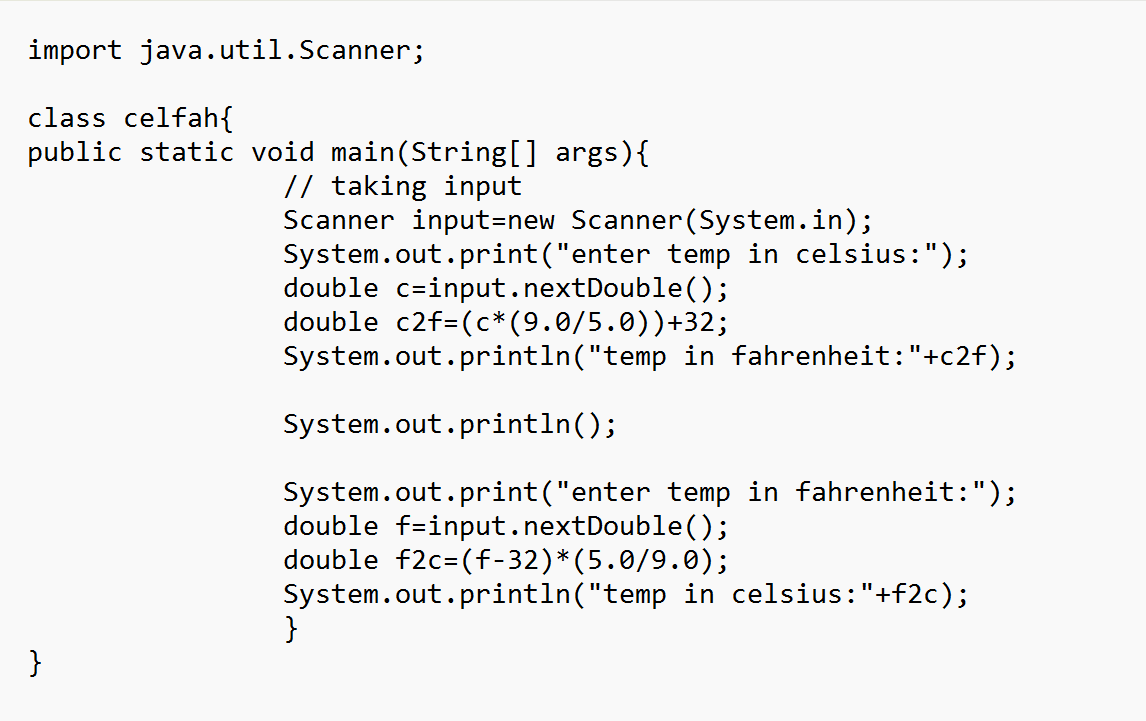
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: cannot find symbol  int b=input.nextint(); | Replace nextint() with nextInt() |

**Concepts to be known:**

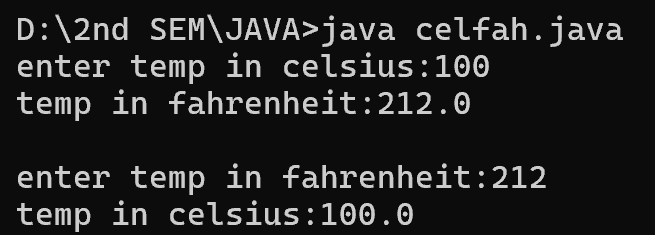
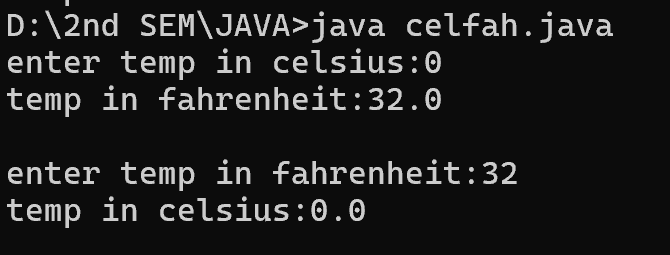
* **Import**- to import the
* System.out.println(" ")-to print the statement
* String – to declare the data type as string
* float – to declare the data types as float
* int – to declare the data types as integer
* // - used to write comments
* Scanner input=new Scanner(System.in):
* int <variable name>=input.nextInt():

**Program 2)**

**AIM- To convert Celsius to fahrenheit and vice versa**

****

**Output**

****

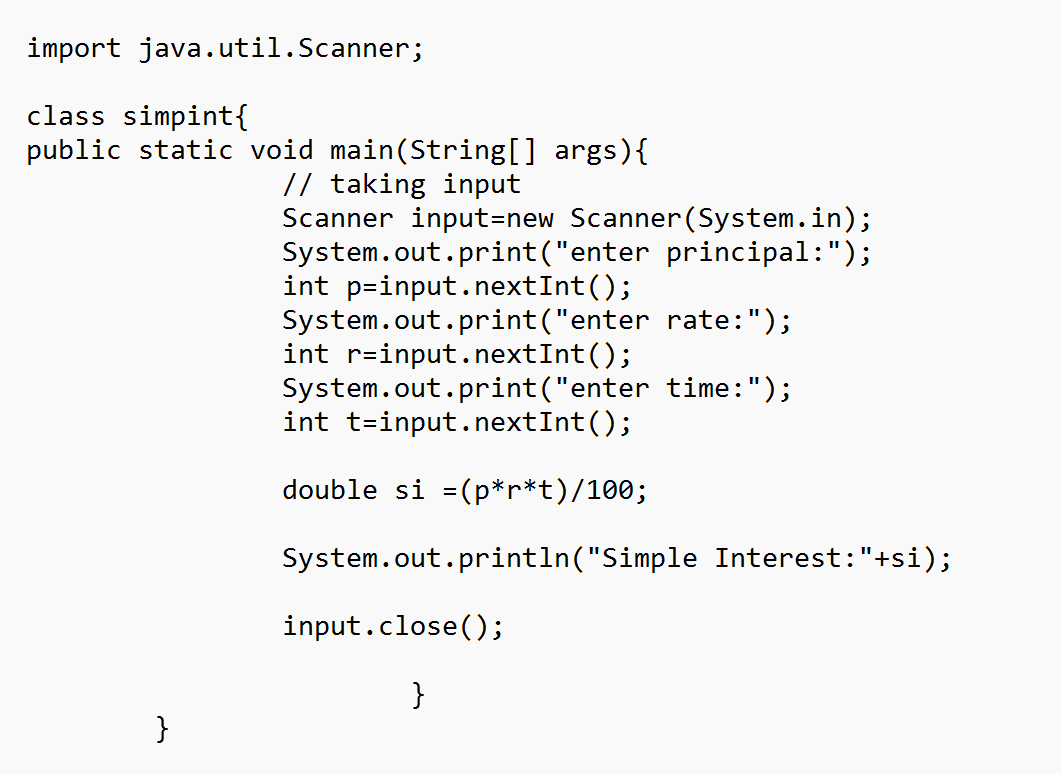
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: incompatible types: possible lossy conversion from double to float  float c2f=(c\*(9.0/5.0))+32; | Replace float with double |
| 2. | error: incompatible types: possible lossy conversion from double to float  float f2c=(f-32)\*(5.0/9.0); | Replace float with double |

**Concepts to be known:**

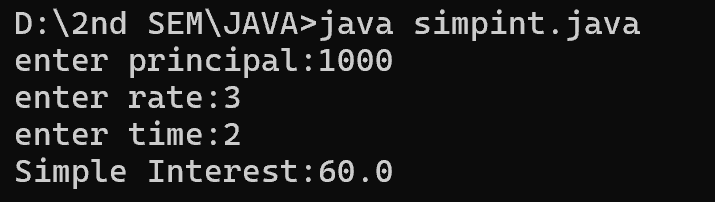
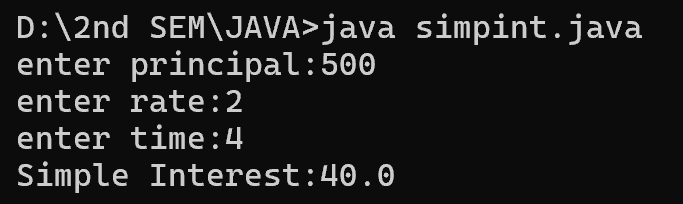
* + **Import** -to import scanner class
  + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + float – to declare the data types as float
  + int – to declare the data types as integer
  + // - used to write comments
  + Scanner input=new Scanner(System.in):
  + int <variable name>=input.nextInt():

**Program 3)**

**AIM- to calculate the simple interest using JAVA**



**Output**



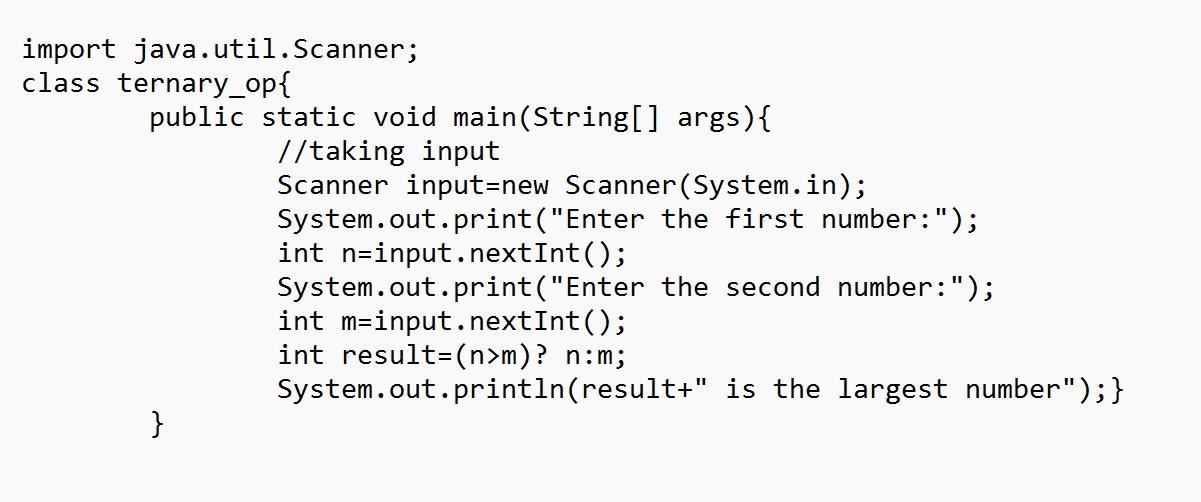
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: <identifier> expected  public Static void main(String[] args) | Replace Static with static |
| 2. | error: cannot find symbol  int r=input.nextint(); | Replace nextint() with nextInt() |

**Concepts to be known:**

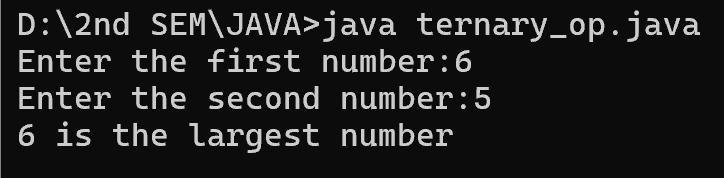
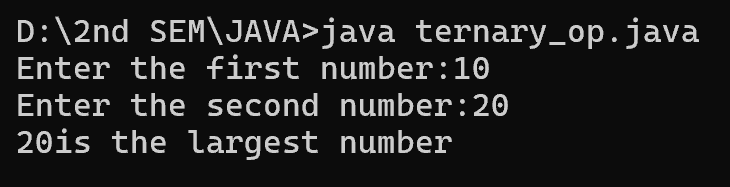
* + **Import** -to import scanner class
  + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + double – to declare the data types as double
  + int – to declare the data types as integer
  + // - used to write comments
  + Scanner input=new Scanner(System.in):
  + int <variable name>=input.nextInt():

**Program 4)**

**AIM- To find the largest of the number using ternatary operator**

****

**Output**



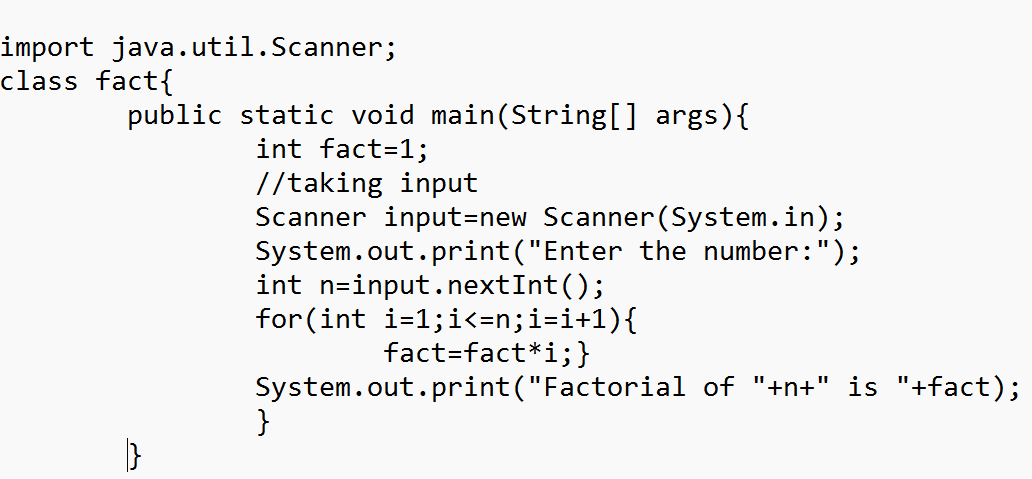
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: cannot find symbol  string result=(n>m)? n:m; | Change the data type of result to int |

**Concepts to be known:**

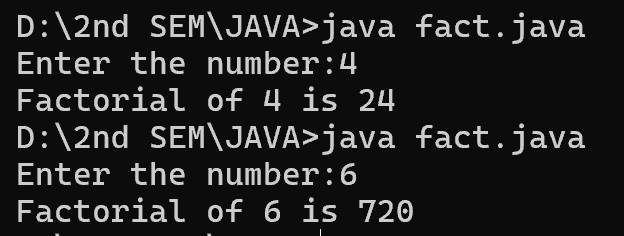
* + **Import** -to import scanner class
  + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments
  + Scanner input=new Scanner(System.in):
  + int <variable name>=input.nextInt():

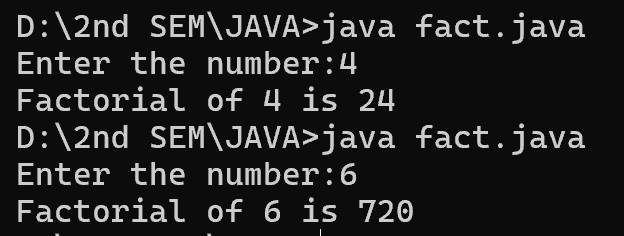
**Program 5)**

**AIM- To find the factorial of the number using JAVA**

****

**Output**

****

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: not a statement  for(int i=1;i<=n;i+i+1) | i=i+1 |
| 2. | error: ';' expected  fact=fact\*i | Adding ; at the end |

**Concepts to be known:**

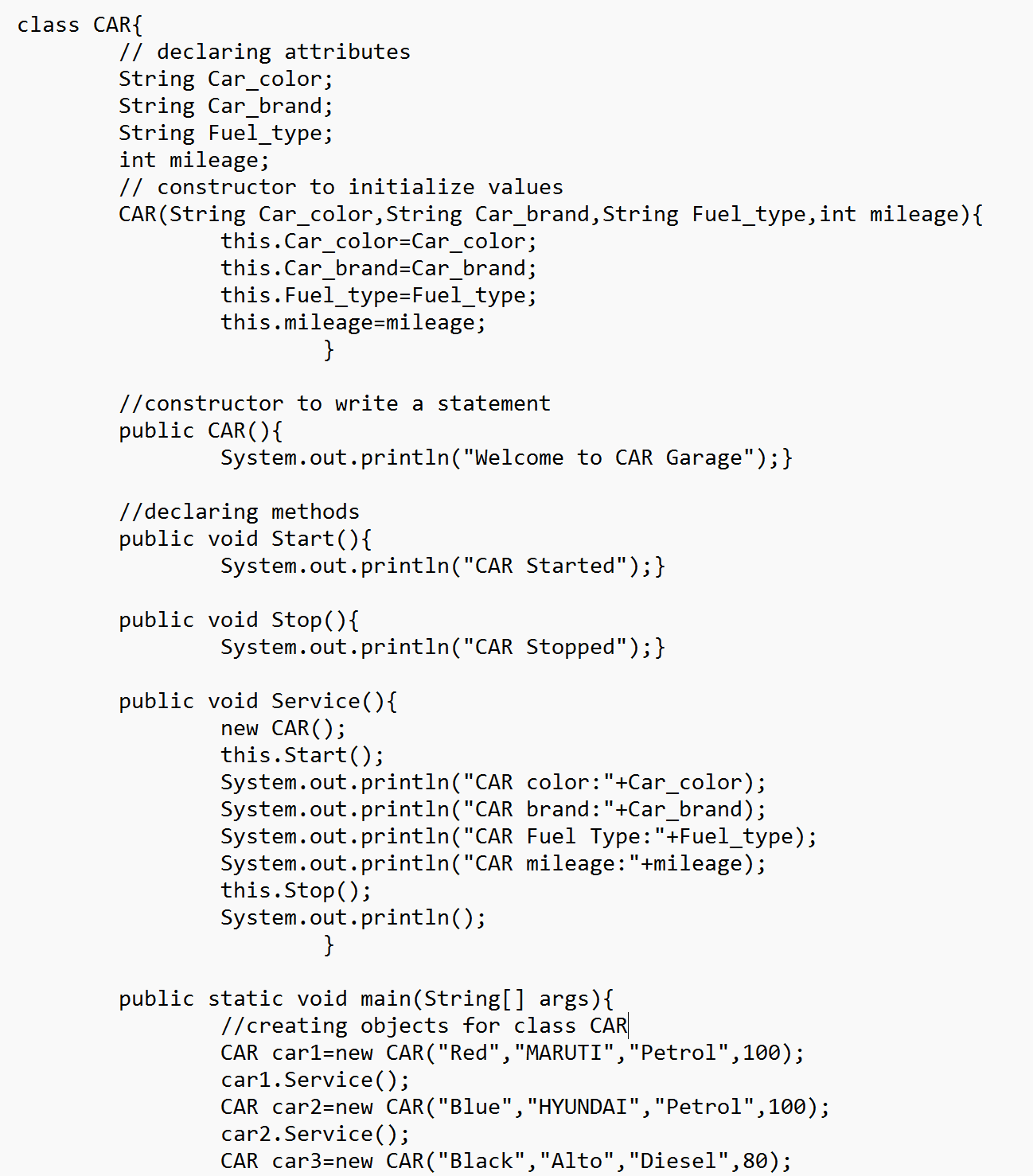
* + **Import** -to import scanner class
  + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments
  + Scanner input=new Scanner(System.in):
  + int <variable name>=input.nextInt():

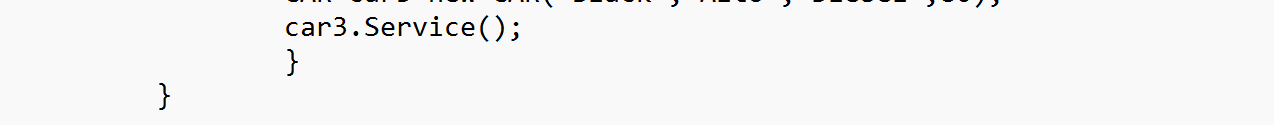
**WEEK-3**

**Program1)**

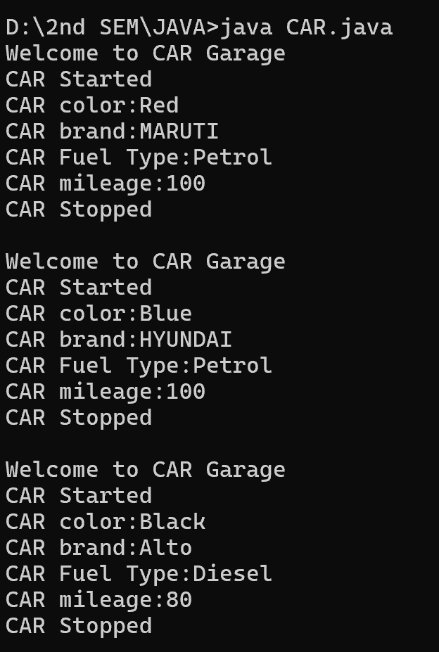
**AIM-** To create a java program with the following instructions:

1. a class with name Car.
2. attributes named Car\_color, Car\_Brand,Fuel\_type,mileage.
3. three methods named start(), stop(), service().
4. three objects car1, car2, car3.
5. one constructor which should print “welcome to CAR garage”.





**OUTPUT**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: cannot find symbol  this.stop(); | Stop() |
| 2. | error: not a statement  this.mileage; | this.mileage=mileage; |

**Concepts to be known:**

* + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments

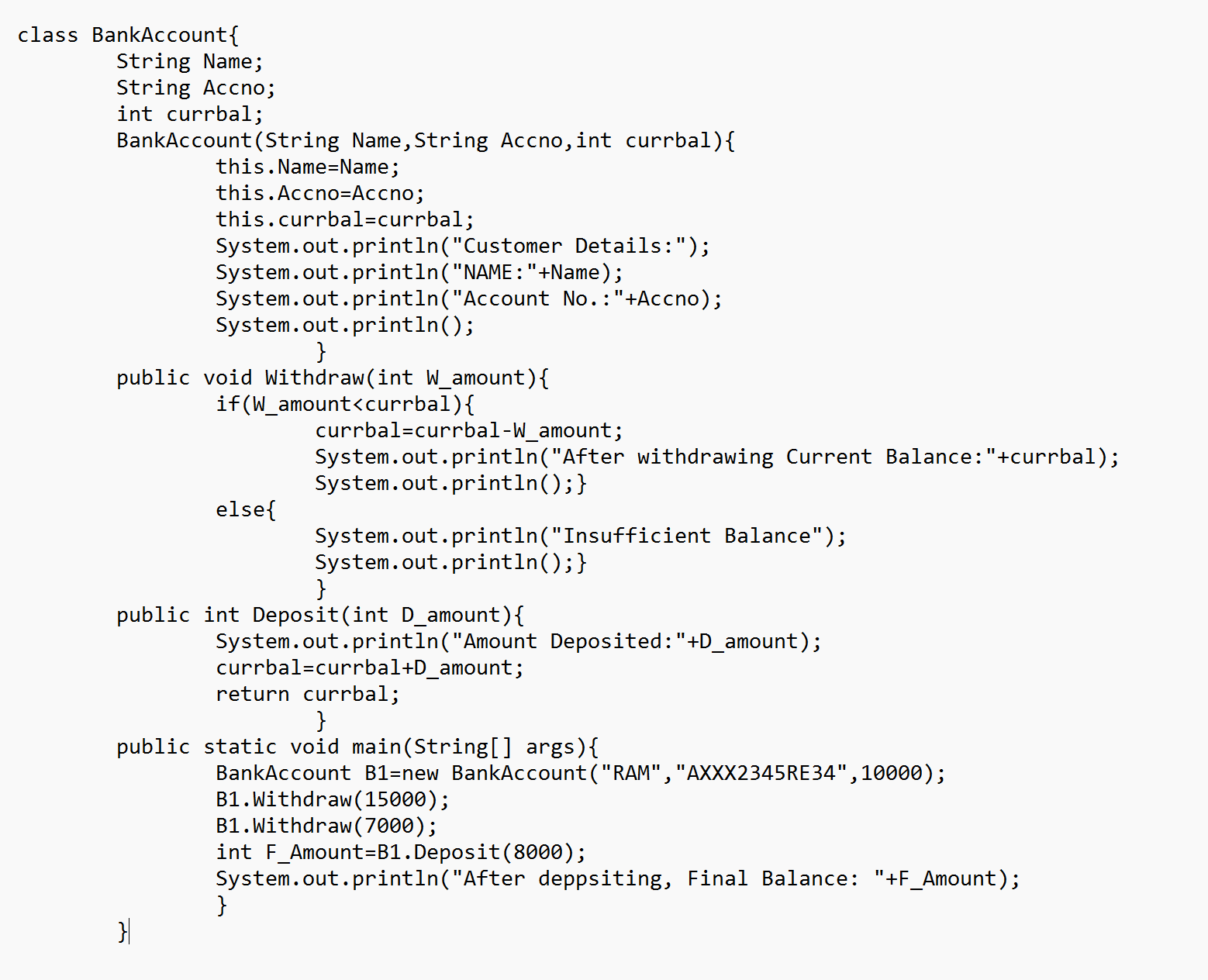
**Class Diagram**

|  |
| --- |
| CAR |
| Car\_color: String  Car\_brand: String  Fuel\_type: String  mileage: int |
| + CAR()  + CAR(String, String, String, int)  + Start(): void  + Stop(): void  + Service(): void  + main(String[]): void |

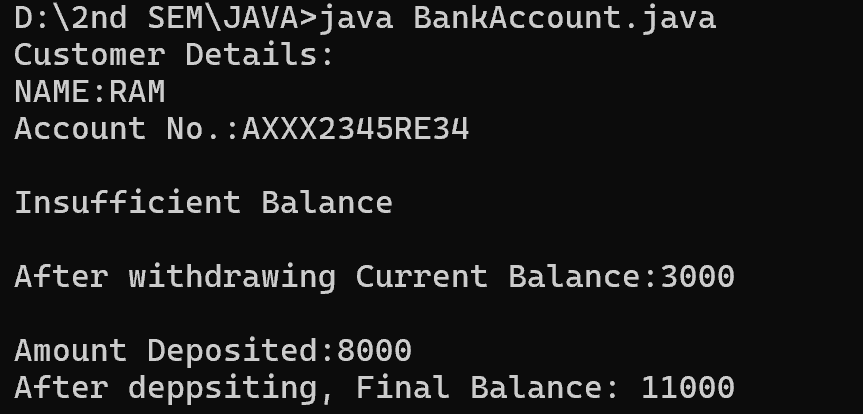
**Program2)**

**AIM**-To create a class named bank account with two methods deposit() and withdraw():

1. In deposit()- whenever an amount is deposited, it has to be updated with the current amount.
2. Withdraw()- whenever an amount is being withdrawn it has to be less than the current balance otherwise print insufficient balance.



**OUTPUT**



|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  currbal=currbal-W\_amount | Adding ; at the end |
| 2. | error: cannot find symbol  thiscurrbal=currbal; | this.currbal =currbal; |

**Concepts to be known:**

* + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments
  + this-

**Class Diagram**

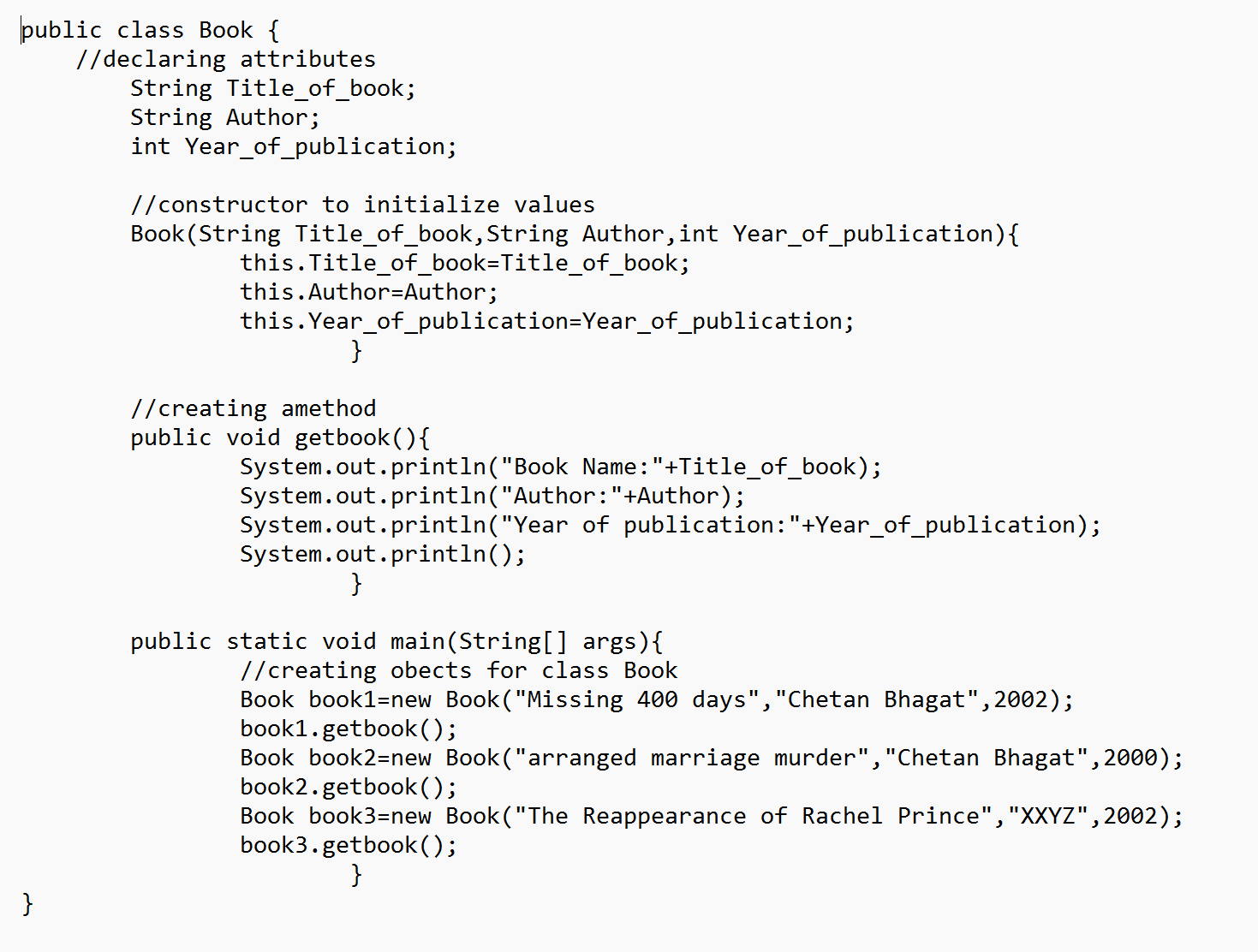
|  |
| --- |
| BankAccount |
| Name: String  Accno: String  currbal: int |
| + BankAccount(String, String, int)  + Withdraw(int): void  + Deposit(int): int  + main(String[]): void |

**WEEK-4**

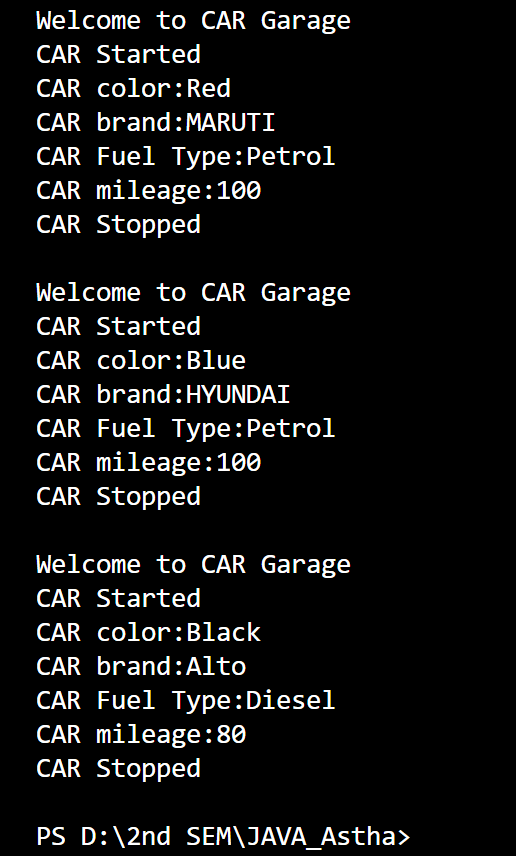
**Program1)**

**AIM-** To write a JAVA Program with class named Book:

1. The class should contain various attributes such as “title\_of\_book, Author, year\_of\_publication”.
2. It should also contain a constructor with parameters which initializes “title\_of\_book, Author, year\_of\_publication”.
3. Create a method which displays the details of the book “title\_of\_book, Author, year\_of\_publication”.
4. Display the details of the two books by creating two objects.



**OUTPUT**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  currbal=currbal-W\_amount | Adding ; at the end |
| 2. | error: cannot find symbol  thiscurrbal=currbal; | this.currbal =currbal; |

**Concepts to be known:**

* + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments
  + this-

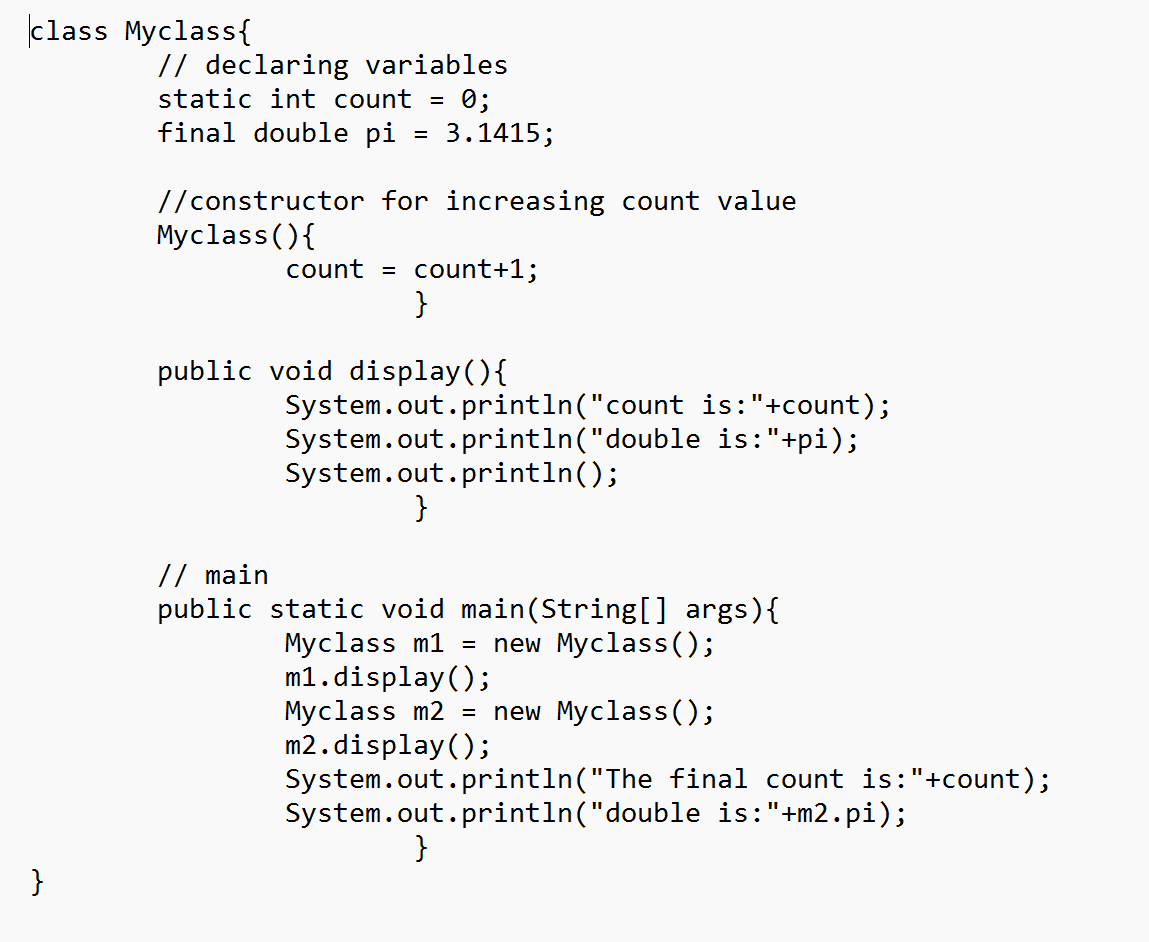
**Class Diagram**

|  |
| --- |
| Book |
| Title\_of\_book: String  Author: String  Year\_of\_publication: int |
| + Book(String, String, int)  + getbook(): void  + main(String[]): void |

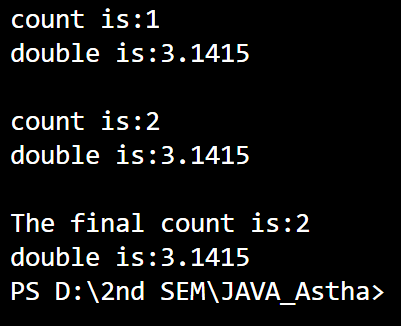
**Program2)**

**AIM-** To create a JAVA program with class named Myclass:

1. with “static variable-count” of int type, initialize to zero and a constant variable “pi-double” to initialize to 3.1415 as attributes of that class.
2. Now define a constructor for Myclass that increments the count variable each time object for Myclass is created. Finally print values of “count” and “pi” variables.

****

**OUTPUT**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  m1.display | Adding ; at the end |
| 2. | Syntax error, insert "}" to complete ClassBody | Adding } at the end |

**Concepts to be known:**

* + System.out.println(" ")-to print the statement
  + String – to declare the data type as string
  + int – to declare the data types as integer
  + // - used to write comments
  + this-

**Class Diagram**

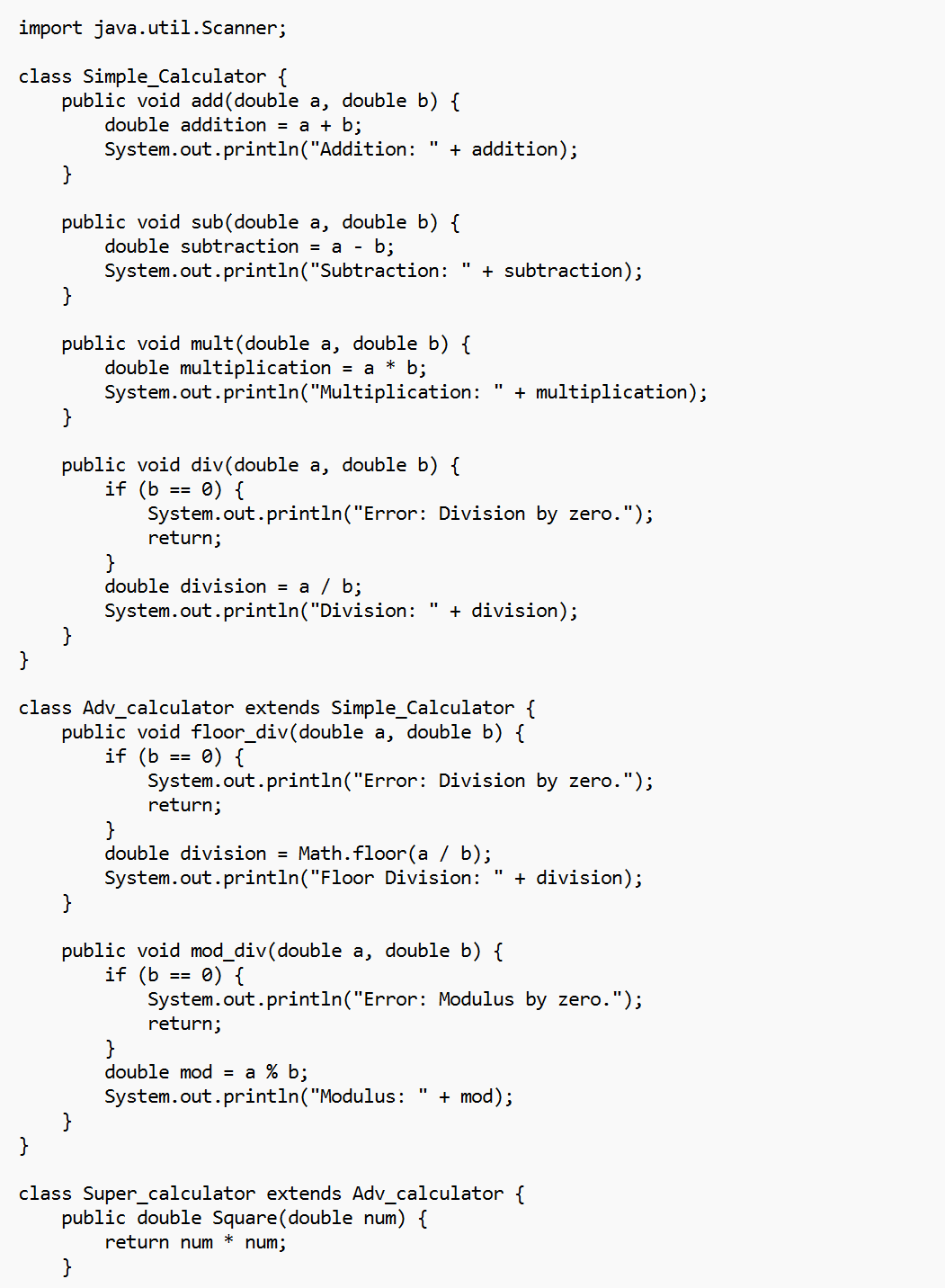
|  |
| --- |
| Myclass |
| count: static int  pi: final double |
| + Myclass()  + display(): void  + main(String[]): void |

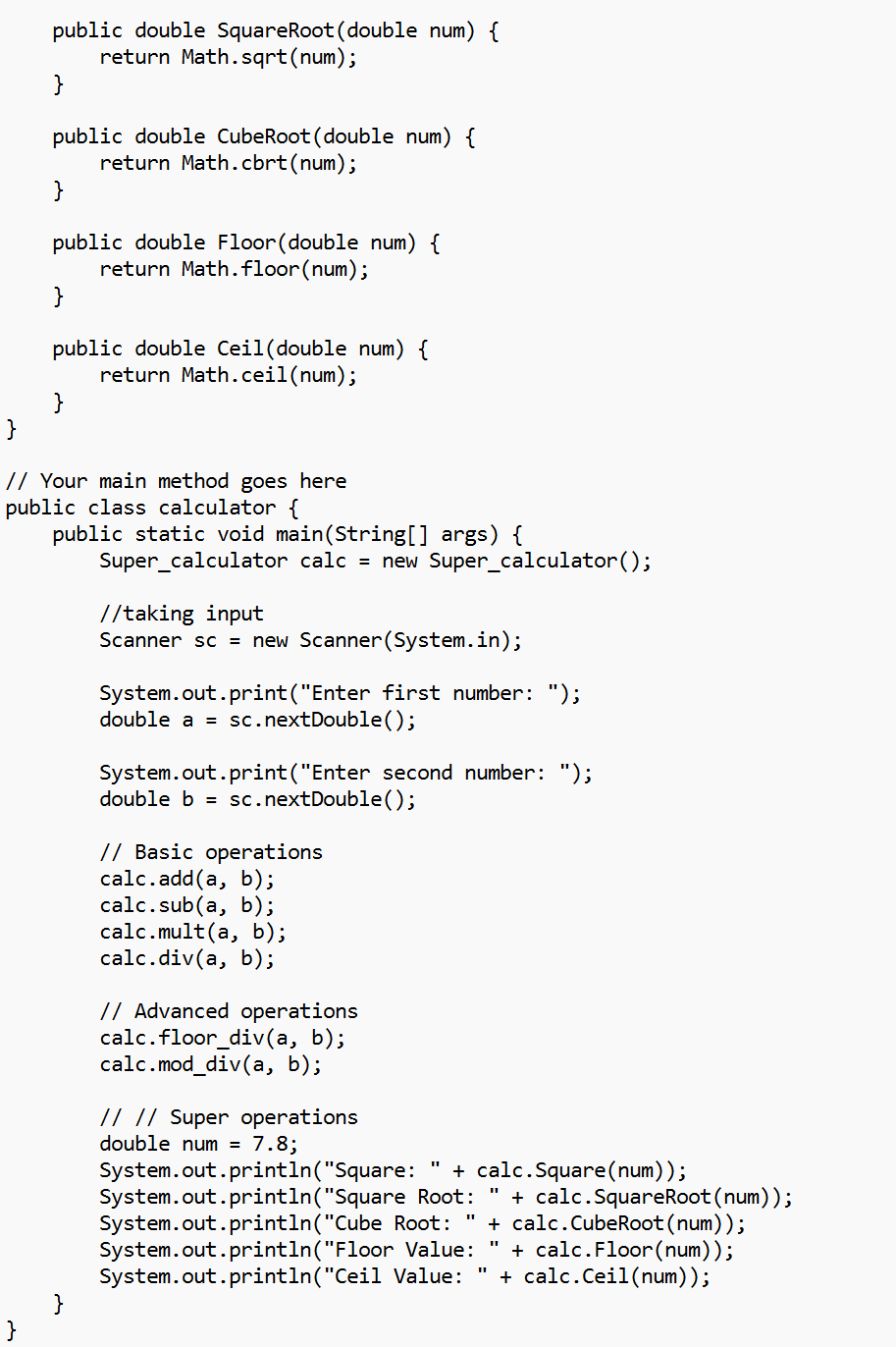
**WEEK-5**

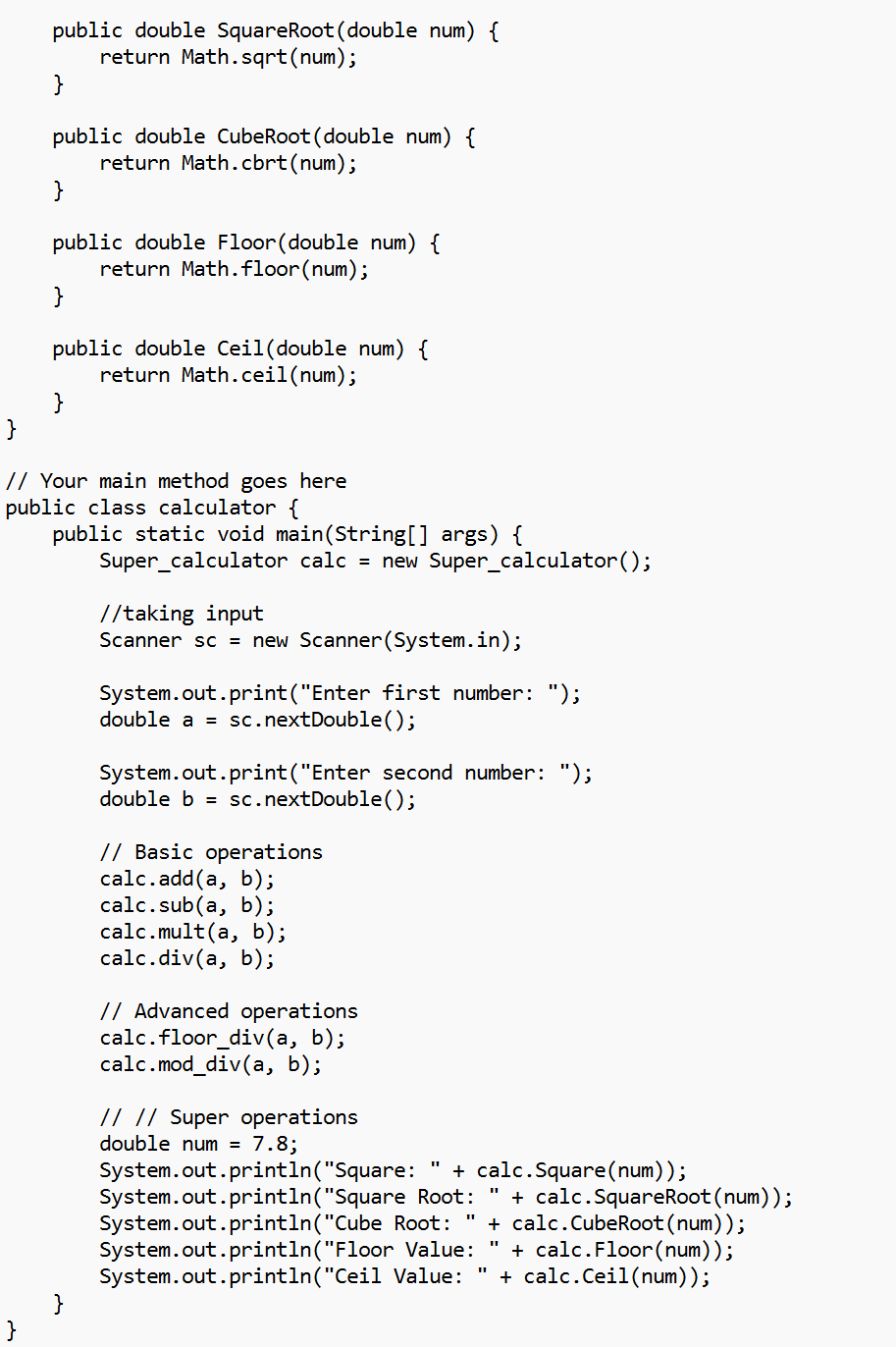
**Program1)**

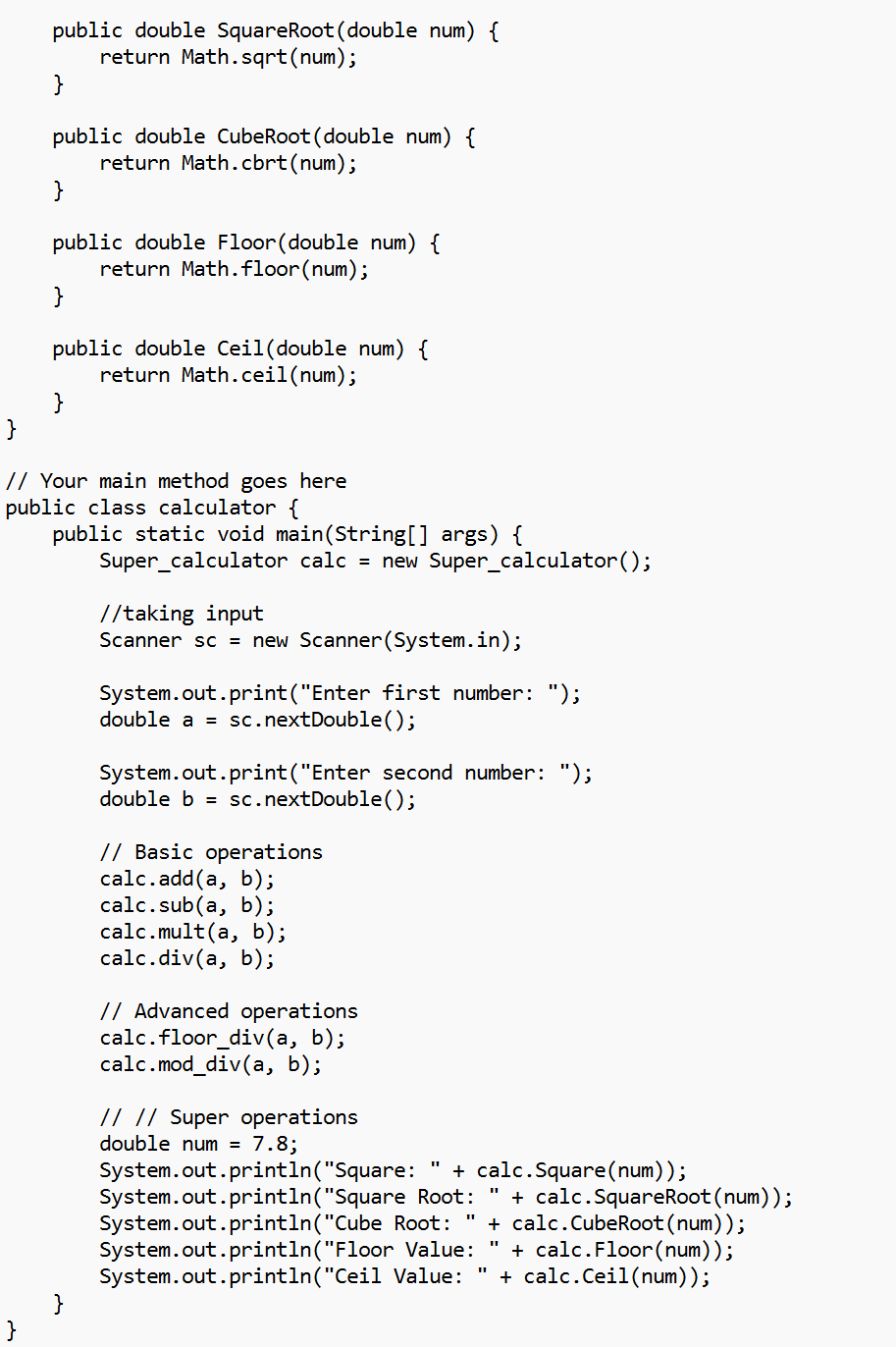
**AIM**- Create a calculator using the opertations including addition , subtraction multiplication and

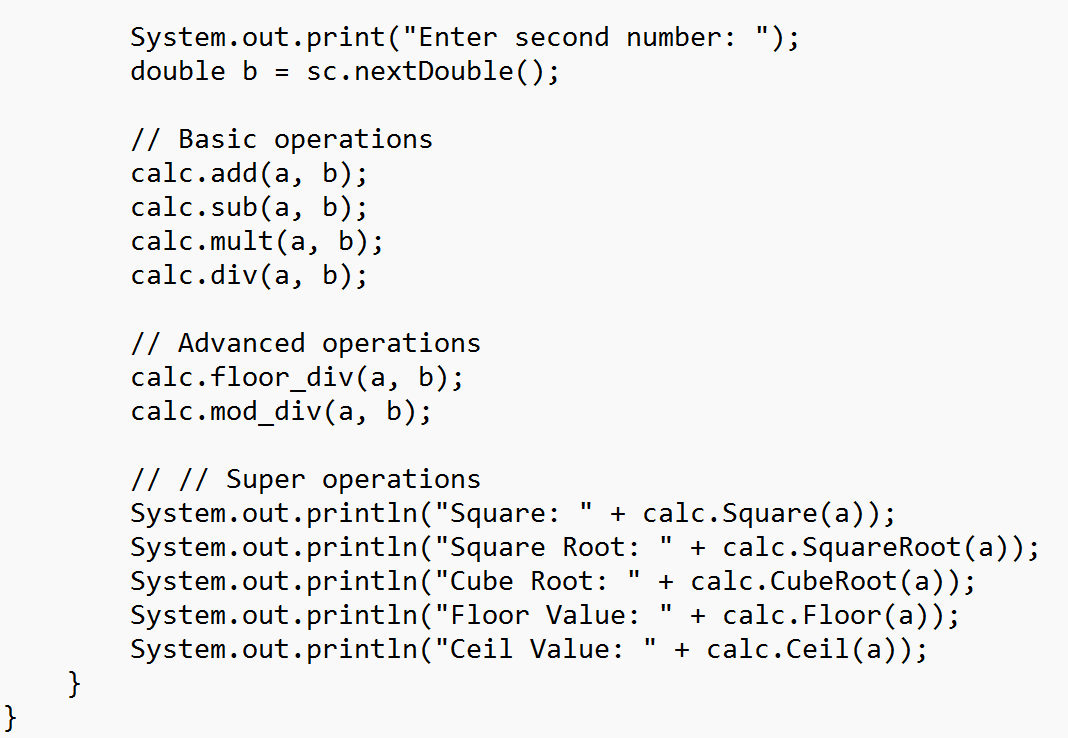
division using multilevel inheritence and desire output.



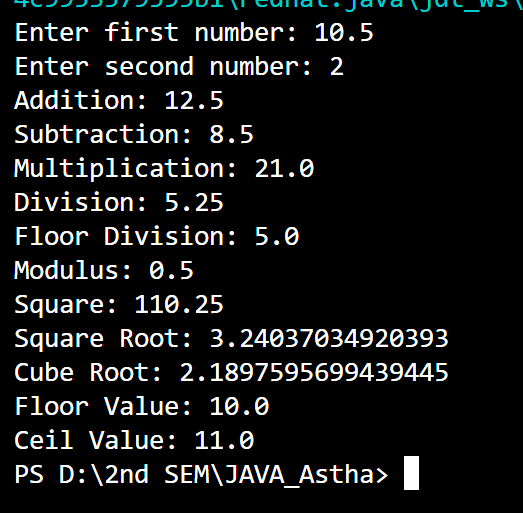








**Output**

****

**Concepts to be known: (Multi-level inheritance) :**

* public int addition(int n1, int n2){ - Method name is addition, whose accessibility is public. It takes it’s parameters in integer data type and returns an integer data type.
* class advanced\_calculator extends simple\_calculator{ - Class named advanced\_calculator inherits it’s some of the properties from parent class simple\_calculator
* return Double.NaN; - Nan stands for Not-a-Number. This is returned when mathematical operation results in an undefined value.It is a part of double class and is used in floating point calculation.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | Return type for the method is missing | Added the return data type as double : public double Floor(double num) |
|  |  |  |

**Class Diagram**

|  |
| --- |
| **Simple**\_**Calculator** |
| + add (double a,double b): void  + sub(double a,double b): void  + mult(double a,double b): void  +div(double a,double b):void |

|  |
| --- |
| **Adv\_Calculator** |
| +floor\_div( double a,double b):void  +mod\_div( double a,double b):void |

|  |
| --- |
| **Super\_calculator** |
| + Square(double num): double  + SquareRoot(double num): double  + CubeRoot(double num): double  + Floor(double num): double  + Ceil(double num): double |

**Program2)**

**AIM**- a) A vehicle rental company wants to develop a system that maintains information about different

types of vehicles available for rent out cars and bikes and they need a program to store details

about each vehicle such as brand and speed.

* Cars should have an additional property/attributes no. of doors , sitting capacities.
* Bikes should have a property indicating whether they have gears or not.
* The system should also include a function to display details about each vehicle and indicate when a vehicle is starting
* Each class should have a constructor

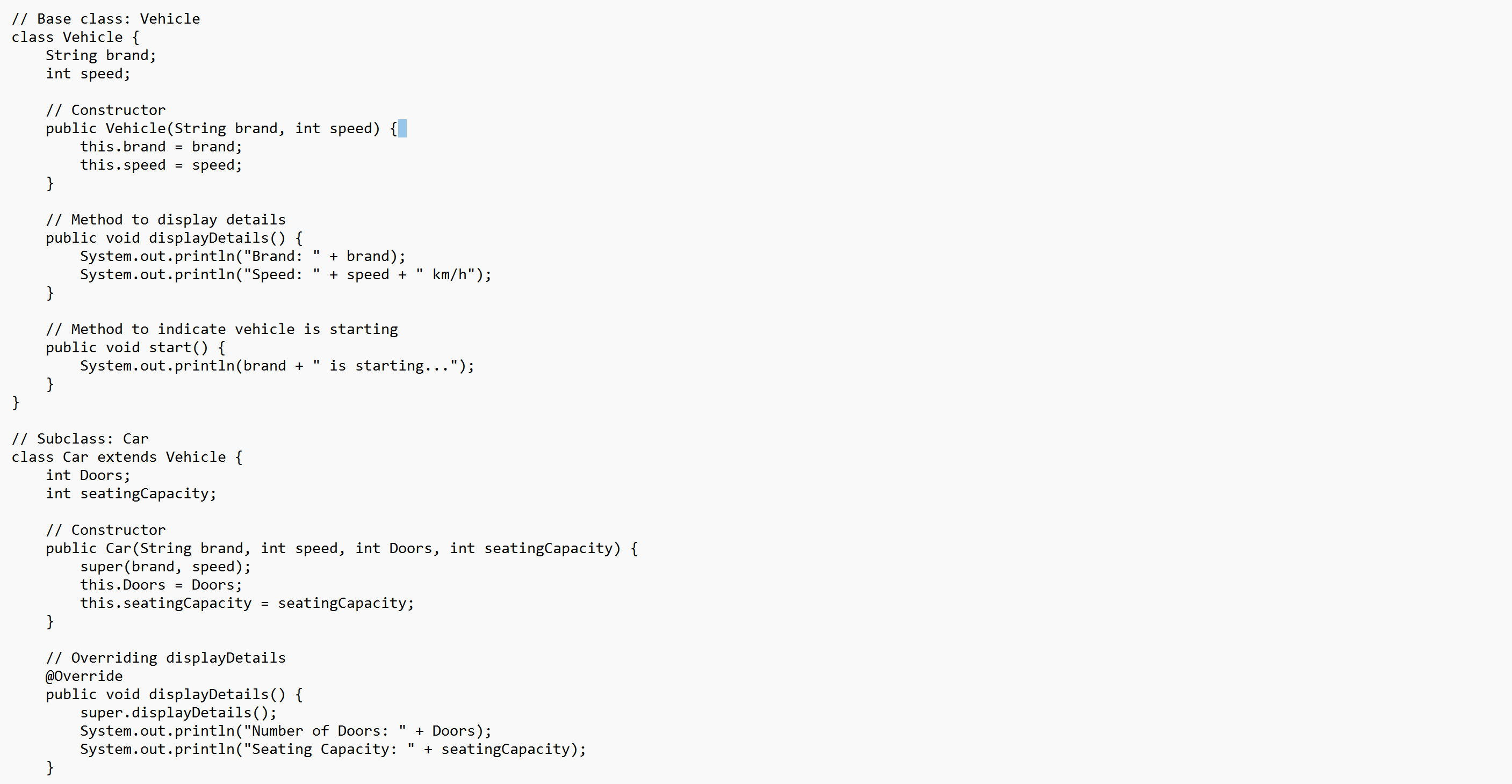
Which obj oriented programming concept is used in the above program? Explain why it

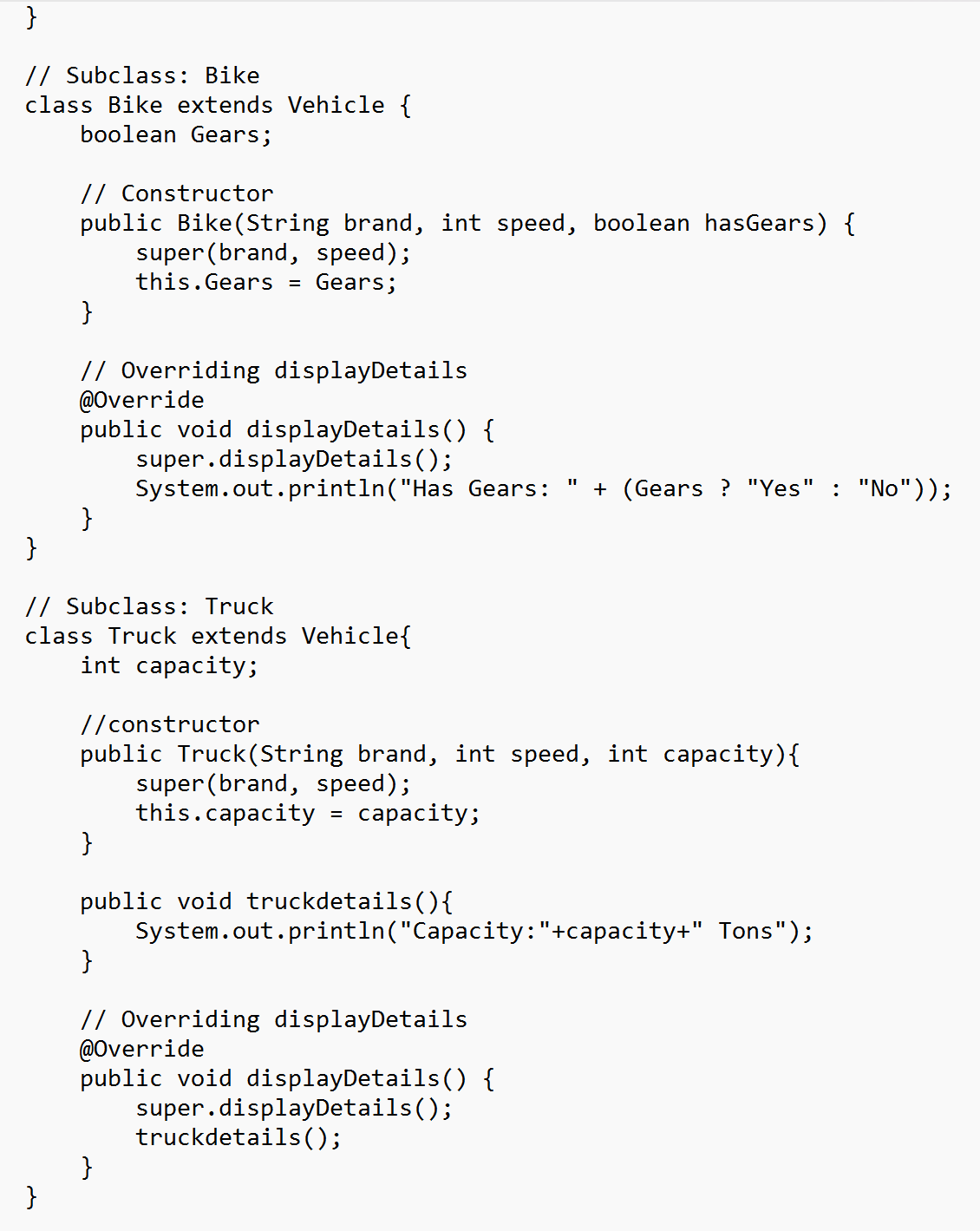
Implement truck class and update main to create a truck object and also create an object for car and

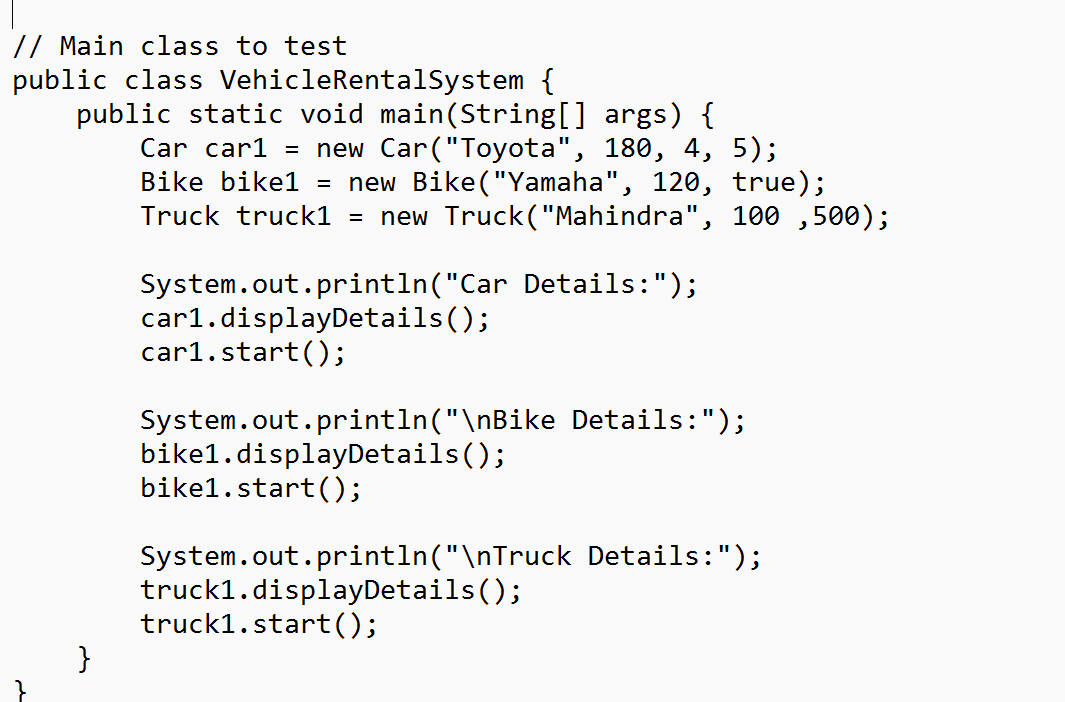
bike subclass. Finally display its details

1. If the company decides to add a new type of vehicle truck, how would you modify the above program:

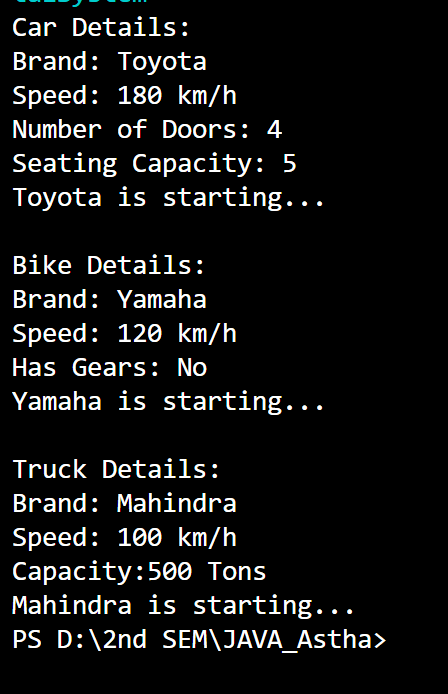
* Truck should include an additional property-capacity(in tons).
* Create a show truckdetails() to display the trucks capacity.
* Write a constructor for the truck that initializes all properties.







**Output**



|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | Syntax error on token "if", ( expected after this token | Changed if percentage>=70 To if (percentage>=70) |
|  |  |  |

**Concepts to be known: (Hierarchial Inheritance) :**

super(brand,speed); - This line of code, corresponds to calling the constructor of a super class, which requires parameters such as brand and speed.

**Class Diagram**

|  |
| --- |
| **Vehicle** |
| String brand  int speed |
| +Vehicle(String brand, int speed)  +displayDetails():void  +start():void |

|  |
| --- |
| **Truck** |
| +capacity:int |
| +Truck(brand:String, speed:int,capacity:double) :void +showTruckdetails( brand:String, speed:int,capacity:double) :void |

|  |
| --- |
| **Car** |
| int Doors  int seatingCapacity |
| +Car(String brand, int speed, int Doors, int seatingCapacity)  +displayDetails():void |

|  |
| --- |
| **Bike** |
| **+**gear**:**String |
| +Bike(brand:String, speed:int,gear:String) :void +displayBike(brand:String, speed:int,gear:String):void |

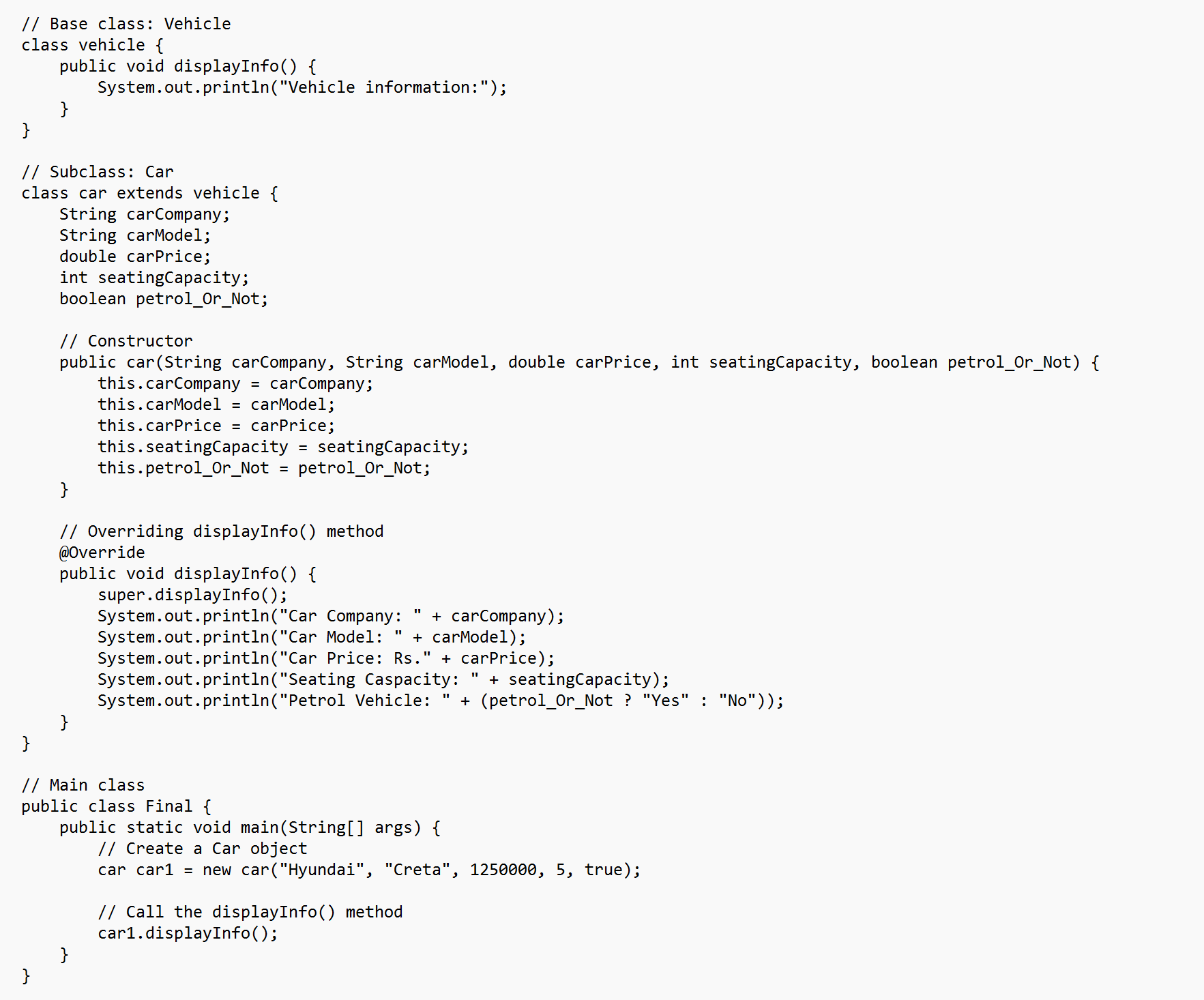
**WEEK-6**

**Program1)**

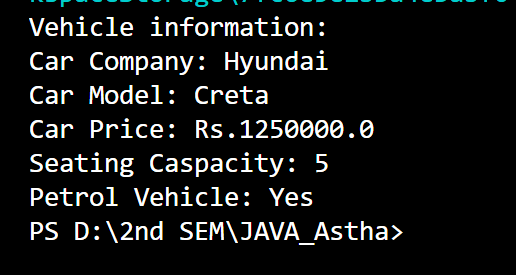
**AIM**- Write a java program to create a vehicle class with a method displayinfo(). Override this in

the Car subclass to provide specific information about a car [ carCompany, carModel,

carPrize, seatingCapacity,petrol\_or\_not(Boolean)]



**Output**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  m1.display | Adding ; at the end |
| 2. | Syntax error, insert "}" to complete ClassBody | Adding } at the end |

**Concepts to be known**:

* class car extends vehicle{} – Single-level Inheritance, where vehicle is the parent class and car is the subclass or child class.
* Overriding – The method displayInfo() is defined in both the parent class and the subclass. Since we create an object of the subclass, the method in the subclass is given priority over the one in the parent class when called, i.e. The displayInfo() method is overridden in the subclass.

**Class Diagram**

|  |
| --- |
| **vehicle** |
| + displayInfo(): void |

|  |
| --- |
| **car** |
| carCompany: String  carModel: String  carPrize: double  seatingCapacity: int  Petrol\_Or\_Not: Boolean |
| + Car(String car\_company,String car\_model,double car\_prize,int seating\_capacity,boolean diesel) :void  + displayInfo() : void |

**Program2)**

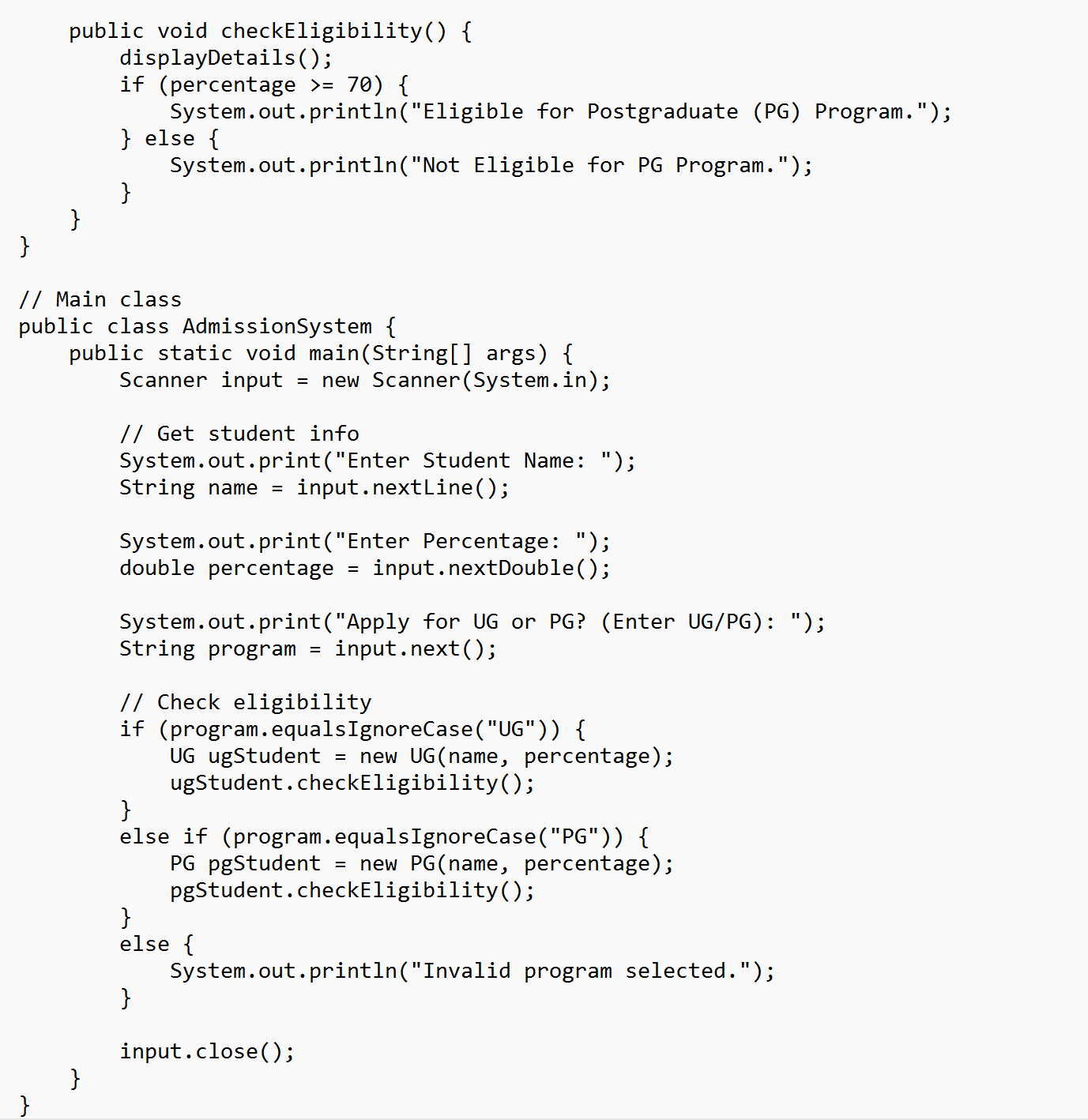
**AIM**- A college is developing an automated admission system that veifies students eligibility for

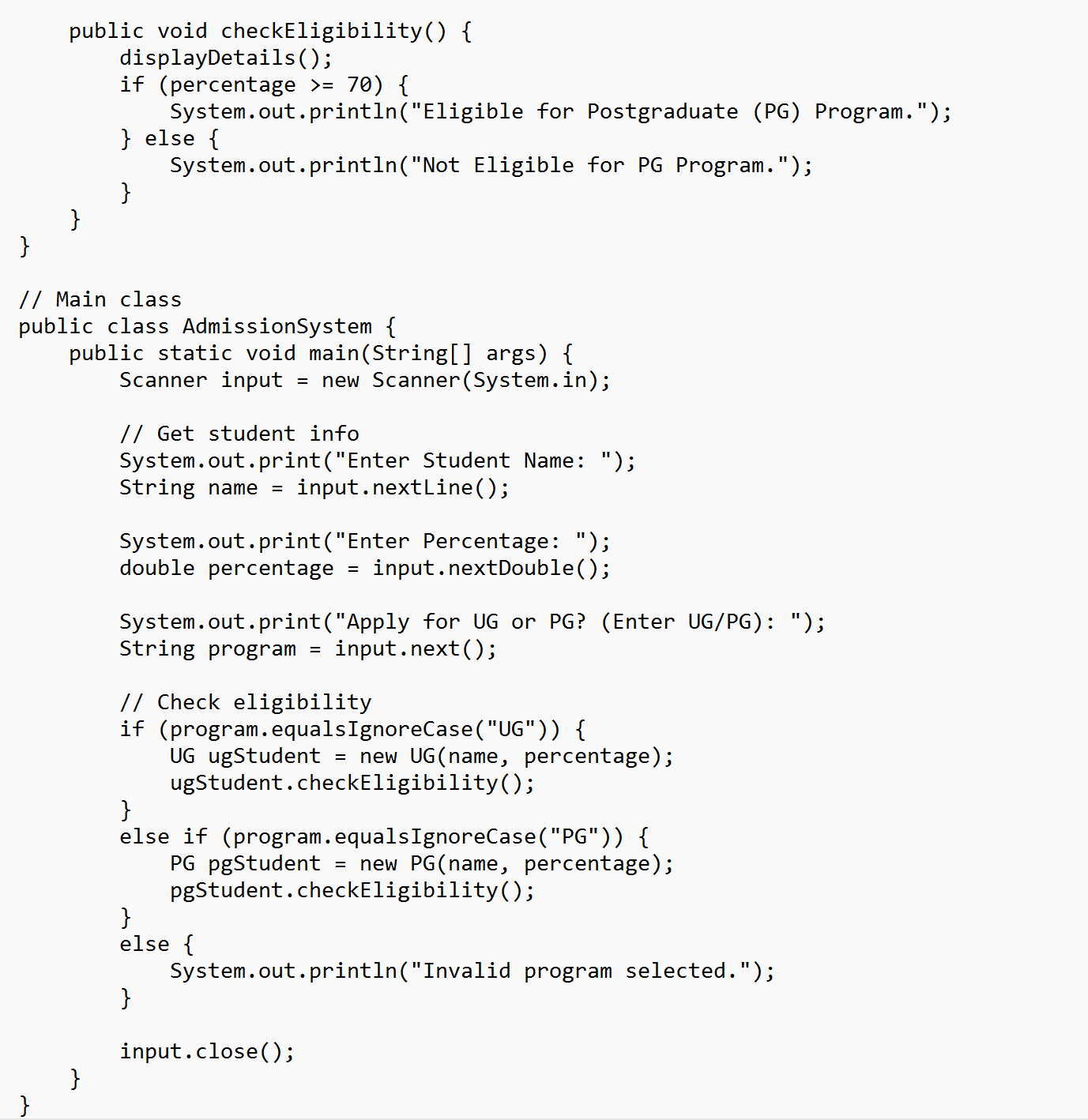
undergraduate(UG) and post-graduate(PG) programs. Each program has different eligibility criteria

based on the students percentage in their previous qualification.

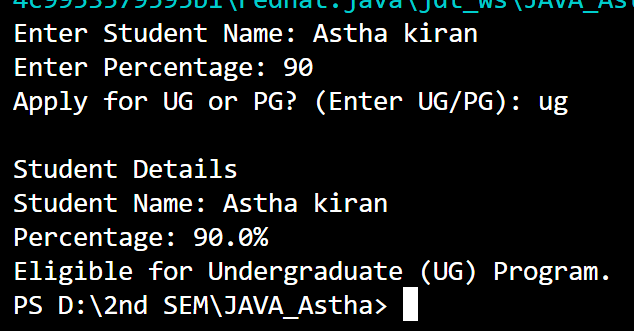
* UG qualification require : min 60%
* PG qualification require : min 70%

****

****

****

**Output**

****

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  String name: input.next(); | Adding  String name: input.nextLine(); |
| 2. | Syntax error, insert "}" to complete ClassBody | Adding } at the end |

**Concepts to be known**:

* Overriding – The method getEligibilty() is defined in both the parent class and the subclass. Since we create an object of the subclass, the method in the subclass is given priority over the one in the parent class when called, i.e. The getEligibilty() method is overridden in the subclass.

**Class Diagram**

|  |
| --- |
| **Student** |
| name:String  percentage: double |
| + Student(String name,String email, double percentage): void  + getEligibilty(): void |

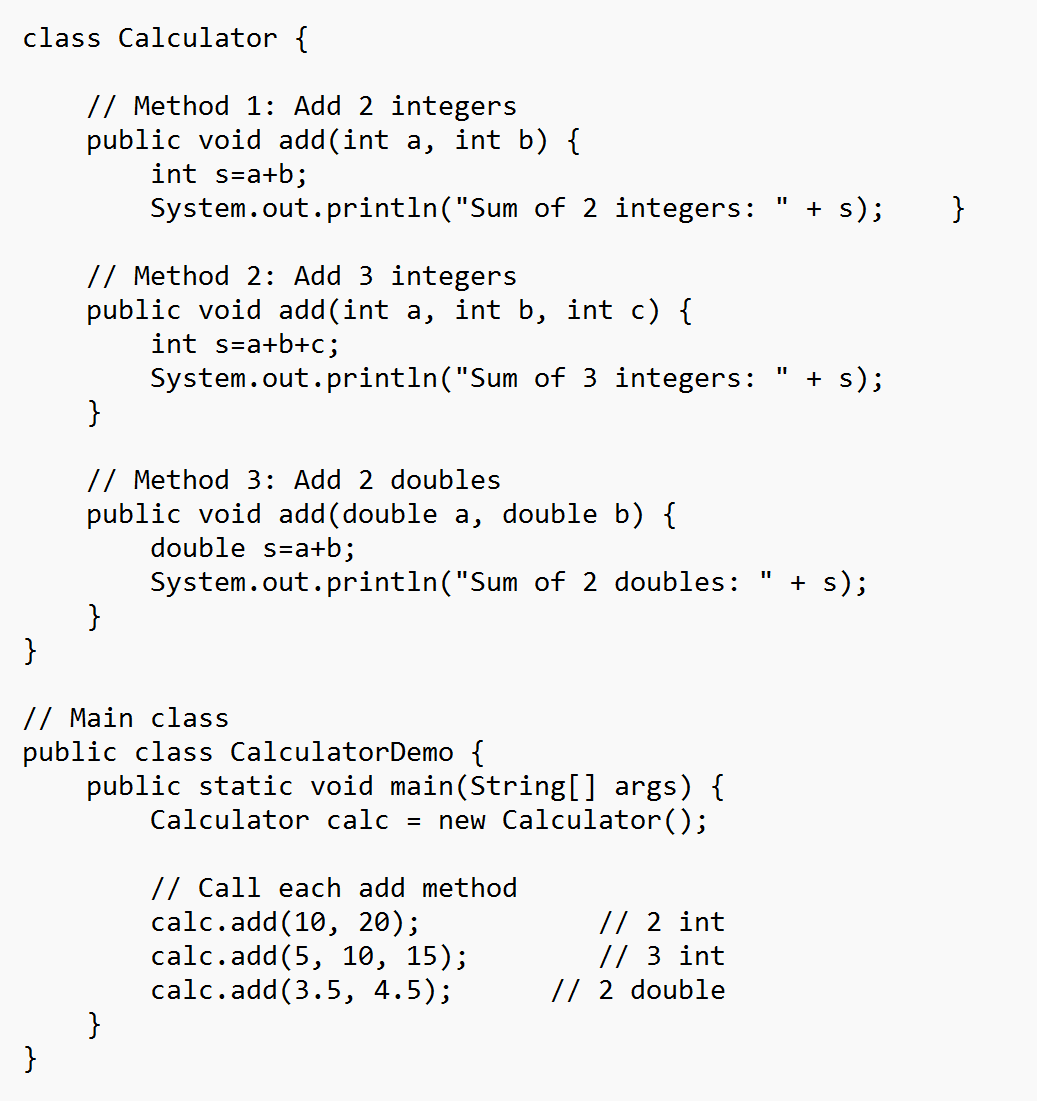
|  |
| --- |
| **UG** |
| + UG(String name, double percentage):void  + getEligibilty():void |

|  |
| --- |
| **PG** |
| + PG(String name, double percentage):void  + getEligibilty():void |

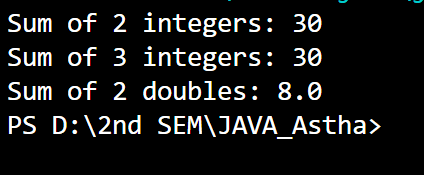
**Program3)**

**AIM-** Create a calculator class with overloaded methods to perform addition:

* Add 2 int
* Add 3 int
* Add 2 doubles



**Output**

****

**Concepts to be known:**

* Overloading- Defining multiple methods with the same name but with different parameters in the same class. Here, there are Multiple add() methods present, but all with different parameters. Depending upon the parameters passed, method is called**.**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | error: ';' expected  String name: input.next(); | Adding  String name: input.nextLine(); |
| 2. | Syntax error, insert "}" to complete ClassBody | Adding } at the end |

**Class Diagram**

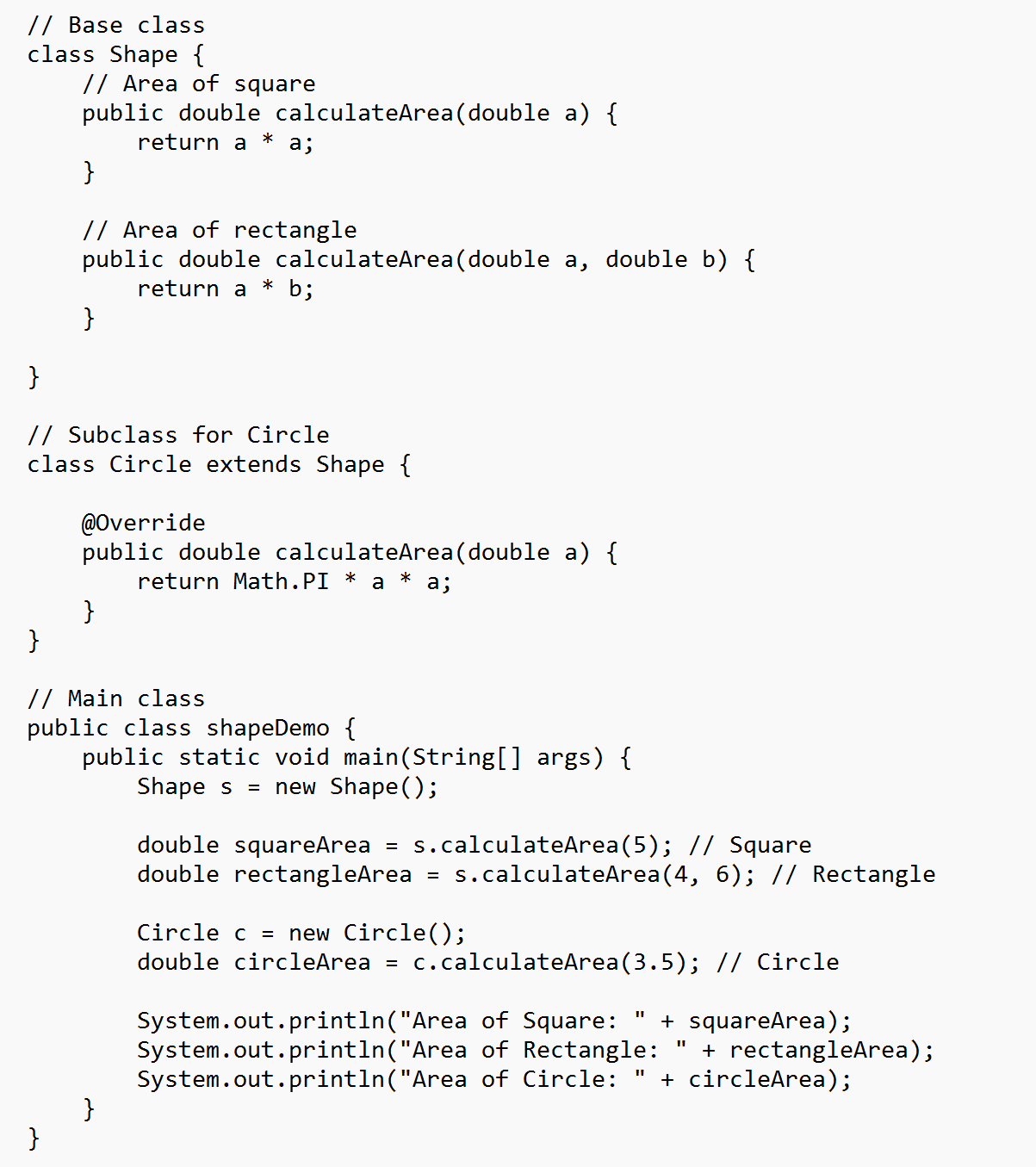
|  |
| --- |
| **Calculator** |
| + add(a: int, b: int): int  + add(a: double, b: double): double + add(a: int, b: int, c: int): int |

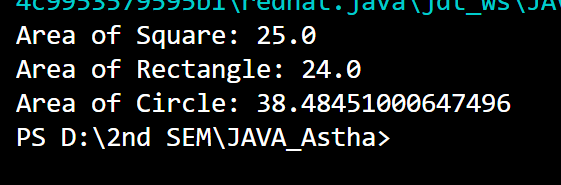
**Program4)**

**AIM-** Create a shape class with a method calculateArea() that is overloaded for different shapes

(e.g. square,rectangle) then, create a subclass circle that overrides the calculateArea() method

for circle.

****

**Output**

**Concepts to be known:**

* The above code explains method Overriding and method Overloading. Method calculatearea() which returns a double data type has been given different kinds of parameters. As per the condition, method executed. On the other hand, calculatearea() described in the Circle class serves as Overriding.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Errors** | **Rectification** |
| 1. | Syntax error, insert "}" to complete ClassBody | Adding } at the end |

**Class diagram**

|  |
| --- |
| **Shape** |
| + calculatearea(a: double): double  + calculatearea(a: double, b: double): double |

|  |
| --- |
| **Circle** |
| + calculatearea(a: double): double |

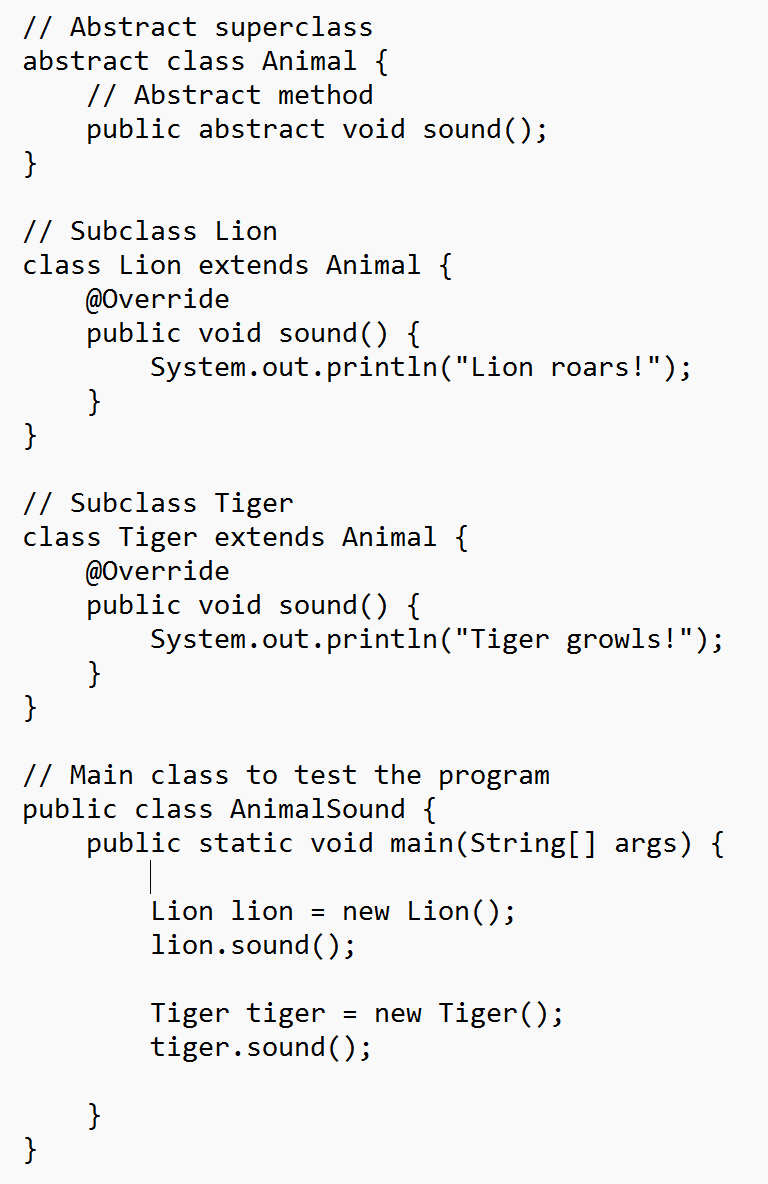
**WEEK-7**

**Program1)**

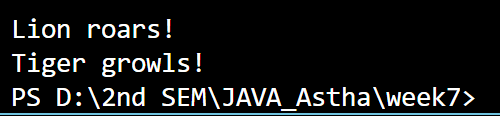
**AIM**- Write a JAVA program to create an abstract class animal with an abstract method called

Sound(). Create subclasses Lion and Tiger and implement the Sound() method to make a

specific sound for each animal.



**Output**

****

**Concepts to be known:**

* **abstract class Animal:** This is the **abstract superclass**. Has an **abstract method** sound(). It sets a method for all animal subclasses to define their own sound.
* **class Lion extends Animal:** Inherits from Animal. Overrides sound() to print: "Lion roars!"
* **class Tiger extends Animal:** Inherits from Animal. Overrides sound() to print: "Tiger growls!"

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Writing abstract method within the class Animal without declaring abstract to the class | abstract class Animal |

**Class diagram**

|  |
| --- |
| ***Animal*** |
| *+sound():void* |

|  |
| --- |
| **Lion** |
| +sound():void |

|  |
| --- |
| **Tiger** |
| +sound():void |

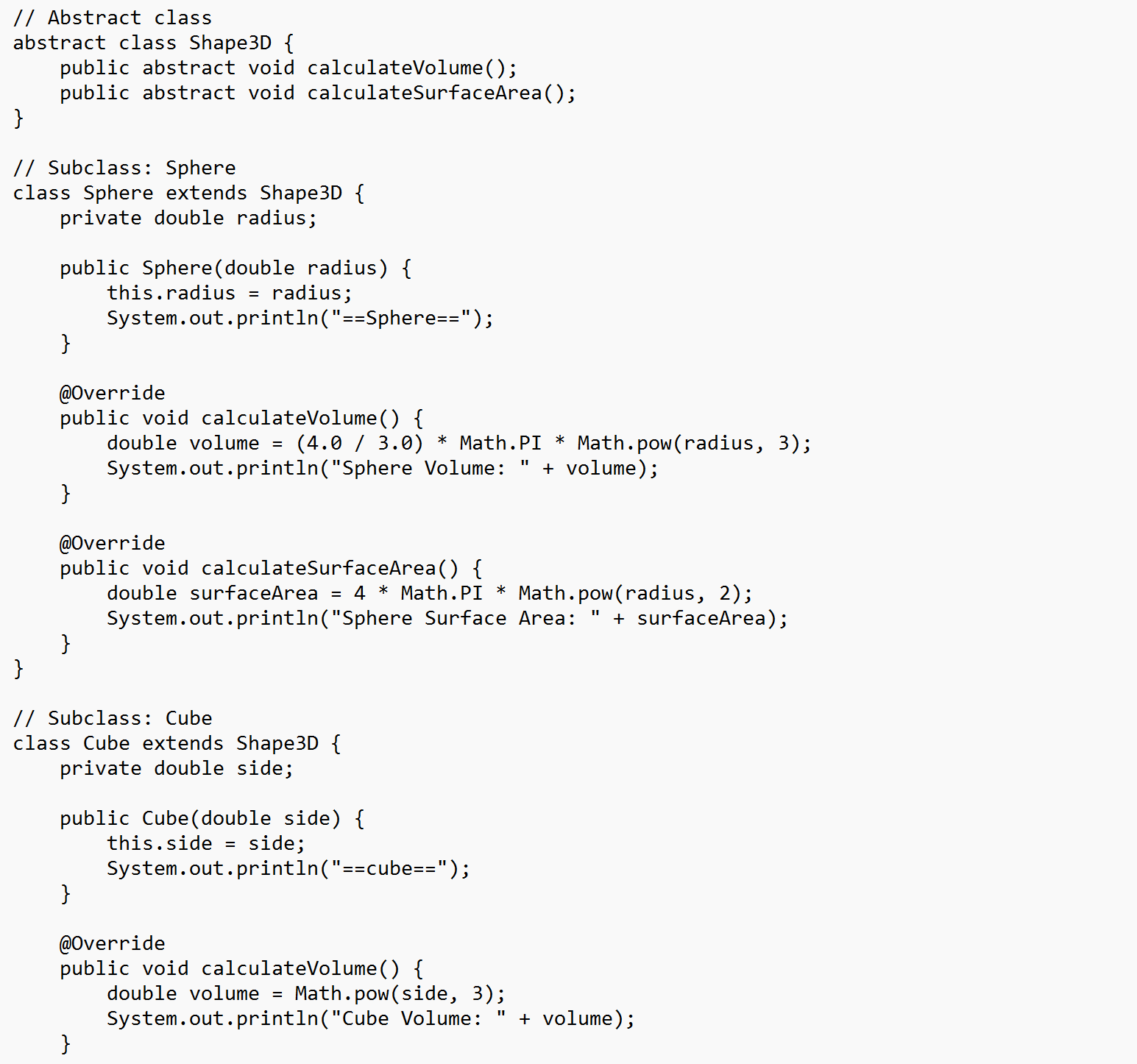
**Program2)**

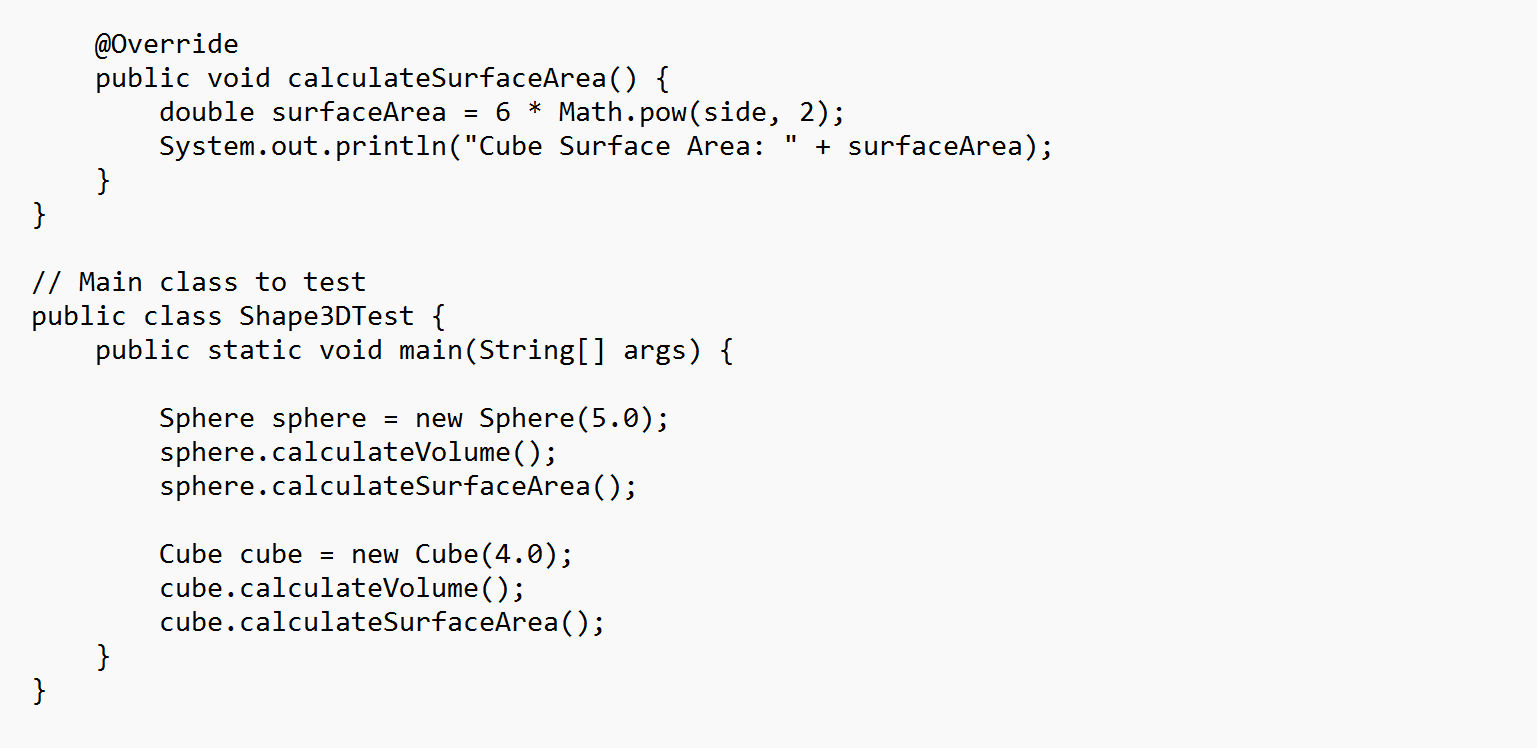
**AIM**- Write a Java program to create an abstract class Shape3D with abstract methods

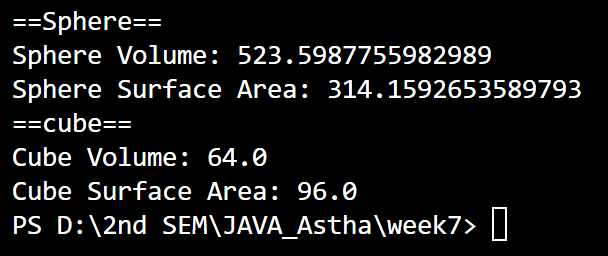
calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend

the Shape3D class and implement the respective methods to calculate the volume and surface

area of each shape.





**Output**

**Concepts to be known:**

* **abstract class Shape3D:abstract base class**. Contains **two abstract methods**: calculateVolume() and calculateSurfaceArea().
* **class Sphere extends Shape3D:**
* **Overrides both abstract methods:** calculateVolume() and calculateSurfaceArea().
* Prints details with appropriate formatting.
* **class Cube extends Shape3D:**
* **Overrides methods** with cube-specific formulas: calculateVolume() and calculateSurfaceArea().
* Constructor: to initialize the values to the variables.

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Writing abstract method within the class Animal without declaring abstract to the class | abstract class Shape3D |

**Class diagram**

|  |
| --- |
| ***Shape3D*** |
| *+calculateVolume():void*  *+calculateSurfaceArea():void* |

|  |
| --- |
| **Sphere** |
| -side:double |
| +calculateVolume():void  +calculateSurfaceArea():void |

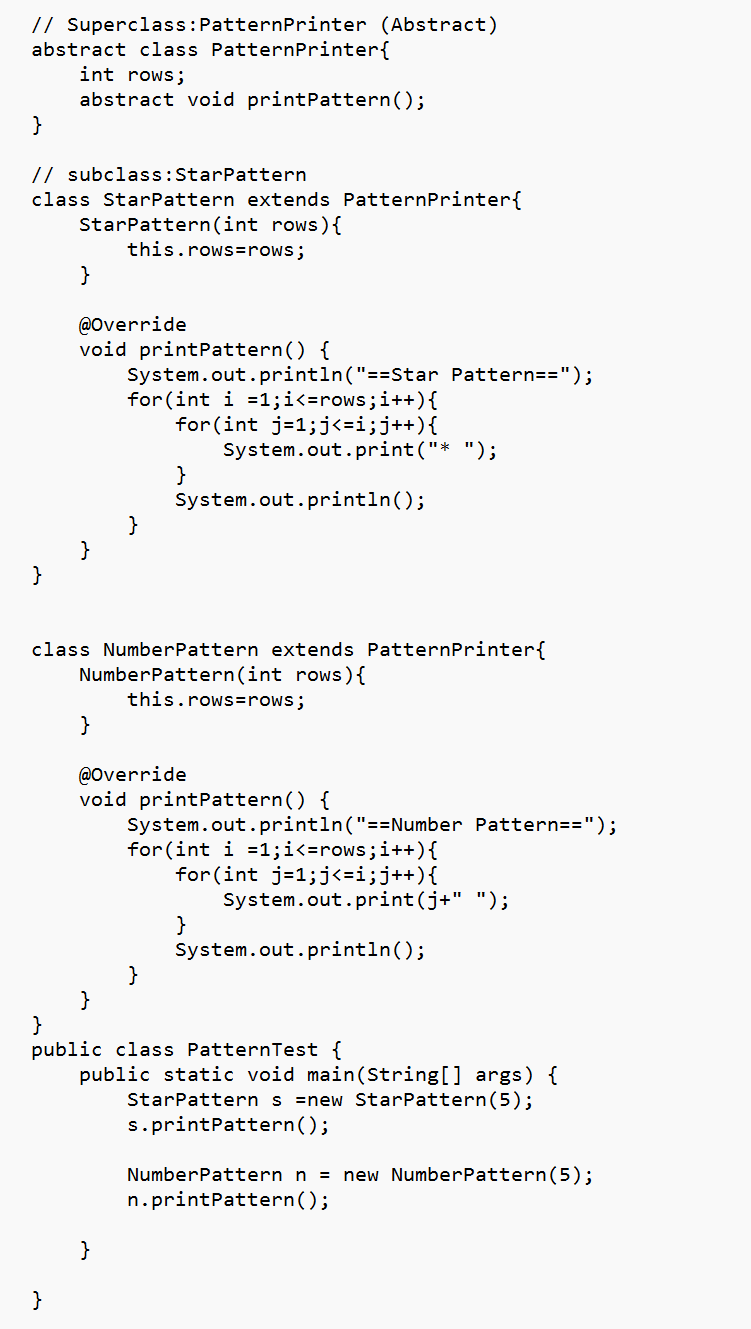
|  |
| --- |
| **Cube** |
| -radius:double |
| +calculateVolume():void  +calculateSurfaceArea():void |

**Program3)**

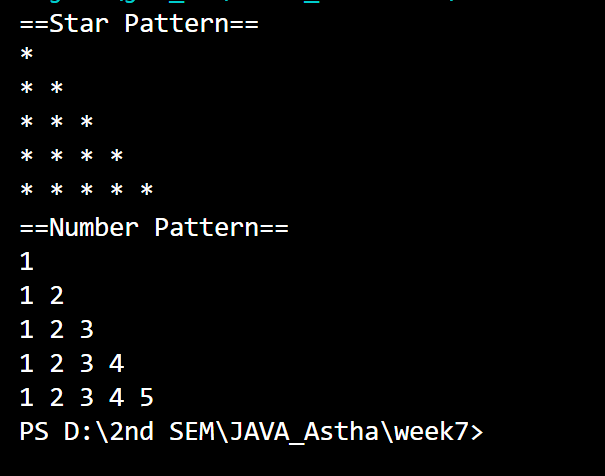
**AIM**- Write a JAVA program using an abstract class to define a method for pattern printing.

* Create an abstract class PatternPrinter with an abstract method printPattern(int rows) and a concrete method to display the pattern title.
* Implement two subclasses:
* StarPattern: prints a right angled triangle of stars(\*).
* NumberPattern: prints a right angled triangle of increasing numbers.

In the main main method , create objects of both subclasses and print the pattern for given numbers of rows.



**Output**

****

**Concepts to be known:**

* **abstract class PatternPrinter:** Abstract base class with one field rows and one abstract method printPattern().
* **class StarPattern extends PatternPrinter: Overrides** the printPattern() method to print a right-angled **star (\*) triangle pattern**. Uses **nested loops** to print increasing stars in each row.
* **Constructor** sets the number of rows.
* **class NumberPattern extends PatternPrinter: Overrides** the printPattern() method to print a right-angled **number triangle pattern**. Prints numbers from 1 to current row index using **nested loops**.
* **Constructor** sets the number of rows.

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Writing abstract method within the class Animal without declaring abstract to the class | abstract class PatternPrinter |

**Class diagram**

|  |
| --- |
| ***PatternPrinter*** |
| rows:int |
| *+printPattern():void* |

|  |
| --- |
| **StarPrinter** |
| +printPattern():void |

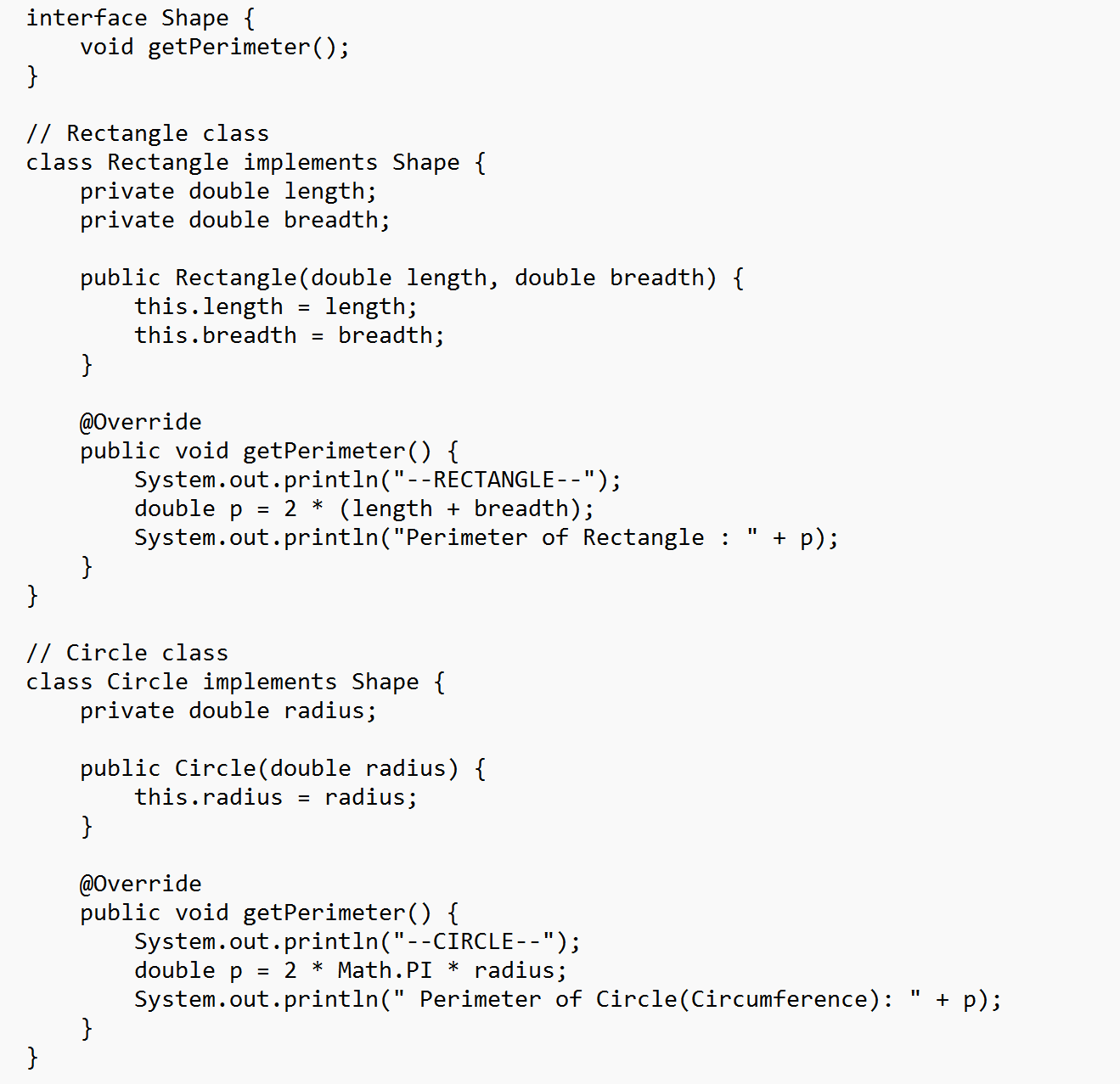
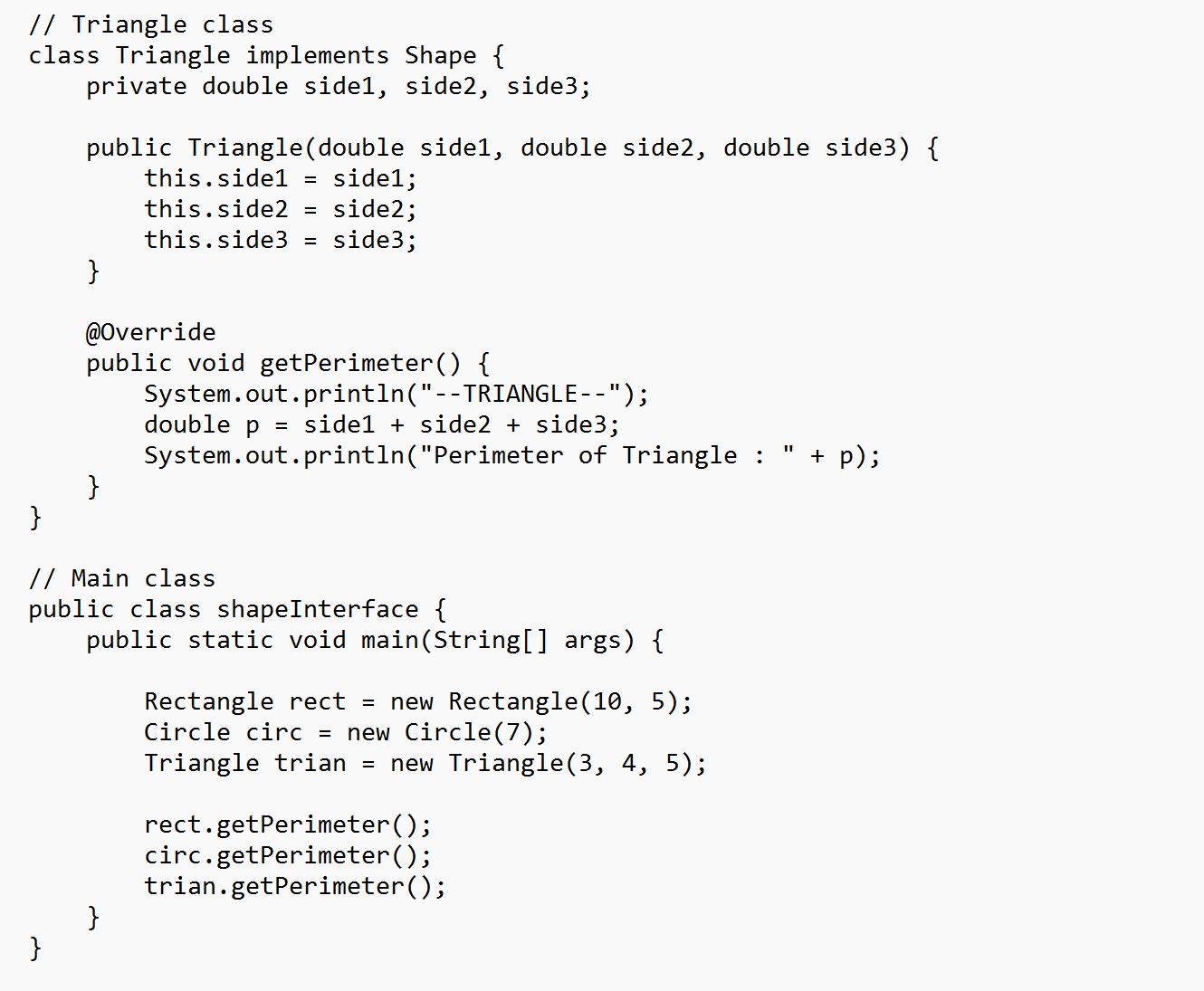
|  |
| --- |
| **StarPrinter** |
| +printPattern():void |

**WEEK-8**

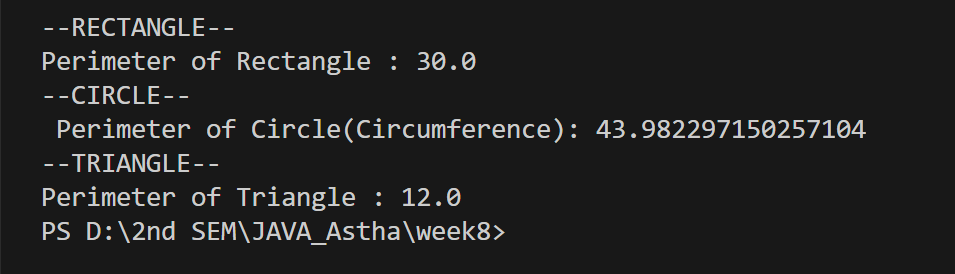
**Program1)**

**AIM**- Write a Java program to create an interface Shape with the getPerimeter() method. Create

three classes Rectangle, Circle, and Triangle that implement the Shape interface.

 Implement the getPerimeter() method for each of the three classes.

**Output**

****

**Concepts to be known:**

* **interface Shape:**Defines a contract with the method getPerimeter(). Any class implementing this interface must provide its own version of getPerimeter().
* class Rectangle implements Shape**.**
* class Circle implements Shape**.**
* class Triangle implements Shape.
* Same method name (getPerimeter()) behaves differently based on the object's class.
* **Encapsulation:** By making class variables private and providing public methods to access them, you encapsulate the data, protecting it from unauthorized access.

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Forgot to add interface keyword to the shape class | interface shape; |

**Class diagram**

|  |
| --- |
| **<<Interface>>**  **Shape** |
| +getPerimeter():void |

|  |
| --- |
| **Rectangle** |
| -length:double  -breadth:double |
| Rectangle():  + getPerimeter():void |

|  |
| --- |
| **Circle** |
| -radius:double |
| Rectangle():  + getPerimeter():void |

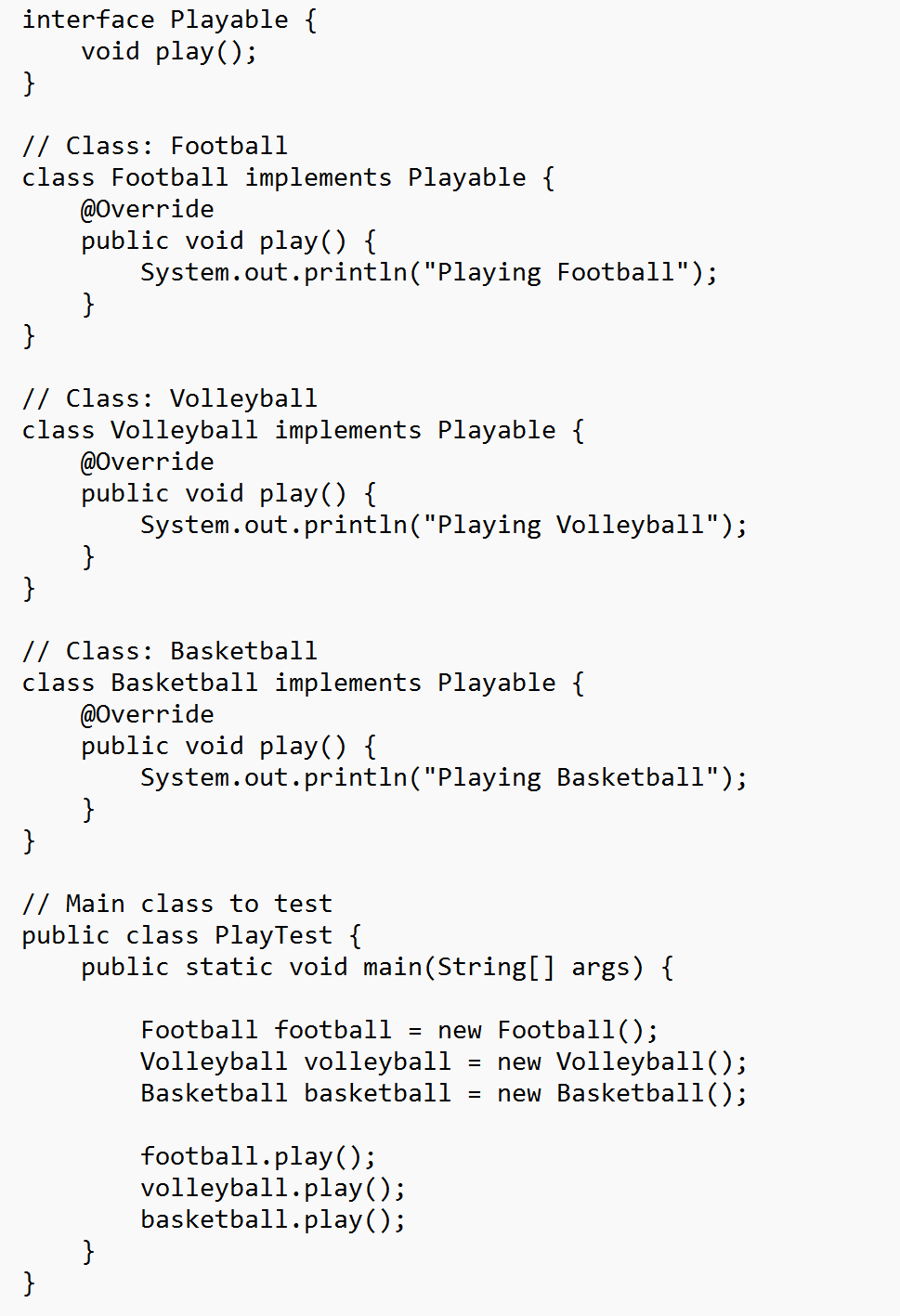
|  |
| --- |
| **Triangle** |
| -side1:double  -side2:double  -side3:double |
| Triangle():  + getPerimeter():void |

**Program2)**

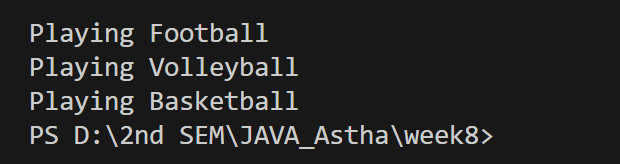
**AIM**- Write a Java program to create an interface Playable with a method play() that takes no

arguments and returns void. Create three classes Football, Volleyball, and Basketball that

implement the Playable interface and override the play() method to play the respective sports

****

**Output**



**Concepts to be known:**

* **interface Playable**: Declares a contract with a method play(). Any class implementing this interface **must define** the play() method.
* **Football, Volleyball, Basketball classes**: These **implement the Playable interface**, providing their unique version of play().
* **Method Implementation**: Each class **defines its own behavior** for how the game is played by overriding play().

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Forgot to add ; at the end  football.play() | Added ; at the end  football.play(); |
| 2. | Written implement instead of implements | Corrected to implements |

**Class diagram**

|  |
| --- |
| **<<Interface>>**  **Playable** |
| +play():void |

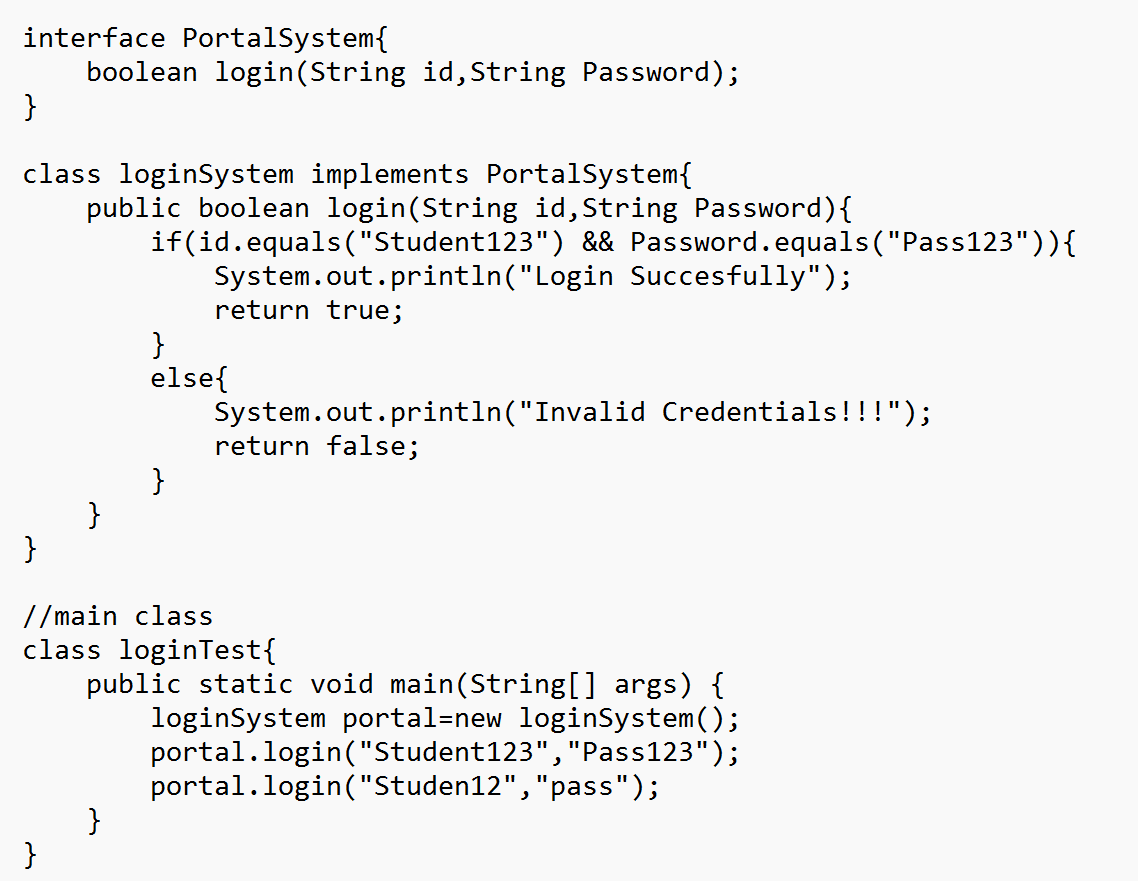
|  |
| --- |
| **Volleyball** |
| +play():void |

|  |
| --- |
| **Football** |
| +play():void |

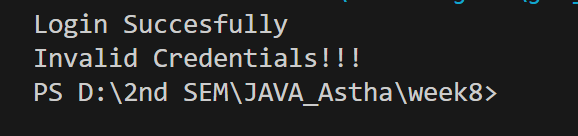
|  |
| --- |
| **Basketball** |
| +play():void |

**Program3)**

**AIM**- Write a Java program to implement a loginSystem using interfaces.



**Output**



**Concepts to be known:**

* **interface PortalSystem**: Declares a contract with a method login(String id, String Password). Any class implementing this interface **must define** the login() method.
* **loginSystem class**: This class **implements the PortalSystem interface**, providing its own version of login() that checks if the id and Password match predefined values.
* **Method Implementation**: The login() method in loginSystem **defines custom behavior**—if credentials match, it prints *"Login Successfully"*, otherwise *"Invalid Credentials!!!"*.
* **loginTest class (main method)**: Creates an object of loginSystem and **calls the login() method** with different sets of credentials to demonstrate both successful and failed login attempts.

**Error:**

|  |  |  |
| --- | --- | --- |
| S.no. | Error | Rectification |
| 1. | Forgot to add ; at the end | Added ; at the end |
| 2. | Written implement instead of implements | Corrected to implements |

**Class diagram**

|  |
| --- |
| **<<Interface>>**  **PortalSystem** |
| +login(id: String,Password: String):boolean |

|  |
| --- |
| **PortalSystem** |
| +login(id: String,Password: String):boolean |