**23CSE111**

OOPS

**LAB MANUAL**

A logo with pink letters

Description automatically generated

**Department of CSE**

**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Verified By Name: A.Vijay Kumar.**

**Roll No: 24002**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Programs | Date | Pg:No | Signature |
| WEEK-1 | a) Download and install Java software.  b) Write a java program that prints message”welcome to java  program”.  c) Write a java program that prints name,roll number,section of a  student. |  |  |  |
| WEEK-2 | a) Write a java program to calculate the area of the rectangle.  b) Write a java program to convert temperature from Celsius to  Fahrenheit and viceversa.  c) Write a java program to calculate the simple intrest.  d) Write a java program to find largest of three numbers using  ternary operator.  e) Write a java program to find factorial of number. |  |  |  |
| WEEK-3 | 1. Write a java program with the following instructions.  a) Create a class with name car.  b) Create four attributes named car\_colour,car\_brand,fuel  type,top\_speed.  c) Create three method named  “Start\_Racing”,”End\_Race”.{ }  d) Create three objects named Car1,Car2,Car3.  e) Create a constructor which should print “Welcome to car  garage”.  2. Write a class by writing java program named Bank Account  with two methods “deposits and withdraw”.  a) In deposit method whenever an amount is deposited it  has to be updated with current amount (logic C.A+D.A).  b) With draw amount whenever an amount is being  withdraw it has to be less than the current amount less  than the amount else print “Insufficient funds”. |  |  |  |
| WEEK-4 | 1.Write a java program with class named |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

Week-1

* Program : 1
* Aim : Download and Install Java Software.
* Step 1 : Visit chrome and search “ java download”.And select Oracle website.

A screenshot of a computer

Description automatically generated

* Step 2 : Now open Oracle website scroll down and now select “JDK 21” for

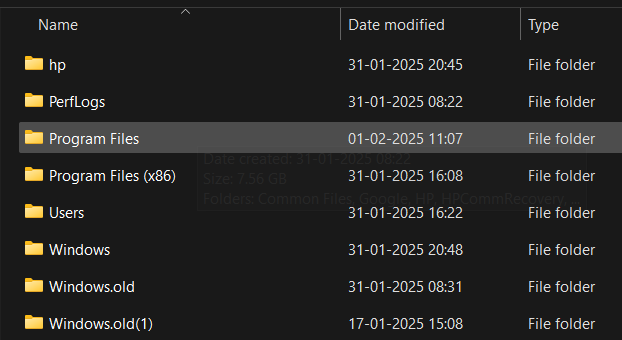
Windows and select “X64 installer” and download it.

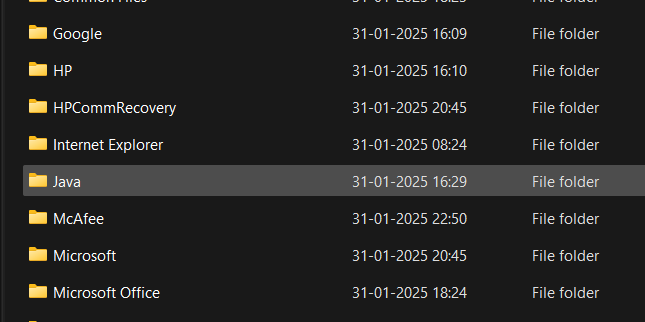
A screenshot of a computer

Description automatically generated

* Step 3 : After downloading open “this pc” in our laptop and open “program

files”,open “java”,open “JDK 21”



 A black screen with white numbers

Description automatically generated

A screenshot of a computer

Description automatically generated

* Step 4 : In the task bar search and open “environment variables of

system”,after opening environment variables, go to the system

variables and see for java if there leave it. Or click path and add

“JAVA” in ‘variable name’ and copy link in ‘variable value’

A screenshot of a computer

Description automatically generatedScreens screenshot of a computer

Description automatically generated

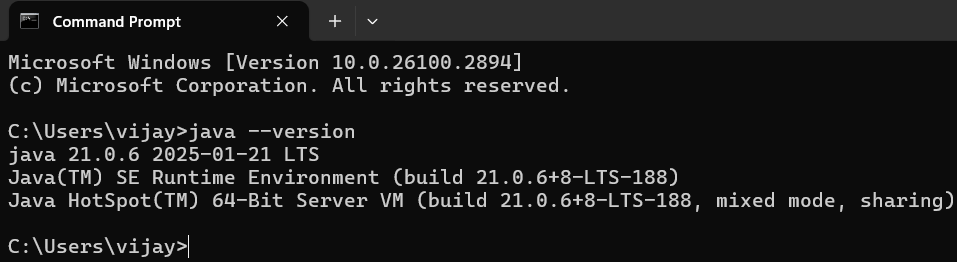
Screens screenshot of a computer

Description automatically generated

* Step 5 : Verifying Installation of Java. Again open task bar and search “cmd”,

open it ant type “java –version” and press enter. It will show the

version of installation of java.



Successfully Java is installed and it will show the version otherwise it will show error and command is not recognized.

* Program : 2

**Q) Write a java program to print the message “welcome to java program**”.

class Main{

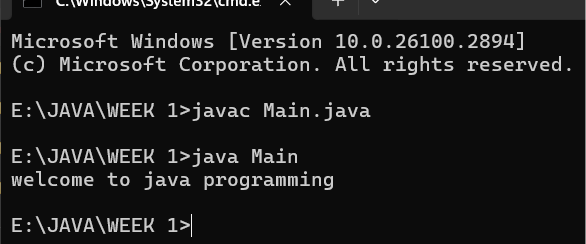
public static void main(String[] args){

System.out.println("welcome to java programming");

}

}

OUTPUT



* Program : 3

**Q) Write a java program that prints name,roll number,section of a student.**

public class My\_Profile

{

public static void main(String args[]) {

System.out.println("Name: A.Vijay Kumar");

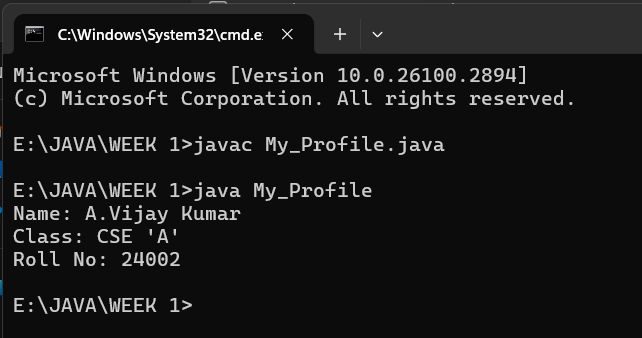
System.out.println("Class: CSE 'A'");

System.out.println("Roll No: 24002");

}

}

OUTPUT



WEEK-2

* Program : 1

**Q) Write a java program to calculate the area of the rectangle.**

**Program:**

import java.util.Scanner;

class rectangle{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the length");

int len=input.nextInt();

System.out.println("enter the bredth");

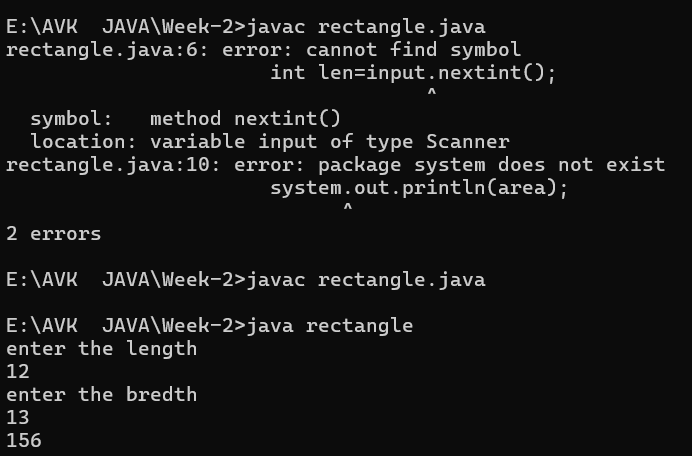
int bred=input.nextInt();

int area=len\*bred;

System.out.println(area);

}}

OUTPUT:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | Symbol method()input.nextint(); | input.nextInt(); |
| 2 | system.out.println(area); | System.out.println(area); |

* Program : 2

**Q) Write a java program to convert temperature from Celsius to Fahrenheit and viceversa.**

**Program:**

import java.util.Scanner;

class temperature{

public static void main(String[]args){

Scanner input =new Scanner(System.in);

System.out.println("enter the the temperature in degrees:");

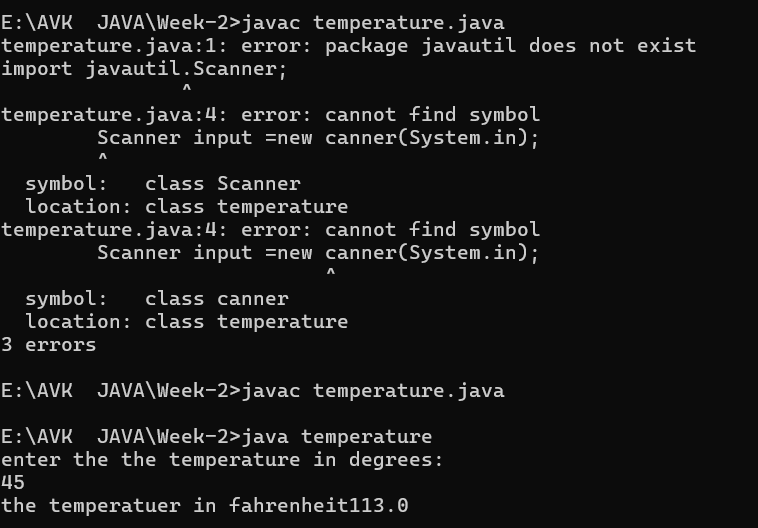
double deg=input.nextDouble();

System.out.println("the temperatuer in fahrenheit"+((deg\*9/5)+32));

}

}

OUTPUT:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | import javautil.Scanner; | import java.util.Scanner; |
| 2 | Scanner input =new scanner(System.in); | Scanner input =new Scanner(System.in); |

* Program : 3

**Q) Write a java program to calculate the simple intrest**.

**Program:**

import java.util.Scanner;

class simpleintrest{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the p value");

int p=input.nextInt();

System.out.println("enter the t value");

int t=input.nextInt();

System.out.println("enter the r value");

int r=input.nextInt();

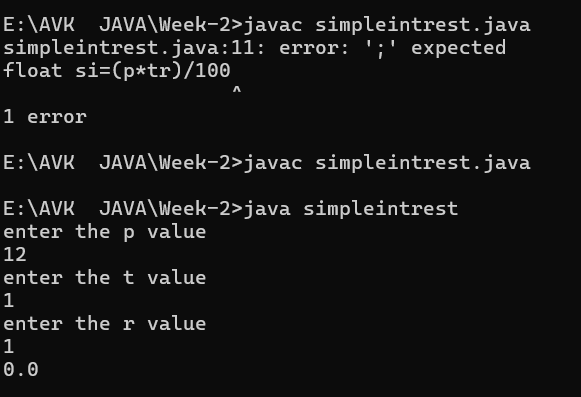
float si=(p\*t\*r)/100;

System.out.println(si);

}

}

OUTPUT:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | ; | ; is expected at end |
| 2 | float si=(p\*tr)/100; | float si=(p\*t\*r)/100; |

* Program : 4

**Q) Write a java program to find largest of three numbers using ternary**

**operator.**

**Program:**

import java.util.Scanner;

class largest{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter value of A");

int a=input.nextInt();

System.out.println("enter value of B");

int b=input.nextInt();

System.out.println("enter value of C");

int c=input.nextInt();

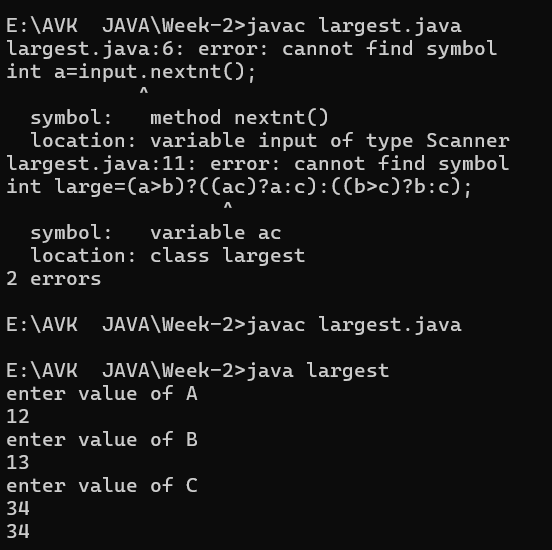
int large=(a>b)?((a>c)?a:c):((b>c)?b:c);

System.out.println(large);0

}

}

OUTPUT:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | int large=(a>b)?((ac)?a:c):((b>c)?b:c); | int large=(a>b)?((a>c)?a:c):((b>c)?b:c); |
| 2 | int a=input.nextint(); | int a=input.nextInt(); |

* Program : 5

**Q) Write a java program to find factorial of number.**

**Program:**

import java.util.Scanner;

class factorial{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the number to find its factorial");

int n=input.nextInt();

int sum=1;

for(int i=1;i<=n;i++){

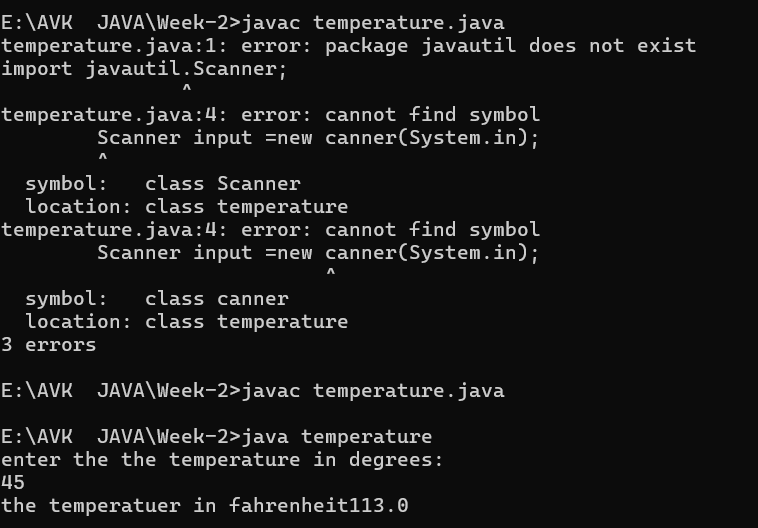
sum=sum\*i;}

System.out.println(sum);

}

}

OUTPUT



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | for(int i=1;i<=n;i+){ | for(int i=1;i<=n;i++){ |
| 2 | int =input.nextInt(); | int n=input.nextInt(); |

WEEK-3

* Program : 1

Q) **Write a java program with the following instructions**.

1. **Create a class with name car.**
2. **Create four attributes named car\_colour,car\_brand,fuel\_type,top\_speed.**
3. **Create three method named “Start\_Racing”,”End\_Race”.{ }**
4. **Create three objects named Car1,Car2,Car3.**
5. **Create a constructor which should print “Welcome to Garage”.**

**Class Diagram:**

|  |
| --- |
| **Car** |
| * carColor: String * carBrand: String * fuelType: String * topspeed: int |
| + Car(Car(carColour: String, carBrand:  String, fuelType: String, topSpeed: int)  + startRacing(): void  + endRace(): void |

cc

// Car.java

public class Car {

// Attributes

private String carColour;

private String carBrand;

private String fuelType;

private int topSpeed;

// Constructor

public Car(String carColour, String carBrand, String fuelType, int

topSpeed) {

this.carColour = carColour;

this.carBrand = carBrand;

this.fuelType = fuelType;

this.topSpeed = topSpeed;

System.out.println("Welcome to car garage");

}//End of the constructor

// Method to start racing

public void startRacing() {

System.out.println(carBrand + " (" + carColour + ") is starting the

race with a top speed of " + topSpeed + " km/h and runs on " +

fuelType + "!");

}//End of Method

// Method to end race

public void endRace() {

System.out.println(carBrand + " (" + carColour + ") has finished

the race!");

}//End of the Method

// Main method to create objects and demonstrate functionality

public static void main(String[] args) {

// Creating three objects

Car car1 = new Car("Red", "Ferrari", "Petrol", 200);

Car car2 = new Car("Blue", "Tesla", "Electric", 250);

Car car3 = new Car("Black", "BMW", "Diesel", 220);

// Starting and ending races

car1.startRacing();

car1.endRace();

car2.startRacing();

car2.endRace();

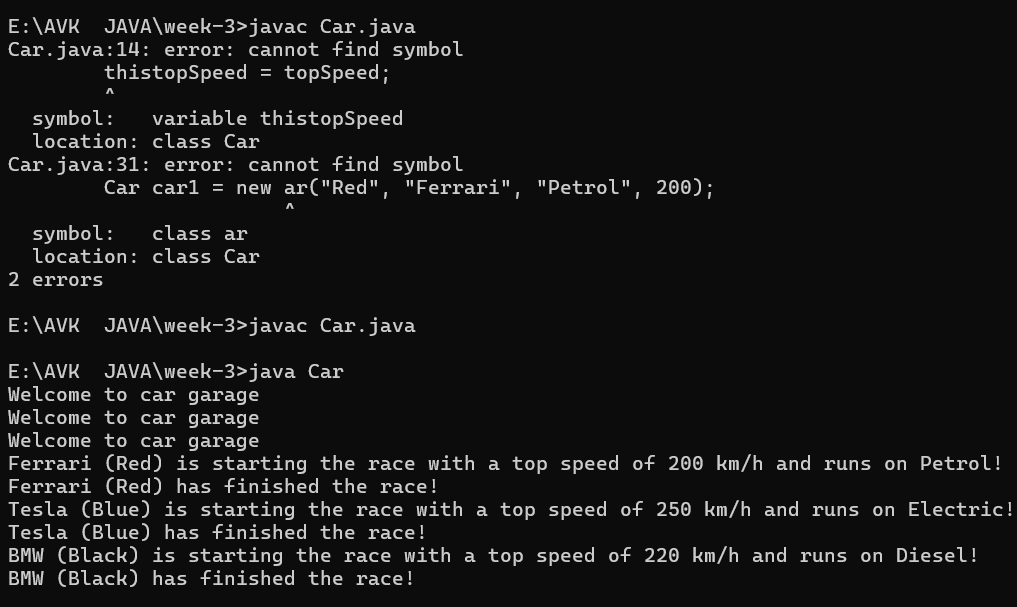
car3.startRacing();

car3.endRace();

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | Car car1 = new ar("Red", "Ferrari", "Petrol", 200); | Car car1 = new Car("Red", "Ferrari", "Petrol", 200); |
| 2 | thistopSpeed = topSpeed; | this.topSpeed = topSpeed; |

* Program : 2

**Q ) Write a class by writing java program named Bank Account**

**with two methods “deposits and withdraw”.**

**a) In deposit method whenever an amount is deposited it**

**has to be updated with current amount (logic C.A+D.A).**

**b) With draw amount whenever an amount is being**

**withdraw it has to be less than the current amount less**

**than the amount else print “Insufficient funds”.**

**Class Diagram:**

|  |
| --- |
| **Bank Account** |
| * currentAmount: double   + BankAccount(initialAmount:double) |
| + deposit(amount: double):void  + withdraw(amount: double):void  + getCurrentAmount():double |

**Program:**

import java.util.Scanner;

class BankAccount {

String name;

int accountNumber;

int currentBalance;

// Constructor to initialize the bank account

BankAccount(String name, int accountNumber, int currentBalance) {

this.name = name;

this.accountNumber = accountNumber;

this.currentBalance = currentBalance;

System.out.println("Customer Details: " + name + ", Account Number: " + accountNumber + ", Current Balance: " + currentBalance);

}

// Method to withdraw an amount

public void withdraw(int withdrawAmount) {

if (withdrawAmount <= currentBalance) {

currentBalance -= withdrawAmount;

System.out.println("Withdrawn: " + withdrawAmount);

System.out.println("Current Balance: " + currentBalance);

} else {

System.out.println("Insufficient Funds");

}

}

// Method to deposit an amount

public int deposit(int depositAmount) {

currentBalance += depositAmount;

System.out.println("Deposited: " + depositAmount);

return currentBalance;

}

// Main method to run the program

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input for account details

System.out.print("Enter your name: ");

String name = scanner.nextLine();

System.out.print("Enter your account number: ");

int accountNumber = scanner.nextInt();

System.out.print("Enter your initial balance: ");

int initialBalance = scanner.nextInt();

// Create a new bank account

BankAccount account = new BankAccount(name,

accountNumber, initialBalance);

// Input for withdrawal and deposit

System.out.print("Enter amount to withdraw: ");

int withdrawAmount = scanner.nextInt();

account.withdraw(withdrawAmount);

System.out.print("Enter amount to deposit: ");

int depositAmount = scanner.nextInt();

account.deposit(depositAmount);

// Final balance

System.out.println("Final Amount: " + account.currentBalance);

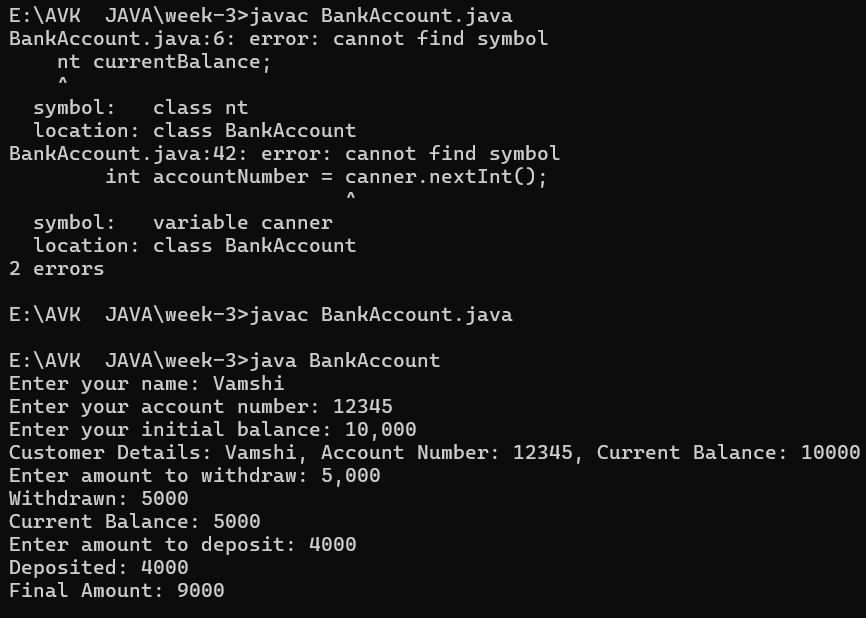
// Close the scanner

scanner.close();

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | int withdrawAmount = canner.nextInt(); | int withdrawAmount = scanner.nextInt(); |
| 2 | nt currentBalance; | int currentBalance; |

WEEK-4

* Program : 1

Q)  **Write a java program with class named “Book”. The class should contain**

**various attributes such as “Title of the book , author , year of publication “.**

**It should also contain a constructor with parameters which initializes**

**“ Title of the book, author, year of publication”. Create a method which displays the**

**details of the book. i.e. “ Title of the book, author and year of publication”. Display the details**

**of two books by creating two objects.**

**Class Diagram:**

|  |
| --- |
| **Book** |
| - title: String  - author: String  - yearOfPublication: int |
| + Book(title: String, author: String,  yearOfPublication: int)  + displayDetails(): void |

**Program:**

class Book {

// beginning of the class Book

public String title; // Changed Title to title for consistency

private String author;

public int yearOfPublication;

// beginning of constructor

Book(String title, String author, int yearOfPublication) {

this.title = title; // Changed Title to title for consistency

this.author = author;

this.yearOfPublication = yearOfPublication;

}

// constructor ends here

// method display starts here

public void display() {

System.out.println("Title of the book is: " + title +

", The name of the author is: " + author +

", The year of publication is: " + yearOfPublication);

}

// method display ends here

// creating objects

public static void main(String[] args) {

Book book1 = new Book("Harry Potter", "J.K. Rowling", 1993);

Book book2 = new Book("Someone Like You", "Nikitha Singh", 2010);

book1.display();

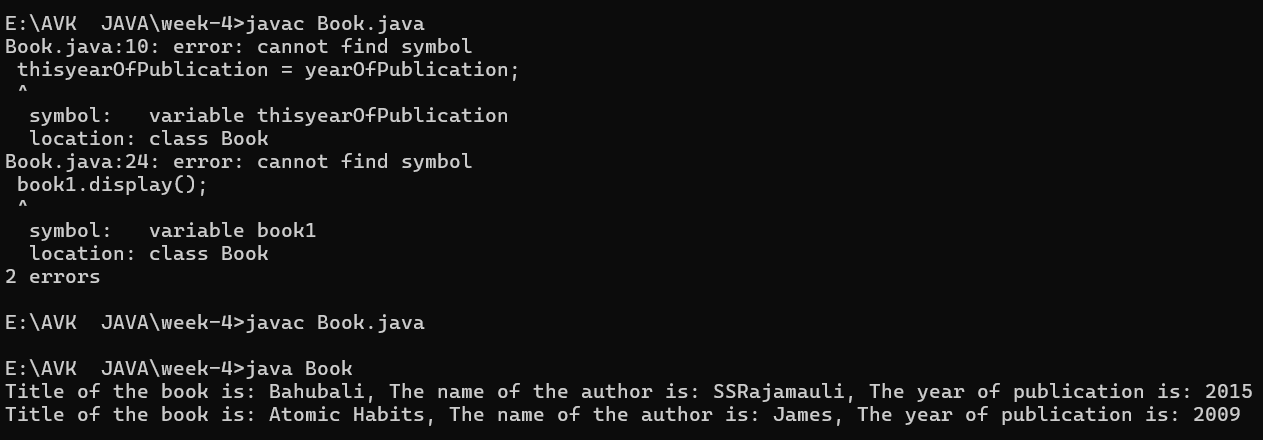
book2.display();

}

}

// class ends here

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | thisyearOfPublication = yearOfPublication;  } | this.yearOfPublication = yearOfPublication;  } |
| 2 | book.display(); | book1.display(); |

* Program : 2

Q)  **To create a java program with class named Myclass with a static**

**variable “Count” of “inttype”, Initialized to 0 and a constant variable**

**“pi” of type double , initialized to 3.1415 as attributes of that class Now,**

**define a constructor for “Myclass” that increments the “Count”**

**variable each that an object of Myclass is created. Finally , print the**

**final values of “Count” and “pi” variables .**

**Class Diagram:**

|  |
| --- |
| **MyClass** |
| - Count: int  + pi: double |
| + MyClass()  + getCount(): int |

**Program:**

class Myclass{

// class starts here

static int Count = 0;

final double pi = 3.1415;

// the constructor starts here

Myclass(){

Count++;

}

// the constructor ends here

public static void main(String[] args){

Myclass c1 = new Myclass();

Myclass c2 = new Myclass();

System.out.println("Count : " + c1.Count);

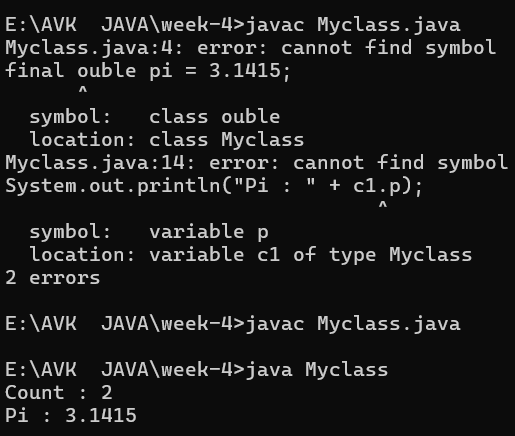
System.out.println("Pi : " + c1.pi);

}

}

// class ends here

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | final ouble pi = 3.1415; | final double pi = 3.1415; |
| 2 | System.out.println("Pi : " + c1.pi); | System.out.println("Pi : " + c1.pi); |

WEEK-5

* Program : 1

Q)  **Create a calculator using the operations including addition using**

**subtraction multiplication and division using multilateral inheritance**

**and display the desired output.**

**Class Diagram:**

|  |
| --- |
| **Basic Operations** |
| + add (a,b)  +subtract (a,b) |

|  |
| --- |
| **Multiplication** |
| +Multiply (a,b) |

|  |
| --- |
| **Division** |
| + Divide (a,b) |

|  |  |
| --- | --- |
| **Subtraction** | |
| + subtraction(a,b) | |
|  | |  | |

|  |
| --- |
| **Calculator** |
| +calculate (op,a,b) |

**Program:**

class bcalc {

int a, b;

int sum, diff;

bcalc(int a, int b) {

this.a = a;

this.b = b;

}

public void add() {

diff = a - b;

sum = a + b;

System.out.println("Difference: " + diff);

System.out.println("Sum: " + sum);

}

}

class acalc extends bcalc {

int mul;

acalc(int a, int b) {

super(a, b);

}

public void mult() {

mul = a \* b;

System.out.println("Multiplication: " + mul); 29

}

}

class aacalc extends acalc {

f loat div;

aacalc(int a, int b) {

super(a, b);

}

public void divi() {

if (b != 0) { // Check to avoid division by zero

div = (float) a / b;

System.out.println("Division: " + div);

}

else {

System.out.println("Division by zero error!");

}

}

}

class ocalc {

public static void main(String[] args) {

aacalc c = new aacalc(10, 2);

c.divi();

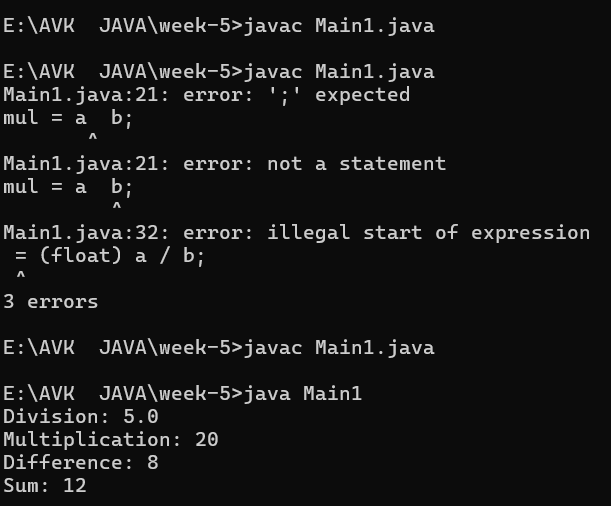
c.mult();

c.add();

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | mul = a b; | mul = a \* b; |
| 2 | = (float) a / b; | div = (float) a / b; |

* Program : 2

Q)  **A Vechile rental company wants to develop a system ,that maintains**

**information about different types of vehicles available for rent.The**

**company rents out cars and bikes and they a need a program to**

**store details about each vehicle such as brand and speed .**

* **Cars should have an additional properties .**
* **“Number of doors “ seating capacity.**
* **Bikes should have a property indicating whether they have gears are not ?**
* **The system should also include a fuction to display details about each vehicle and indicate when a vechicle is starting .**
* **If the company describes to add a new type of vechile ‘truck’ how would you modify above program.**
* **Truck should include an addition property capacity ‘in tons’.**
* **Create a show truck details method to display the trucks capacity.**
* **Write a constructor for truck that initializes all properties.**
* **Implement the truck class and update the main method to create a truck object and also create an object and also create an object car and bike subclass find display it details.**

**Class Diagram:**

|  |
| --- |
| Vechile |
| * Brand: String * speed: int |
| + Vechile(String,b int)  + Start()  + DisplayDetails() |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | Car | | * numberofdoors: int * seatingCapacity: int | | + car(String,int,int,int)  + displaydetails() | | |  | | --- | | Bike | | -hasGears: boolean | | + Bike(String,int,Boolean)  +displayetails() | |

|  |
| --- |
| Truck |
| -capacity: double |
| + truck(String,int,double)  +showtruckdetails()  +displaydetails() |

**Program:**

// Base class for Vehicle

class Vehicle {

protected String brand;

protected int speed;

public Vehicle(String brand, int speed) {

this.brand = brand;

this.speed = speed;

}

public void start() {

System.out.println(brand + " is starting.");

}

public void displayDetails() {

System.out.println("Brand: " + brand);

System.out.println("Speed: " + speed + " km/h");

}

}

// Car class that extends Vehicle

class Car extends Vehicle {

private int numberOfDoors;

private int seatingCapacity;

public Car(String brand, int speed, int numberOfDoors, int seatingCapacity) {

super(brand, speed);

this.numberOfDoors = numberOfDoors;

this.seatingCapacity = seatingCapacity;

}

@Override

public void displayDetails() {

super.displayDetails();

System.out.println("Number of Doors: " + numberOfDoors);

System.out.println("Seating Capacity: " + seatingCapacity);

}

}

// Bike class that extends Vehicle

class Bike extends Vehicle {

private boolean hasGears;

public Bike(String brand, int speed, boolean hasGears) {

super(brand, speed);

this.hasGears = hasGears;

}

@Override

public void displayDetails() {

super.displayDetails();

System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));

}

}

// Truck class that extends Vehicle

class Truck extends Vehicle {

private double capacity; // in tons

public Truck(String brand, int speed, double capacity) {

super(brand, speed);

this.capacity = capacity;

}

public void showTruckDetails() {

System.out.println("Truck Capacity: " + capacity + " tons");

}

@Override

public void displayDetails() {

super.displayDetails();

showTruckDetails();

}

}

// Main class to test the implementation

public class Main {

public static void main(String[] args) {

// Create a Car object

Car car = new Car("Toyota", 180, 4, 5);

car.start();

car.displayDetails();

System.out.println();

// Create a Bike object

Bike bike = new Bike("Yamaha", 120, true);

bike.start();

bike.displayDetails();

System.out.println();

// Create a Truck object

Truck truck = new Truck("Volvo", 100, 10.5);

truck.start();

truck.displayDetails();

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | super(, speed); | super(brand, speed); |
| 2 | public Car(String brand, nt speed, int numberOfDoors, int seatingCapacity) { | public Car(String brand, int speed, int numberOfDoors, int seatingCapacity) { |

WEEK-6

* Program : 1

**Q)Write a java program to create a vehicle class with a method override**

**this method in the class subclass to provide. Specific information about**

**the cars that is “car company , model price ,seating capacity, petrol or**

**diesel, true or false”**

**Class Diagram:**

|  |
| --- |
| Vehicle |
|  |
| * displayInfo() |

|  |
| --- |
| Car |
| - carCompany: String  - model: String  - price: double  - seatingCapacity: int  - fuelType: String  - isAvailable: boolean |
| + displayInfo()   + Car(company: String, model: String, price: double, seatingCapacity: int, fuelType: String, isAvailable: boolean) |

**Program:**

// Base class

class Vehicle {

// Method to be overridden

public void displayInfo() {

System.out.println("This is a vehicle.");

}

}

// Subclass

class Car extends Vehicle {

private String company;

private String model;

private double price;

private int seatingCapacity;

private String fuelType; // "Petrol" or "Diesel"

private boolean isAvailable; // true or false

// Constructor

public Car(String company, String model, double price, int seatingCapacity, String fuelType, boolean isAvailable) {

this.company = company;

this.model = model;

this.price = price;

this.seatingCapacity = seatingCapacity;

this.fuelType = fuelType;

this.isAvailable = isAvailable;

}

// Overriding the displayInfo method

@Override

public void displayInfo() {

System.out.println("Car Company: " + company);

System.out.println("Model: " + model);

System.out.println("Price: $" + price);

System.out.println("Seating Capacity: " + seatingCapacity);

System.out.println("Fuel Type: " + fuelType);

System.out.println("Available: " + (isAvailable ? "Yes" : "No"));

}

}

// Main class to test the functionality

public class VehicleMain {

public static void main(String[] args) {

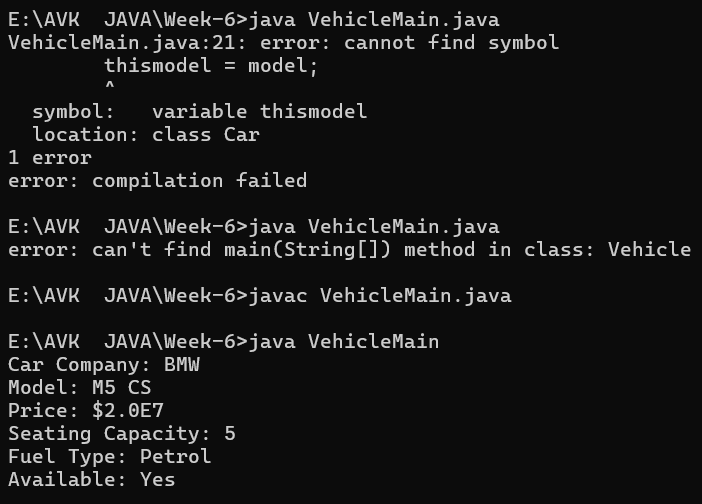
Vehicle myCar = new Car("BMW", "M5 CS", 20000000, 5, "Petrol", true);

myCar.displayInfo();

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | thismodel = model; | this.model = model; |
| 2 | System.out.println(" " + model); | System.out.println(" Model" + model); |

* Program : 2

Q)**Write a java program that a Collage is developing a automated**

**admission system that verifies student eligibility for UG and PG**

**programs each program has a different eligibility criteria based on the**

**students percentage in the previous qualification.**

* **UG admission require minimum 60%**
* **PG admission require minimum 70%**

**Class Diagram:**

|  |
| --- |
| Admission |
| +eligibility():void |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | UG | | + eligible(): void | | |  | | --- | | PG | | + eligible(): void | |

**Program:**

import java.util.Scanner;

class College {

String name;

int qualification;

int percentage;

// Constructor

College(String name, int qualification, int percentage) {

this.name = name;

this.qualification = qualification;

this.percentage = percentage;

}

// Default Eligibility method

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is a fluke");

}

}

class UG extends College {

UG(String name, int qualification, int percentage) {

super(name, qualification, percentage);

}

@Override

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is eligible for UG");

}

}

class PG extends College {

PG(String name, int qualification, int percentage) {

super(name, qualification, percentage);

}

@Override

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is eligible for PG");

}

}

public class CollageMain {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Taking inputs

System.out.println("Enter your name:");

String name = input.nextLine();

System.out.println("Enter your qualification (e.g., 12 for high school, 10 for 10th, etc.):");

int qualification = input.nextInt();

System.out.println("Enter your percentage:");

int percentage = input.nextInt();

// Close scanner

input.close();

// Logic to check eligibility

College candidate;

if (percentage >= 70) {

candidate = new PG(name, qualification, percentage);

} else if (percentage >= 60) {

candidate = new UG(name, qualification, percentage);

} else {

candidate = new College(name, qualification, percentage);

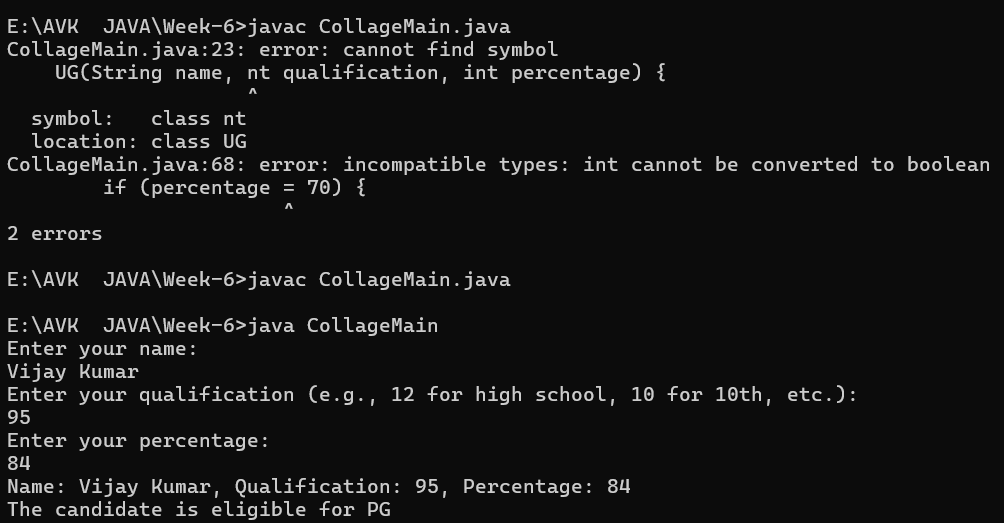
}

candidate.Eligibility();``

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | if (percentage = 70) { | if (percentage > = 70) { |
| 2 | UG(String name, nt qualification, int percentage) { | UG(String name, int qualification, int percentage) { |

* Program : 3

Q)**Create a calculator class with overloading methods to perform addition.**

* **Add two doubles**
* **Add two doubles**
* **Add three Integers**

**Class Diagram:**

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

**Program:**

public class Calculator{

// Method to add two integers

public int add(int a, int b) {

return a + b;

}

// Method to add two tuples (represented as arrays)

public int add(int[] tuple1, int[] tuple2) {

int sum = 0;

for (int i = 0; i < tuple1.length; i++) {

sum += tuple1[i] + tuple2[i];

}

return sum;

}

// Method to add three integers

public int add(int a, int b, int c) {

return a + b + c;

}

public static void main(String[] args) {

Calculator calc = new Calculator();

// Adding two integers

int result1 = calc.add(10, 20);

System.out.println("Addition of two integers: " + result1);

// Adding two tuples (arrays)

int[] tuple1 = {1, 2};

int[] tuple2 = {3, 4};

int result2 = calc.add(tuple1, tuple2);

System.out.println("Addition of two tuples: " + result2);

// Adding three integers

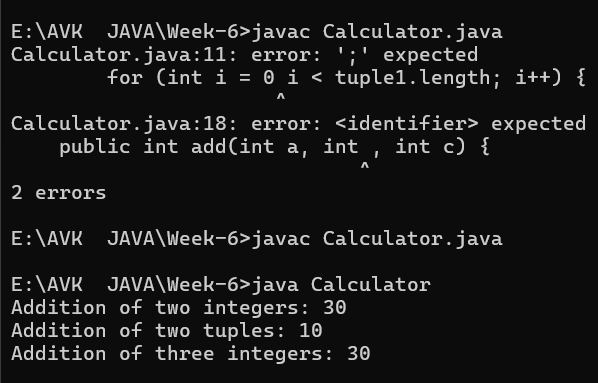
int result3 = calc.add(5, 10, 15);

System.out.println("Addition of three integers: " + result3);

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | for (int i = 0 i < tuple1.length; i++) { | for (int i = 0; i < tuple1.length; i++) { |
| 2 | public int add(int a, int , int c) { | public int add(int a, int b, int c) { |

* Program : 4

Q)**Write a java program create a shape class with a method calculate area**

**that is overloaded for different shapes Square,Rectangle then create a**

**sub class circle that overerides the calculate area methods for a circle.**

**Class Diagram:**

|  |
| --- |
| Shape |
| +calarea(float side):float  +calarea(float l,float b):float  +calarea(float c):float |

|  |
| --- |
| Circle |
| +calarea(double r):double |

**Program:**

public class Shape {

public double areaOfSquare(double side) {

return side \* side;

}

public double areaOfRectangle(double length, double width) {

return length \* width;

}

public double areaOfCircle(double radius) {

return 3.14 \* radius \* radius;

}

public static void main(String[] args) {

Shape shape = new Shape();

double square = shape.areaOfSquare(5);

System.out.println("Area of a square: " + square);

double rectangle = shape.areaOfRectangle(10, 20);

System.out.println("Area of a rectangle: " + rectangle);

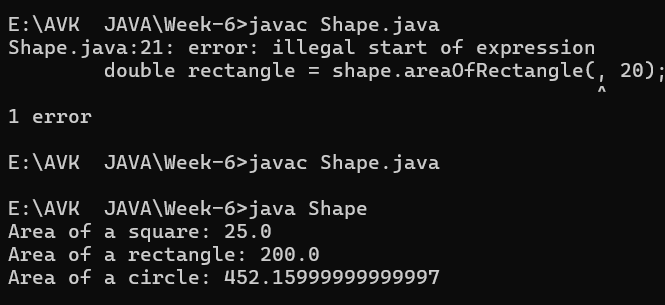
double circle = shape.areaOfCircle(12);

System.out.println("Area of a circle: " + circle);

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | Shape shape = Shape(); | Shape shape = new Shape(); |
| 2 | double rectangle = shape.areaOfRectangle(, 20); | double rectangle = shape.areaOfRectangle(10, 20); |

WEEK-6

* Program : 1

**Q) Write a Java program to create an abstract class Animal with an**

**abstract method called sound(). Create subclasses Lion and Tiger that**

**extend the Animal class and implement the sound() method to make a**

**specific sound for each animal**.

**Class Diagram:**

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | Lion | | + sound() | | |  | | --- | | Tiger | | + sound() | |

**Program:**

// Superclass

class Animal {

// Method to be overridden by subclasses

public String eat() {

return "This animal eats food.";

}

}

// Subclass for Herbivores

class Herbivores extends Animal {

@Override

public String eat() {

return "This herbivore eats plants.";

}

}

// Subclass for Carnivores

class Carnivores extends Animal {

@Override

public String eat() {

return "This carnivore eats meat.";

}

}

// Subclass for Omnivores

class Omnivores extends Animal {

@Override

public String eat() {

return "This omnivore eats both plants and meat.";

}

}

// Main class renamed to AnimalTest

public class AnimalHouse {

public static void main(String[] args) {

// Create instances of each subclass

Animal herbivore = new Herbivores();

Animal carnivore = new Carnivores();

Animal omnivore = new Omnivores();

// Call the eat() method for each animal

System.out.println(herbivore.eat());

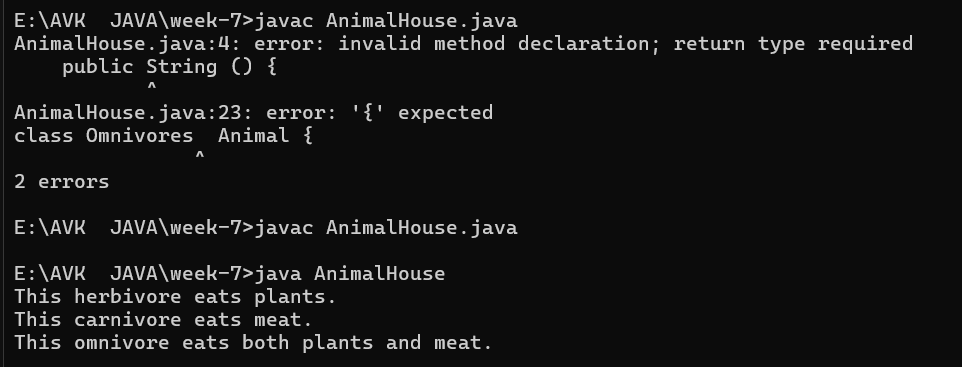
System.out.println(carnivore.eat());

System.out.println(omnivore.eat());

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | public String () { | public String eat() { |
| 2 | class extends Animal { | class Omnivores extends Animal { |

* Program : 2

**Q) 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.**

**Class Diagram:**

|  |
| --- |
| Shape3D |
| + calculateVolume()  +calculateSurfaceArea() |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | Sphere | | - radius  + calculateVolume()  +calculateSurfaceArea() | | |  | | --- | | Cube | | - side | |

**Program:**

abstract class Shape3D {

    // Abstract method to calculate volume

    public abstract double calculateVolume();

    // Abstract method to calculate surface area

    public abstract double calculateSurfaceArea();

}

// Subclass for Sphere

class Sphere extends Shape3D {

    private double radius;

    // Constructor

    public Sphere(double radius) {

        this.radius = radius;

    }

    @Override

    public double calculateVolume() {

        return (4.0 / 3.0) \* Math.PI \* Math.pow(radius, 3);

    }

    @Override

    public double calculateSurfaceArea() {

        return 4 \* Math.PI \* Math.pow(radius, 2);

    }

}

// Subclass for Cube

class Cube extends Shape3D {

    private double side;

    // Constructor

    public Cube(double side) {

        this.side = side;

    }

    @Override

    public double calculateVolume() {

        return Math.pow(side, 3);

    }

    @Override

    public double calculateSurfaceArea() {

        return 6 \* Math.pow(side, 2);

    }

}

// Main class to test the implementation

public class Shape3DCheck {

public static void main(String[] args) {

Shape3D sphere = new Sphere(5);

Shape3D cube = new Cube(3);

// Output without formatting to 2 decimal places

System.out.println("Sphere Volume: " + sphere.calculateVolume());

System.out.println("Sphere Surface Area: " +

sphere.calculateSurfaceArea());

System.out.println("Cube Volume: " + cube.calculateVolume());

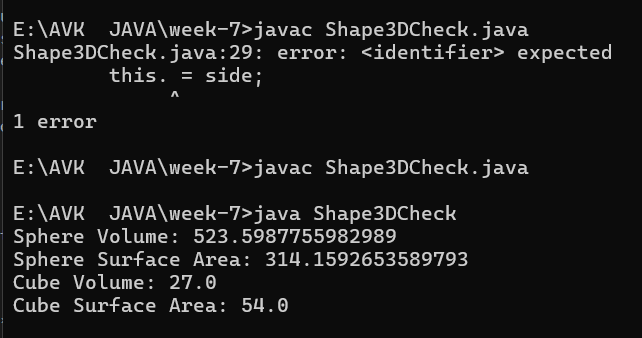
System.out.println("Cube Surface Area: " +

cube.calculateSurfaceArea());

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | this. = side; | this.side = side; |

* Program : 3

**Q) write a java program using an abstract class to define a method for pettern**

**printing create an abstract class named pattern prints with an abstract method**

**print pattern and a converts methods to display the pattern title.**

* **Implement the subclass**

**1.Star pattern prints a right angled triangle of class() number pattern –**

**prints a right angle triangle with increasing number**

**2.In the main method create object of both subclass and print a pattern for**

**the given no of rows.**

**Class Diagram:**

|  |
| --- |
| PatternPrints |
| +printTitle(title)  +printPattern(rows) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | StarPattern | | +printPattern() | | |  | | --- | | NumberPattern | | +printPattern() | |

**Program:**

// Abstract class

abstract class PatternPrints {

// Abstract method for printing the pattern

public abstract void printPattern(int rows);

// Concrete method to display the title of the pattern

public void printTitle(String title) {

System.out.println("=== " + title + " ===");

}

}

// Subclass 1: Star pattern (right-angled triangle)

class StarPattern extends PatternPrints {

@Override

public void printPattern(int rows) {

for (int i = 1; i <= rows; i++) {

for (int j = 1; j <= i; j++) {

System.out.print("\* ");

}

System.out.println();

}

}

}

// Subclass 2: Number pattern (right-angled triangle with increasing numbers)

class NumberPattern extends PatternPrints {

@Override

public void printPattern(int rows) {

int num = 1;

for (int i = 1; i <= rows; i++) {

for (int j = 1; j <= i; j++) {

System.out.print(num++ + " ");

}

System.out.println();

}

}

}

// Main class

public class PatternClock{

public static void main(String[] args) {

int numberOfRows = 5;

PatternPrints starPattern = new StarPattern();

PatternPrints numberPattern = new NumberPattern();

starPattern.printTitle("Star Pattern");

starPattern.printPattern(numberOfRows);

System.out.println();

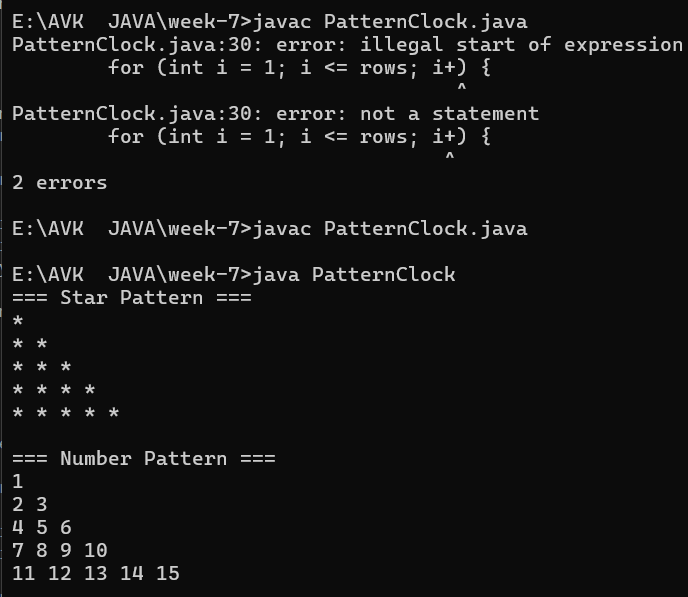
numberPattern.printTitle("Number Pattern");

numberPattern.printPattern(numberOfRows);

}

}

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | public void printPattern(int row) { | public void printPattern(int rows) { |
| 2 | for (int i = 1; i <= rows; i++) { | for (int i = 1; i <= rows; i+) { |