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 • Withdraw()- whenever an amount is being withdrawn it has to be less than the current balance otherwise print insufficient balance.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.	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 is useful in this scenario.			

SNo	Title	Date	Page No	Signature
	b) 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: o StarPattern: prints a right angled triangle of stars(*). o 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

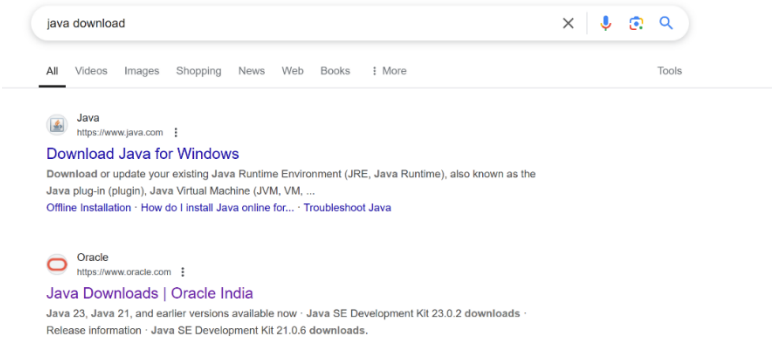
Program1)

AIM – To download and install JAVA

PROCEDURE

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

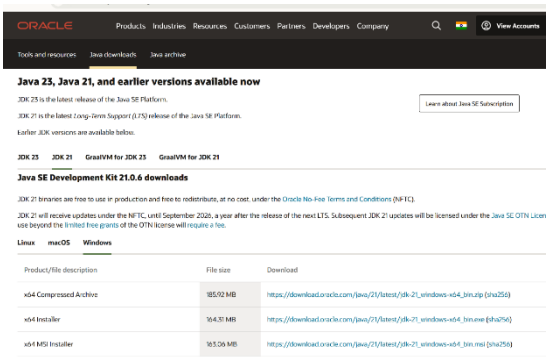
2) Go to the website of Oracle



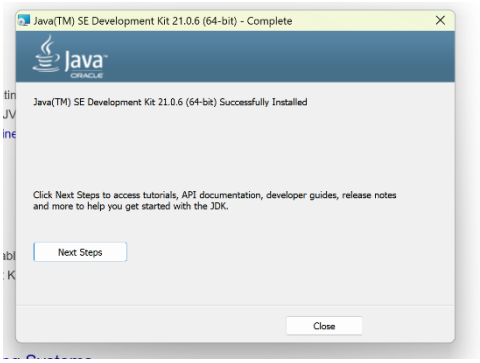
3)Download the LTS(Long-Term Support) version of jdk.

Here it is “jdk21”

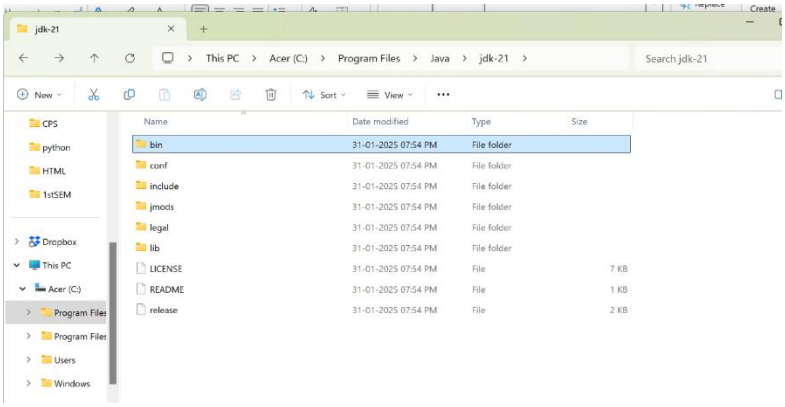
4) Select your operating System i.e. for me it is windows so I m selecting windows option. Then select x64 installer



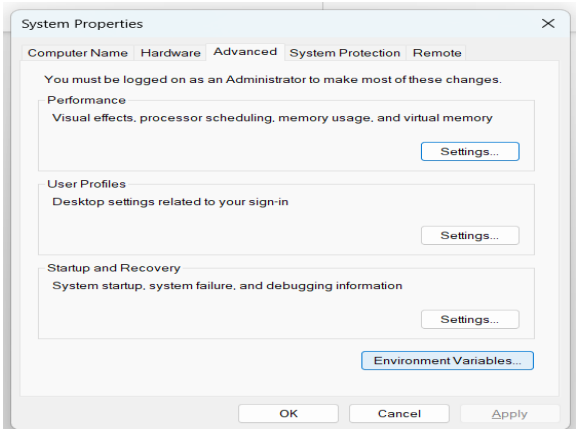
5) Download and installation.



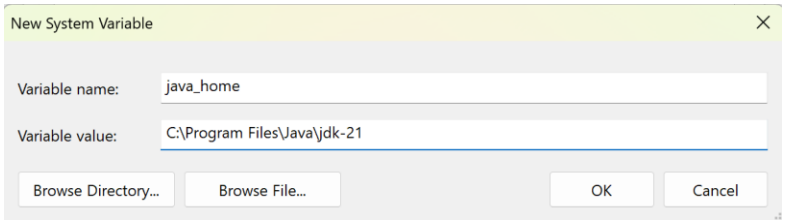
6) C Drive → Program files → Java → jdk21 → libraries + modules →bin
Now select and copy the path**.**



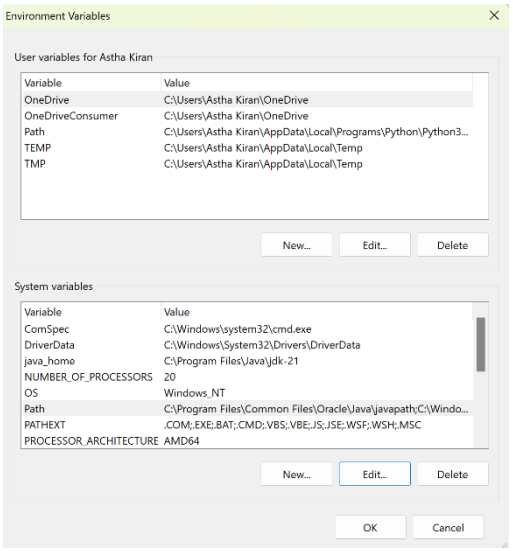
7) Press Windows + R, type sysdm.cpl, and click Ok.
8) The System Properties window will open.
9) Navigate to the Advanced tab.
10) Click on Environment Variables at the bottom.

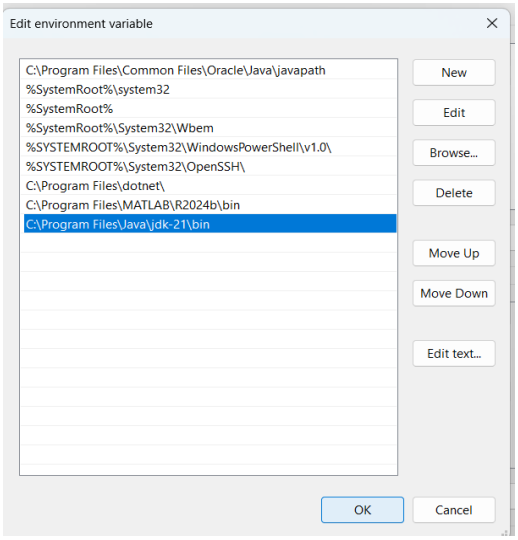


11) Under System Variables, click New.
12)Set the Variable name as Java_home.
13) Set Variable value as C:\Program Files\Java\jdk-21
(or your installation path).
Click OK.



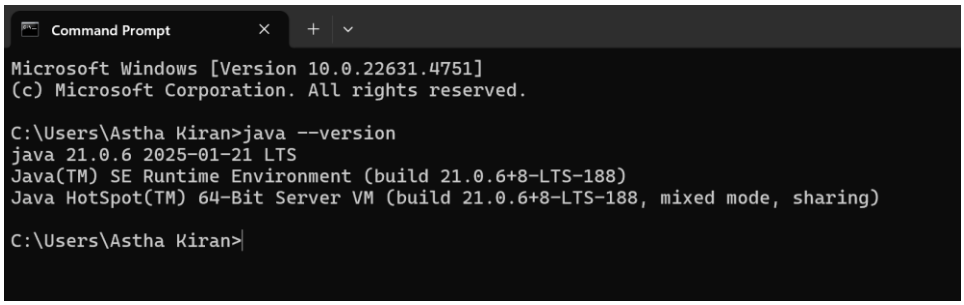
14) In System Variables, find Path and double click on it.
15)Click New and add: C:\Program Files\Java\jdk-21\bin
16)Click OK to save.





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

```
class example1{
        public static void main(String[] arg){
                System.out.println("welcome to java");}
}
```

Output

```
D:\2nd SEM\JAVA>java example1.java
welcome to java

D:\2nd SEM\JAVA>
```

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

```
class print{
        public static void main(String[] args){
                //declare variables
                String nm="Astha kiran";
                String cl="CSE/A";
                String rl="AV.SC.U4CSE24010";
                System.out.println("NAME: "+nm);
                System.out.println("CLASS: "+cl);
                System.out.println("ROLL: "+rl);}
}
```

Output

```
D:\2nd SEM\JAVA>java print.java
NAME: Astha kiran
CLASS: CSE/A
ROLL: AV.SC.U4CSE24010

D:\2nd SEM\JAVA>
```

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

```
import java.util.Scanner;

class task2{
public static void main(String[] args){
                // taking input
                Scanner input=new Scanner(System.in);
                System.out.print("enter length:");
                int l=input.nextInt();
                System.out.print("enter breadth:");
                int b=input.nextInt();

                float area_of_rect=l*b;

                System.out.println("area_of_rect:"+area_of_rect);

                input.close();

                        }
        }
```

Output

```
D:\2nd SEM\JAVA>java task2.java
enter length:10
enter breadth:5
area_of_rect:50.0

D:\2nd SEM\JAVA>
```

```
D:\2nd SEM\JAVA>java task2.java
enter length:8
enter breadth:4
area_of_rect:32.0

D:\2nd SEM\JAVA>
```

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

```
import java.util.Scanner;

class celfah{
public static void main(String[] args){
                // taking input
                Scanner input=new Scanner(System.in);
                System.out.print("enter temp in celsius:");
                double c=input.nextDouble();
                double c2f=(c*(9.0/5.0))+32;
                System.out.println("temp in fahrenheit:"+c2f);

                System.out.println();

                System.out.print("enter temp in fahrenheit:");
                double f=input.nextDouble();
                double f2c=(f-32)*(5.0/9.0);
                System.out.println("temp in celsius:"+f2c);
                }
}
```

Output

```
D:\2nd SEM\JAVA>java celfah.java
enter temp in celsius:0
temp in fahrenheit:32.0

enter temp in fahrenheit:32
temp in celsius:0.0
```

```
D:\2nd SEM\JAVA>java celfah.java
enter temp in celsius:100
temp in fahrenheit:212.0

enter temp in fahrenheit:212
temp in celsius:100.0
```

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

```
import java.util.Scanner;

class simpint{
public static void main(String[] args){
                // taking input
                Scanner input=new Scanner(System.in);
                System.out.print("enter principal:");
                int p=input.nextInt();
                System.out.print("enter rate:");
                int r=input.nextInt();
                System.out.print("enter time:");
                int t=input.nextInt();

                double si =(p*r*t)/100;

                System.out.println("Simple Interest:"+si);

                input.close();

                        }
        }
```

Output

```
D:\2nd SEM\JAVA>java simpint.java
enter principal:1000
enter rate:3
enter time:2
Simple Interest:60.0
```

```
D:\2nd SEM\JAVA>java simpint.java
enter principal:500
enter rate:2
enter time:4
Simple Interest:40.0
```

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

```
import java.util.Scanner;
class ternary_op{
        public static void main(String[] args){
                //taking input
                Scanner input=new Scanner(System.in);
                System.out.print("Enter the first number:");
                int n=input.nextInt();
                System.out.print("Enter the second number:");
                int m=input.nextInt();
                int result=(n>m)? n:m;
                System.out.println(result+" is the largest number");}
        }
```

Output

```
D:\2nd SEM\JAVA>java ternary_op.java
Enter the first number:6
Enter the second number:5
6 is the largest number
```

```
D:\2nd SEM\JAVA>java ternary_op.java
Enter the first number:10
Enter the second number:20
20is the largest number
```

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

```
import java.util.Scanner;
class fact{
        public static void main(String[] args){
                int fact=1;
                //taking input
                Scanner input=new Scanner(System.in);
                System.out.print("Enter the number:");
                int n=input.nextInt();
                for(int i=1;i<=n;i=i+1){
                        fact=fact*i;}
                System.out.print("Factorial of "+n+" is "+fact);
                }
        }
```

Output

```
D:\2nd SEM\JAVA>java fact.java
Enter the number:4
Factorial of 4 is 24
```

```
D:\2nd SEM\JAVA>java fact.java
Enter the number:6
Factorial of 6 is 720
```

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:

a) a class with name Car.
b) attributes named Car_color, Car_Brand,Fuel_type,mileage.
c) three methods named start(), stop(), service().
d) three objects car1, car2, car3.
e) one constructor which should print "welcome to CAR garage".

```
class CAR{
        // declaring attributes
        String Car_color;
        String Car_brand;
        String Fuel_type;
        int mileage;
        // constructor to initialize values
        CAR(String Car_color,String Car_brand,String Fuel_type,int mileage){
                this.Car_color=Car_color;
                this.Car_brand=Car_brand;
                this.Fuel_type=Fuel_type;
                this.mileage=mileage;
                        }

        //constructor to write a statement
        public CAR(){
                System.out.println("Welcome to CAR Garage");}

        //declaring methods
        public void Start(){
                System.out.println("CAR Started");}

        public void Stop(){
                System.out.println("CAR Stopped");}

        public void Service(){
                new CAR();
                this.Start();
                System.out.println("CAR color:"+Car_color);
                System.out.println("CAR brand:"+Car_brand);
                System.out.println("CAR Fuel Type:"+Fuel_type);
                System.out.println("CAR mileage:"+mileage);
                this.Stop();
                System.out.println();
                        }

        public static void main(String[] args){
                //creating objects for class CAR
                CAR car1=new CAR("Red","MARUTI","Petrol",100);
                car1.Service();
                CAR car2=new CAR("Blue","HYUNDAI","Petrol",100);
                car2.Service();
                CAR car3=new CAR("Black","Alto","Diesel",80);
                car3.Service();
                }
        }
```

OUTPUT

```
D:\2nd SEM\JAVA>java CAR.java
Welcome to CAR Garage
CAR Started
CAR color:Red
CAR brand:MARUTI
CAR Fuel Type:Petrol
CAR mileage:100
CAR Stopped

Welcome to CAR Garage
CAR Started
CAR color:Blue
CAR brand:HYUNDAI
CAR Fuel Type:Petrol
CAR mileage:100
CAR Stopped

Welcome to CAR Garage
CAR Started
CAR color:Black
CAR brand:Alto
CAR Fuel Type:Diesel
CAR mileage:80
CAR Stopped
```

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():

a) In deposit()- whenever an amount is deposited, it has to be updated with the current amount.

b) Withdraw()- whenever an amount is being withdrawn it has to be less than the current balance otherwise print insufficient balance.

```
class BankAccount{
        String Name;
        String Accno;
        int currbal;
        BankAccount(String Name,String Accno,int currbal){
                this.Name=Name;
                this.Accno=Accno;
                this.currbal=currbal;
                System.out.println("Customer Details:");
                System.out.println("NAME:"+Name);
                System.out.println("Account No.:"+Accno);
                System.out.println();
                        }
        public void Withdraw(int W_amount){
                if(W_amount<currbal){
                        currbal=currbal-W_amount;
                        System.out.println("After withdrawing Current Balance:"+currbal);
                        System.out.println();}
                else{
                        System.out.println("Insufficient Balance");
                        System.out.println();}
                        }
        public int Deposit(int D_amount){
                System.out.println("Amount Deposited:"+D_amount);
                currbal=currbal+D_amount;
                return currbal;
                        }
        public static void main(String[] args){
                BankAccount B1=new BankAccount("RAM","AXXX2345RE34",10000);
                B1.Withdraw(15000);
                B1.Withdraw(7000);
                int F_Amount=B1.Deposit(8000);
                System.out.println("After deppsiting, Final Balance: "+F_Amount);
                }
        }
```

OUTPUT

```
D:\2nd SEM\JAVA>java BankAccount.java
Customer Details:
NAME:RAM
Account No.:AXXX2345RE34

Insufficient Balance

After withdrawing Current Balance:3000

Amount Deposited:8000
After deppsiting, Final Balance: 11000
```

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:

a) The class should contain various attributes such as “title_of_book, Author, year_of_publication”.
b) It should also contain a constructor with parameters which initializes “title_of_book, Author, year_of_publication”.
c) Create a method which displays the details of the book “title_of_book, Author, year_of_publication”.
d) Display the details of the two books by creating two objects.

```
public class Book {
    //declaring attributes
        String Title_of_book;
        String Author;
        int Year_of_publication;

        //constructor to initialize values
        Book(String Title_of_book,String Author,int Year_of_publication){
                this.Title_of_book=Title_of_book;
                this.Author=Author;
                this.Year_of_publication=Year_of_publication;
                        }

        //creating amethod
        public void getbook(){
                System.out.println("Book Name:"+Title_of_book);
                System.out.println("Author:"+Author);
                System.out.println("Year of publication:"+Year_of_publication);
                System.out.println();
                        }

        public static void main(String[] args){
                //creating obects for class Book
                Book book1=new Book("Missing 400 days","Chetan Bhagat",2002);
                book1.getbook();
                Book book2=new Book("arranged marriage murder","Chetan Bhagat",2000);
                book2.getbook();
                Book book3=new Book("The Reappearance of Rachel Prince","XXYZ",2002);
                book3.getbook();
                        }
}
```

OUTPUT

```
Welcome to CAR Garage
CAR Started
CAR color:Red
CAR brand:MARUTI
CAR Fuel Type:Petrol
CAR mileage:100
CAR Stopped

Welcome to CAR Garage
CAR Started
CAR color:Blue
CAR brand:HYUNDAI
CAR Fuel Type:Petrol
CAR mileage:100
CAR Stopped

Welcome to CAR Garage
CAR Started
CAR color:Black
CAR brand:Alto
CAR Fuel Type:Diesel
CAR mileage:80
CAR Stopped

PS D:\2nd SEM\JAVA_Astha>
```

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:

a) 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.

a) 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.

```
class Myclass{
        // declaring variables
        static int count = 0;
        final double pi = 3.1415;

        //constructor for increasing count value
        Myclass(){
                count = count+1;
                        }

        public void display(){
                System.out.println("count is:"+count);
                System.out.println("double is:"+pi);
                System.out.println();
                        }

        // main
        public static void main(String[] args){
                Myclass m1 = new Myclass();
                m1.display();
                Myclass m2 = new Myclass();
                m2.display();
                System.out.println("The final count is:"+count);
                System.out.println("double is:"+m2.pi);
                        }
}
```

OUTPUT

```
count is:1
double is:3.1415

count is:2
double is:3.1415

The final count is:2
double is:3.1415
PS D:\2nd SEM\JAVA_Astha>
```

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.

```
import java.util.Scanner;

class Simple_Calculator {
    public void add(double a, double b) {
        double addition = a + b;
        System.out.println("Addition: " + addition);
    }

    public void sub(double a, double b) {
        double subtraction = a - b;
        System.out.println("Subtraction: " + subtraction);
    }

    public void mult(double a, double b) {
        double multiplication = a * b;
        System.out.println("Multiplication: " + multiplication);
    }

    public void div(double a, double b) {
        if (b == 0) {
            System.out.println("Error: Division by zero.");
            return;
        }
        double division = a / b;
        System.out.println("Division: " + division);
    }
}

class Adv_calculator extends Simple_Calculator {
    public void floor_div(double a, double b) {
        if (b == 0) {
            System.out.println("Error: Division by zero.");
            return;
        }
        double division = Math.floor(a / b);
        System.out.println("Floor Division: " + division);
    }

    public void mod_div(double a, double b) {
        if (b == 0) {
            System.out.println("Error: Modulus by zero.");
            return;
        }
        double mod = a % b;
        System.out.println("Modulus: " + mod);
    }
}

class Super_calculator extends Adv_calculator {
    public double Square(double num) {
        return num * num;
    }

   public double SquareRoot(double num) {
       return Math.sqrt(num);
   }

   public double CubeRoot(double num) {
       return Math.cbrt(num);
   }
```

```
    public double Floor(double num) {
        return Math.floor(num);
    }

    public double Ceil(double num) {
        return Math.ceil(num);
    }
}
```

```
// Your main method goes here
public class calculator {
    public static void main(String[] args) {
        Super_calculator calc = new Super_calculator();

        //taking input
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter first number: ");
        double a = sc.nextDouble();
```

```
        System.out.print("Enter second number: ");
        double b = sc.nextDouble();

        // Basic operations
        calc.add(a, b);
        calc.sub(a, b);
        calc.mult(a, b);
        calc.div(a, b);

        // Advanced operations
        calc.floor_div(a, b);
        calc.mod_div(a, b);

        // // Super operations
        System.out.println("Square: " + calc.Square(a));
        System.out.println("Square Root: " + calc.SquareRoot(a));
        System.out.println("Cube Root: " + calc.CubeRoot(a));
        System.out.println("Floor Value: " + calc.Floor(a));
        System.out.println("Ceil Value: " + calc.Ceil(a));
    }
}
```

Output

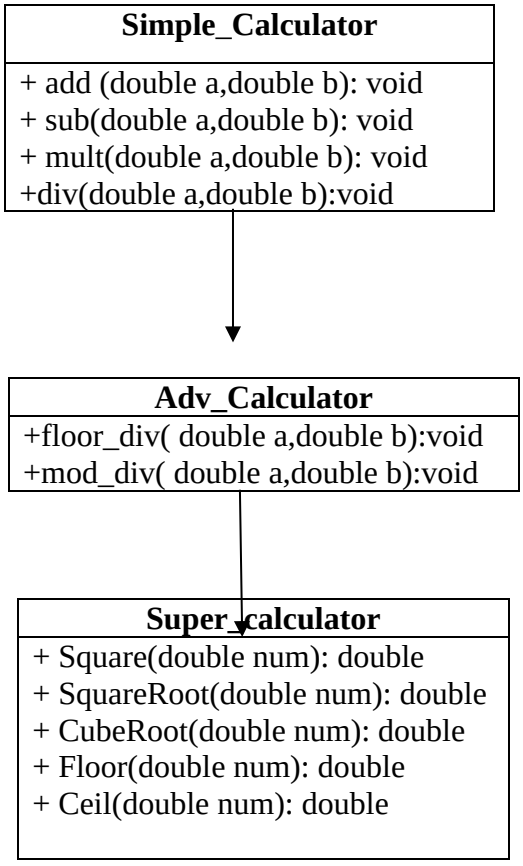
```
Enter first number: 10.5
Enter second number: 2
Addition: 12.5
Subtraction: 8.5
Multiplication: 21.0
Division: 5.25
Floor Division: 5.0
Modulus: 0.5
Square: 110.25
Square Root: 3.24037034920393
Cube Root: 2.1897595699439445
Floor Value: 10.0
Ceil Value: 11.0
PS D:\2nd SEM\JAVA_Astha>
```

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

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.

Class Diagram



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

b) 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

```
// Base class: Vehicle
class Vehicle {
    String brand;
    int speed;

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

    // Method to display details
    public void displayDetails() {
        System.out.println("Brand: " + brand);
        System.out.println("Speed: " + speed + " km/h");
    }

    // Method to indicate vehicle is starting
    public void start() {
        System.out.println(brand + " is starting...");
    }
}

// Subclass: Car
class Car extends Vehicle {
    int Doors;
    int seatingCapacity;

    // Constructor
    public Car(String brand, int speed, int Doors, int seatingCapacity) {
        super(brand, speed);
        this.Doors = Doors;
        this.seatingCapacity = seatingCapacity;
    }

    // Overriding displayDetails
    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Number of Doors: " + Doors);
        System.out.println("Seating Capacity: " + seatingCapacity);
    }
```

```
}

// Subclass: Bike
class Bike extends Vehicle {
    boolean Gears;

    // Constructor
    public Bike(String brand, int speed, boolean hasGears) {
        super(brand, speed);
        this.Gears = Gears;
    }

    // Overriding displayDetails
    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Has Gears: " + (Gears ? "Yes" : "No"));
    }
}

// Subclass: Truck
class Truck extends Vehicle{
    int capacity;

    //constructor
    public Truck(String brand, int speed, int capacity){
        super(brand, speed);
        this.capacity = capacity;
    }

    public void truckdetails(){
        System.out.println("Capacity:"+capacity+" Tons");
    }

    // Overriding displayDetails
    @Override
    public void displayDetails() {
        super.displayDetails();
        truckdetails();
    }
}

// Main class to test
public class VehicleRentalSystem {
    public static void main(String[] args) {
        Car car1 = new Car("Toyota", 180, 4, 5);
        Bike bike1 = new Bike("Yamaha", 120, true);
        Truck truck1 = new Truck("Mahindra", 100 ,500);

        System.out.println("Car Details:");
        car1.displayDetails();
        car1.start();

        System.out.println("\nBike Details:");
        bike1.displayDetails();
        bike1.start();

        System.out.println("\nTruck Details:");
        truck1.displayDetails();
        truck1.start();
    }
}
```

Output

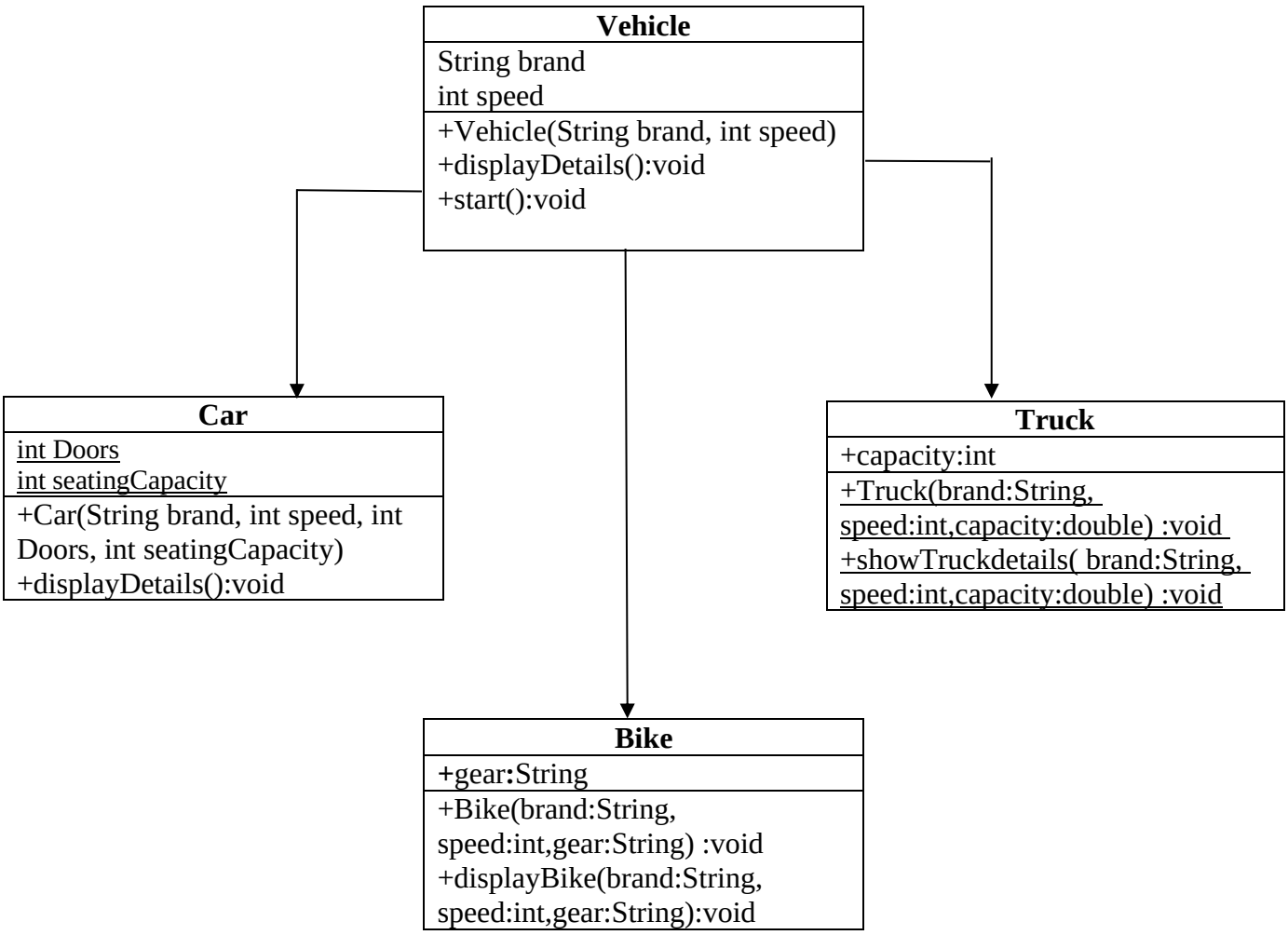
```
Car Details:
Brand: Toyota
Speed: 180 km/h
Number of Doors: 4
Seating Capacity: 5
Toyota is starting...

Bike Details:
Brand: Yamaha
Speed: 120 km/h
Has Gears: No
Yamaha is starting...

Truck Details:
Brand: Mahindra
Speed: 100 km/h
Capacity:500 Tons
Mahindra is starting...
PS D:\2nd SEM\JAVA_Astha>
```

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



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)]

```
// Base class: Vehicle
class vehicle {
    public void displayInfo() {
        System.out.println("Vehicle information:");
    }
}

// Subclass: Car
class car extends vehicle {
    String carCompany;
    String carModel;
    double carPrice;
    int seatingCapacity;
    boolean petrol_Or_Not;

    // Constructor
    public car(String carCompany, String carModel, double carPrice, int seatingCapacity, boolean petrol_Or_Not) {
        this.carCompany = carCompany;
        this.carModel = carModel;
        this.carPrice = carPrice;
        this.seatingCapacity = seatingCapacity;
        this.petrol_Or_Not = petrol_Or_Not;
    }

    // Overriding displayInfo() method
    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Car Company: " + carCompany);
        System.out.println("Car Model: " + carModel);
        System.out.println("Car Price: Rs." + carPrice);
        System.out.println("Seating Caspacity: " + seatingCapacity);
        System.out.println("Petrol Vehicle: " + (petrol_Or_Not ? "Yes" : "No"));
    }
}

// Main class
public class Final {
    public static void main(String[] args) {
        // Create a Car object
        car car1 = new car("Hyundai", "Creta", 1250000, 5, true);

        // Call the displayInfo() method
        car1.displayInfo();
    }
}
```

Output

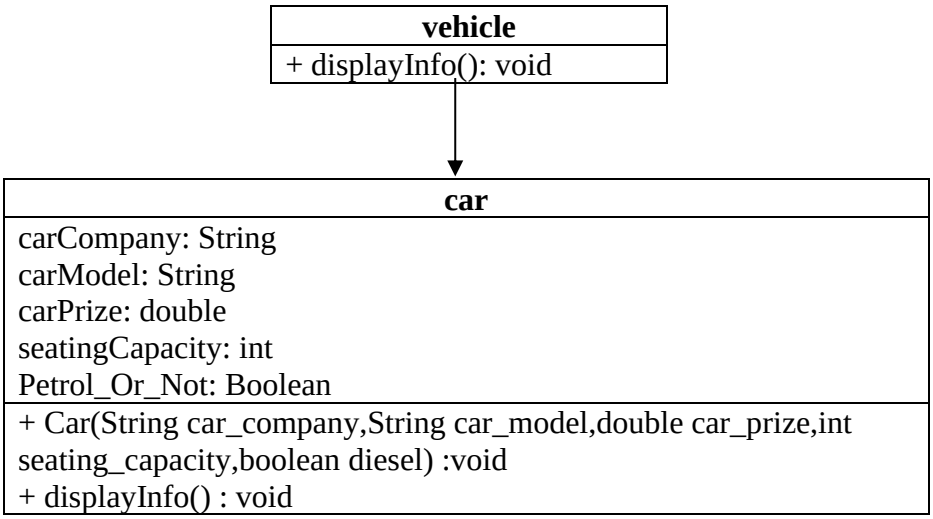
```
Vehicle information:
Car Company: Hyundai
Car Model: Creta
Car Price: Rs.1250000.0
Seating Caspacity: 5
Petrol Vehicle: Yes
PS D:\2nd SEM\JAVA_Astha>
```

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



Program2)

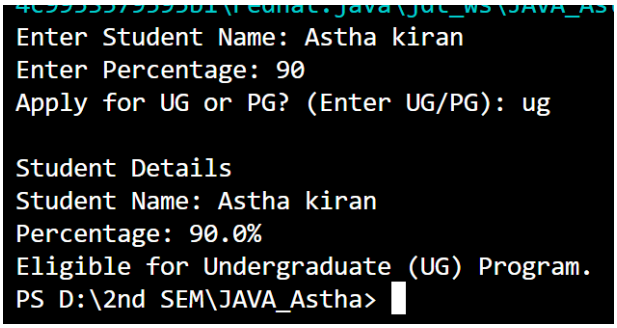
AIM- A college is developing an automated admission system that veifies students eligibility for undergraduate(UG) and post-graduate(PG) programs. Each program has different eligibility criteria based on the students percentage in their previous qualification.

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

```
import java.util.Scanner;

// Base class: Student
class Student {
    String name;
    double percentage;

    // Constructor
    public Student(String name, double percentage) {
        this.name = name;
        this.percentage = percentage;
    }

    // Method to display basic details
    public void displayDetails() {
        System.out.println("\nStudent Details");
        System.out.println("Student Name: " + name);
        System.out.println("Percentage: " + percentage + "%");
    }
}

// Subclass for UG Admission
class UG extends Student {
    public UG(String name, double percentage) {
        super(name, percentage);
    }

    public void checkEligibility() {
        displayDetails();
        if (percentage >= 60) {
            System.out.println("Eligible for Undergraduate (UG) Program.");
        } else {
            System.out.println("Not Eligible for UG Program.");
        }
    }
}

// Subclass for PG Admission
class PG extends Student {
    public PG(String name, double percentage) {
        super(name, percentage);
    }

    public void checkEligibility() {
        displayDetails();
        if (percentage >= 70) {
            System.out.println("Eligible for Postgraduate (PG) Program.");
        } else {
            System.out.println("Not Eligible for PG Program.");
        }
    }
}

// Main class
public class AdmissionSystem {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
```

```
        // Get student info
        System.out.print("Enter Student Name: ");
        String name = input.nextLine();

        System.out.print("Enter Percentage: ");
        double percentage = input.nextDouble();

        System.out.print("Apply for UG or PG? (Enter UG/PG): ");
        String program = input.next();

        // Check eligibility
        if (program.equalsIgnoreCase("UG")) {
            UG ugStudent = new UG(name, percentage);
            ugStudent.checkEligibility();
        }
        else if (program.equalsIgnoreCase("PG")) {
            PG pgStudent = new PG(name, percentage);
            pgStudent.checkEligibility();
        }
        else {
            System.out.println("Invalid program selected.");
        }

        input.close();
    }
}
```

Output

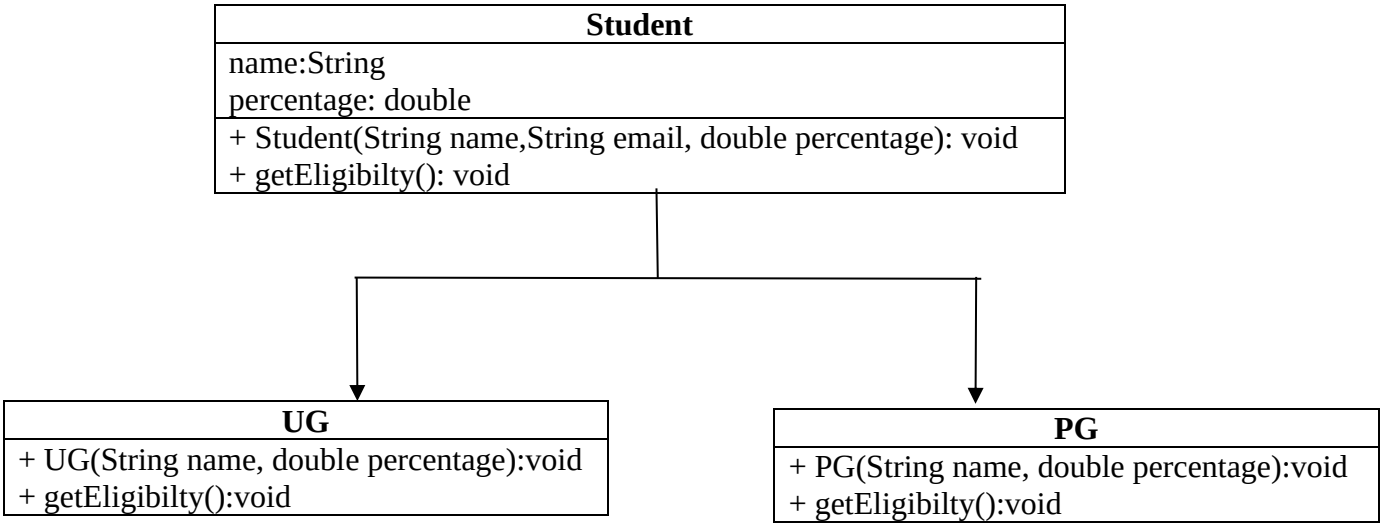


S.No.	Errors	Rectification
1.	error: ';' expected String name: input.next();	Adding String name: input.nextLine();
2.	Syntax error, insert "}" to complete ClassBody	Adding } at the end

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



Program3)

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

- Add 2 int
- Add 3 int
- Add 2 doubles

```
class Calculator {

    // Method 1: Add 2 integers
    public void add(int a, int b) {
        int s=a+b;
        System.out.println("Sum of 2 integers: " + s);    }

    // Method 2: Add 3 integers
    public void add(int a, int b, int c) {
        int s=a+b+c;
        System.out.println("Sum of 3 integers: " + s);
    }

    // Method 3: Add 2 doubles
    public void add(double a, double b) {
        double s=a+b;
        System.out.println("Sum of 2 doubles: " + s);
    }
}

// Main class
public class CalculatorDemo {
    public static void main(String[] args) {
        Calculator calc = new Calculator();

        // Call each add method
        calc.add(10, 20);           // 2 int
        calc.add(5, 10, 15);        // 3 int
        calc.add(3.5, 4.5);      // 2 double
    }
}
```

Output

```
Sum of 2 integers: 30
Sum of 3 integers: 30
Sum of 2 doubles: 8.0
PS D:\2nd SEM\JAVA_Astha>
```

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.

```
// Base class
class Shape {
    // Area of square
    public double calculateArea(double a) {
        return a * a;
    }

    // Area of rectangle
    public double calculateArea(double a, double b) {
        return a * b;
    }

}

// Subclass for Circle
class Circle extends Shape {

    @Override
    public double calculateArea(double a) {
        return Math.PI * a * a;
    }
}

// Main class
public class shapeDemo {
    public static void main(String[] args) {
        Shape s = new Shape();

        double squareArea = s.calculateArea(5); // Square
        double rectangleArea = s.calculateArea(4, 6); // Rectangle

        Circle c = new Circle();
        double circleArea = c.calculateArea(3.5); // Circle

        System.out.println("Area of Square: " + squareArea);
        System.out.println("Area of Rectangle: " + rectangleArea);
        System.out.println("Area of Circle: " + circleArea);
    }
}
```

Output

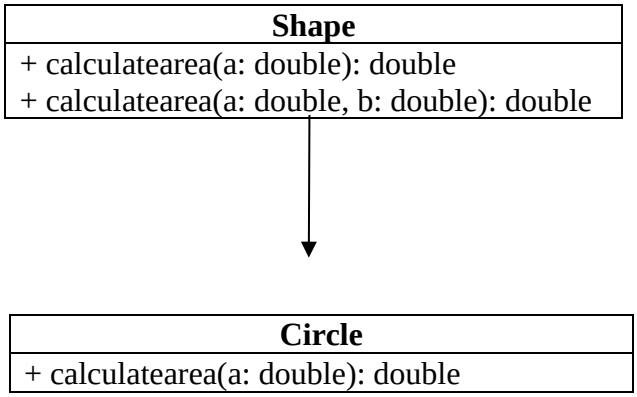
```
Area of Square: 25.0
Area of Rectangle: 24.0
Area of Circle: 38.48451000647496
PS D:\2nd SEM\JAVA_Astha>
```

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



WEEK-6

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.

```
// Abstract superclass
abstract class Animal {
    // Abstract method
    public abstract void sound();
}

// Subclass Lion
class Lion extends Animal {
    @Override
    public void sound() {
        System.out.println("Lion roars!");
    }
}

// Subclass Tiger
class Tiger extends Animal {
    @Override
    public void sound() {
        System.out.println("Tiger growls!");
    }
}

// Main class to test the program
public class AnimalSound {
    public static void main(String[] args) {

        Lion lion = new Lion();
        lion.sound();

        Tiger tiger = new Tiger();
        tiger.sound();

    }
}
```

Output

```
Lion roars!
Tiger growls!
PS D:\2nd SEM\JAVA_Astha\week7>
```

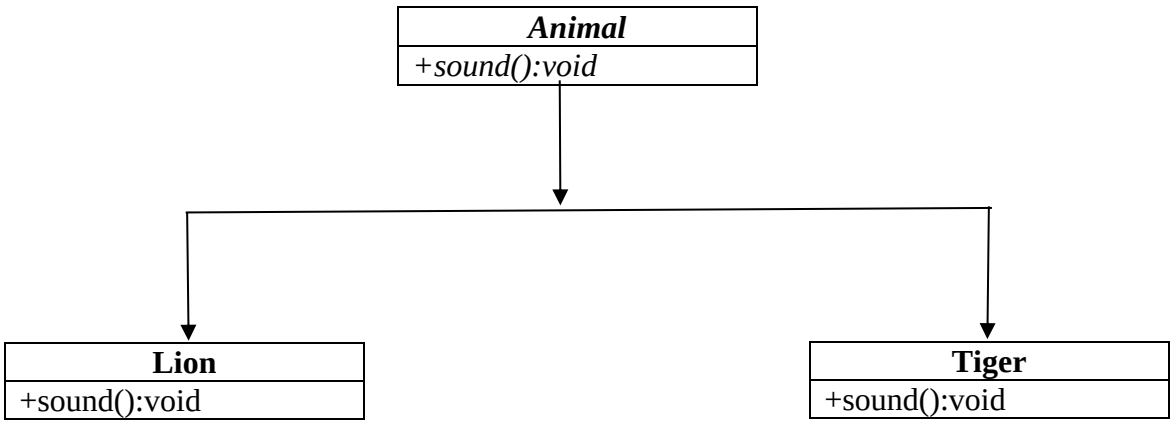
Concepts to be known:

- **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



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.

```
// Abstract class
abstract class Shape3D {
    public abstract void calculateVolume();
    public abstract void calculateSurfaceArea();
}

// Subclass: Sphere
class Sphere extends Shape3D {
    private double radius;

    public Sphere(double radius) {
        this.radius = radius;
        System.out.println("==Sphere==");
    }

    @Override
    public void calculateVolume() {
        double volume = (4.0 / 3.0) * Math.PI * Math.pow(radius, 3);
        System.out.println("Sphere Volume: " + volume);
    }

    @Override
    public void calculateSurfaceArea() {
        double surfaceArea = 4 * Math.PI * Math.pow(radius, 2);
        System.out.println("Sphere Surface Area: " + surfaceArea);
    }
}

// Subclass: Cube
class Cube extends Shape3D {
    private double side;

    public Cube(double side) {
        this.side = side;
        System.out.println("==cube==");
    }

    @Override
    public void calculateVolume() {
        double volume = Math.pow(side, 3);
        System.out.println("Cube Volume: " + volume);
    }

     @Override
     public void calculateSurfaceArea() {
         double surfaceArea = 6 * Math.pow(side, 2);
         System.out.println("Cube Surface Area: " + surfaceArea);
     }
 }

 // Main class to test
 public class Shape3DTest {
     public static void main(String[] args) {

         Sphere sphere = new Sphere(5.0);
         sphere.calculateVolume();
         sphere.calculateSurfaceArea();

         Cube cube = new Cube(4.0);
         cube.calculateVolume();
         cube.calculateSurfaceArea();
     }
 }
```

Output

```
==Sphere==
Sphere Volume: 523.5987755982989
Sphere Surface Area: 314.1592653589793
==cube==
Cube Volume: 64.0
Cube Surface Area: 96.0
PS D:\2nd SEM\JAVA_Astha\week7>
```

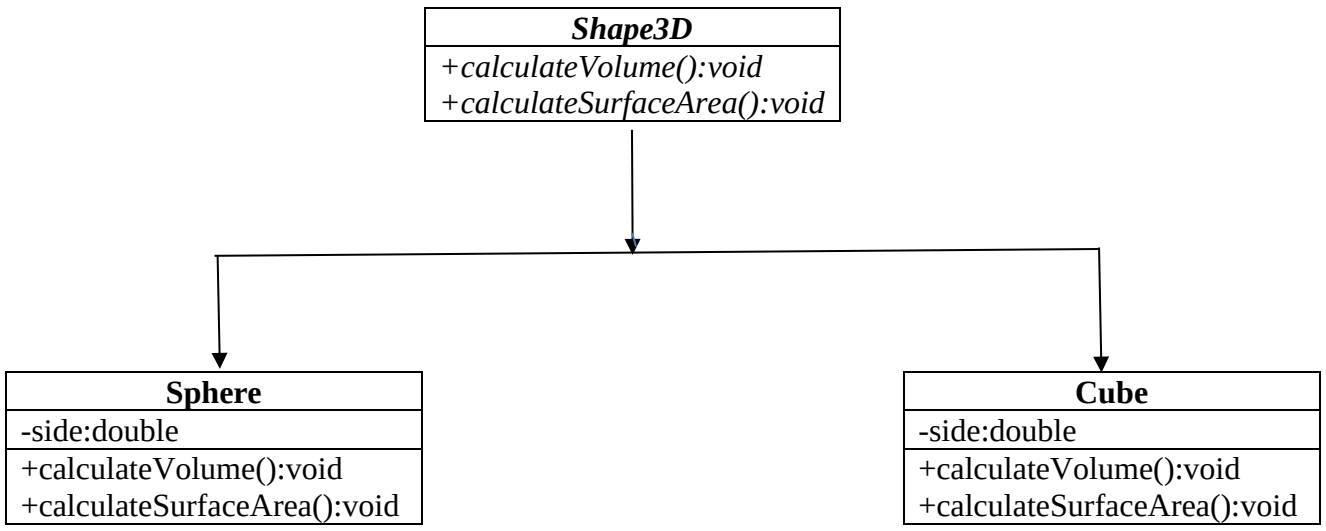
Concepts to be known:

- **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



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.

```
// Superclass:PatternPrinter (Abstract)
abstract class PatternPrinter{
    int rows;
    abstract void printPattern();
}

// subclass:StarPattern
class StarPattern extends PatternPrinter{
    StarPattern(int rows){
        this.rows=rows;
    }

    @Override
    void printPattern() {
        System.out.println("==Star Pattern==");
        for(int i =1;i<=rows;i++){
            for(int j=1;j<=i;j++){
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}


class NumberPattern extends PatternPrinter{
    NumberPattern(int rows){
        this.rows=rows;
    }

    @Override
    void printPattern() {
        System.out.println("==Number Pattern==");
        for(int i =1;i<=rows;i++){
            for(int j=1;j<=i;j++){
                System.out.print(j+" ");
            }
            System.out.println();
        }
    }
}
public class PatternTest {
    public static void main(String[] args) {
        StarPattern s =new StarPattern(5);
        s.printPattern();

        NumberPattern n = new NumberPattern(5);
        n.printPattern();

    }

}
```

Output

```
==Star Pattern==
*
* *
* * *
* * * *
* * * * *
==Number Pattern==
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
PS D:\2nd SEM\JAVA_Astha\week7>
```

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