23CSE111

OBJECT ORIENTED PROGRAMMING

LAB REPORT



## 

## Department of Computer Science Engineering Amrita School of Computing

## Amrita Vishwa Vidyapeetham, Amaravati Campus

**Name: PVS KRISHNAKANTH**

**Roll No: AV.SC.U424304**

**SECTION:CSE-B**

**Verified By :**

**Index**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Title** | **Pgno** |
| **1** | **Week-1** | **3-7** |
| **2** | **Week-2** | **8-21** |
| **3** | **Week-3** | **22-29** |
| **4** | **Week-4** | **30-37** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Experiment** | **Page No** |
| **1** | **Installation of jdk21** | **4-5** |
| **2** | **Write a simple java program for printing Name, Class, Roll No, of a Student** | **6-7** |

**INDEX**

Week 1

**WEEK-1**

1. **Process of Installing JDK (Java Development Kit)**

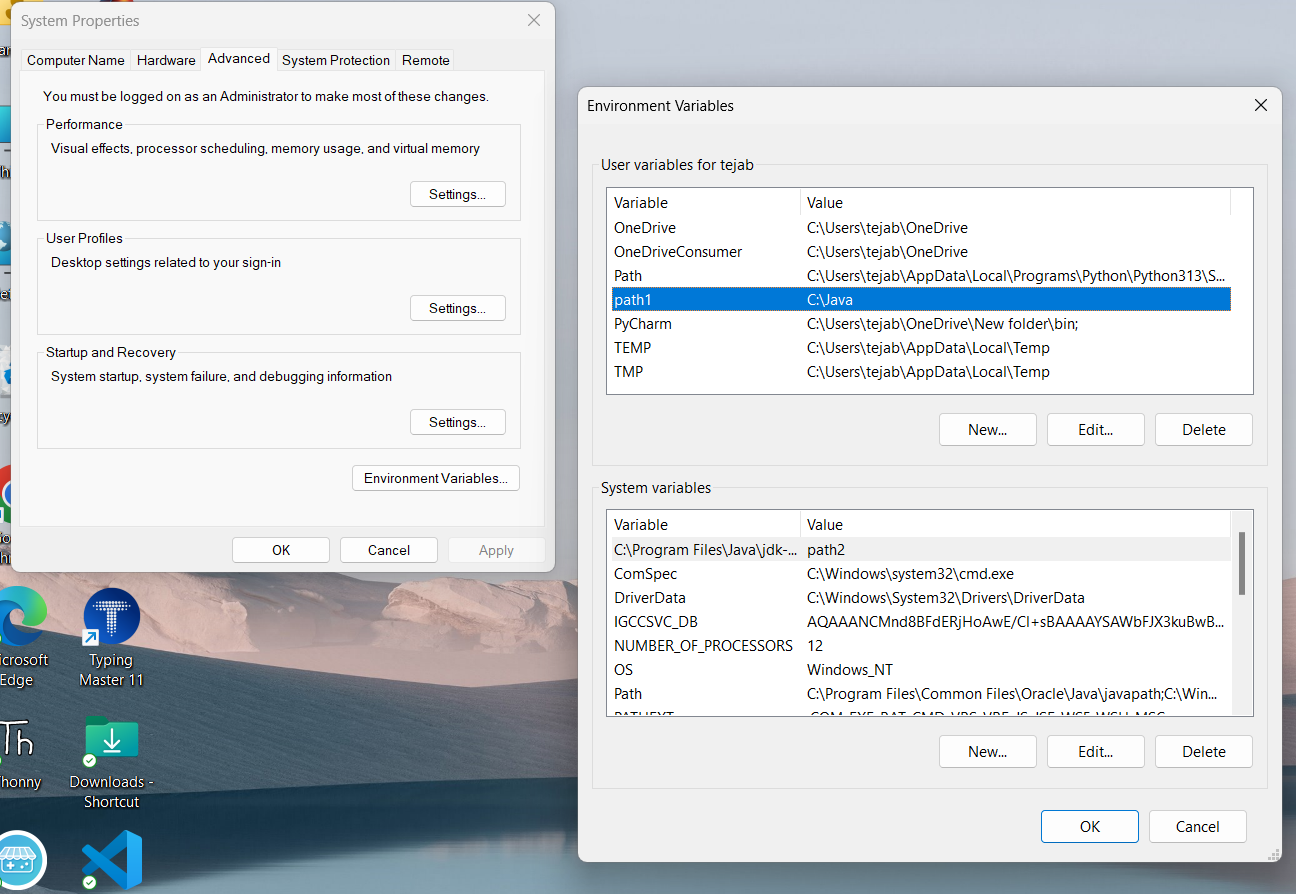
**Installing JDK (Java Development Kit):**

* 1. **Download JDK:**
* Go to the Oracle JDK download page in google and click on JDK-21 version which is Long term support (LTS) version.
* Click the download link as your operating system (Windows, macOS, or Linux).
  1. **Install JDK:**
* Once downloaded, run the installer.
* Follow the given instructions and keep clicking "Next" until it is done.
  1. **Set Environment Variables (Windows):**
* Open file explorer, then right click on This PC next select on properties then it will take you to the settings app then click on advanced system settings and then click on **Environment Variables**.
* Click on path and new under **System Variables**:

**Variable value:** The folder address where JDK is installed (like

C:\Program Files\Java\jdk-21\bin)

* Find Path under **System Variables**, click **New**, and add the path of the jdk-21(C:\Program Files\Java\jdk-21\bin)



**Checking JDK Version: -**

* 1. **Open Command Prompt:**
* Presswin+R, typecmd, and press Enter.
  1. **Check Version:**
* Type java -version and press Enter.
* Type javac --version and press Enter.



1. **Simple Java Program for printing Name, Class, Roll No, of a Student**

Aim:

Write your code in Notepad and execute it in cmd prompt

**CODE: -**

class Main

{

public static void main(String[] args)

{

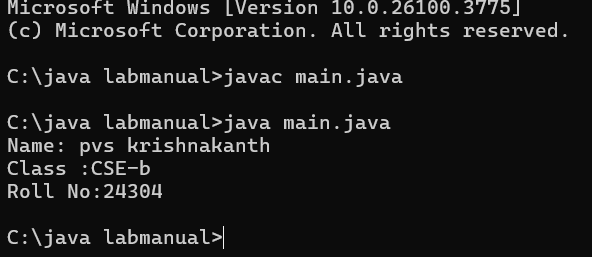
System.out.println("Name: pvs krishnakanth");

System.out.println("Class :CSE-b");

System.out.println("Roll No:24304");

}

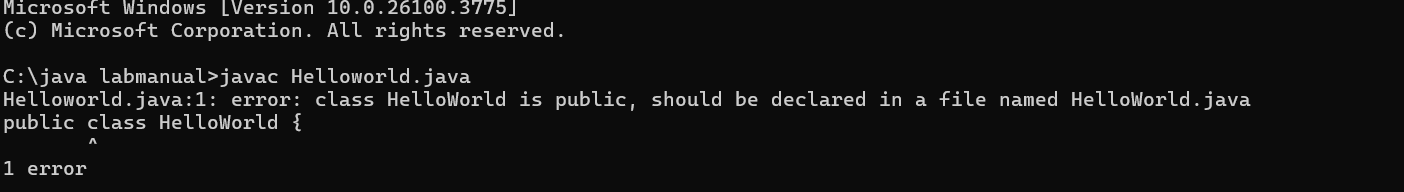
}**Output: -**

****

Errors

|  |  |  |
| --- | --- | --- |
| 1 | Syntax error | Semicolon added |
| 2 | Runtime error | Copied correct path |
| 3 | Name error | rectified |

Negative Case:



Week-2

|  |  |  |
| --- | --- | --- |
| S.No | Title | Pg no |
| 1 | Write a java program to find simple interest where all inputs are taken from user | 9-10 |
| 2 | Write a java program to calculate factorial of a number , read the input from user | 10-12 |
| 3 | Write a java program to calculate the Fibonacci  Sequence of a input taken from user | 12-14 |
| 4  A | Write a java program to convert temperature from Celsius to Fahrenheit | 14-15 |
| 4  B | Write a java program to convert temperature from Fahrenheit to Celsius | 16-17 |
| 5 | Write a java program to calculate the area of rectangle | 17-19 |
| 6 | Write a java program to calculate the area of triangle by using heron’s formula | 19-21 |

1. **Write a Simple Java Program for finding simple interest by taking input**

Simport java.util.\*;

class Simple {

public static void main(String args[]) {

float SI;

Scanner sc = new Scanner(System.in);

System.out.println("Enter principal value:");

int P = sc.nextInt();

System.out.println("Enter time:");

int T = sc.nextInt();

System.out.println("Enter rate of interest:");

float R = sc.nextFloat();

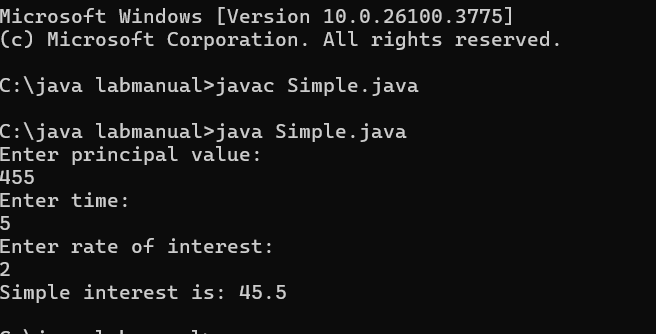
SI = (P \* T \* R) / 100;

System.out.println("Simple interest is: " + SI);

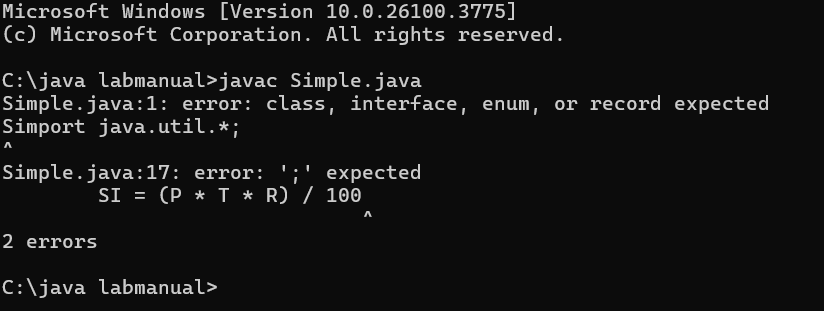
}

}

**Output:**

****

**Negative Case:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Runtime error** | **Incorrect path** | **Copied correct path** |
| **2** | **Syntax error** | **{ missing** | **{ added** |
| **3** | **Logical error** | **Wrong formula** | **Formula rectified** |

**2.Aim: Write a simple java program to calculate factorial of a number and read the**

**input from user**

**code:**

**class fac{**

**public static void main(String[] args){**

**int fac=1;**

**int n=10;**

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

**fac\*=I;**

**}**

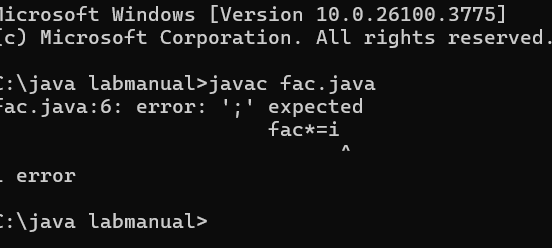
**System.out.println("fac of first 10 natural numbers="+fac);**

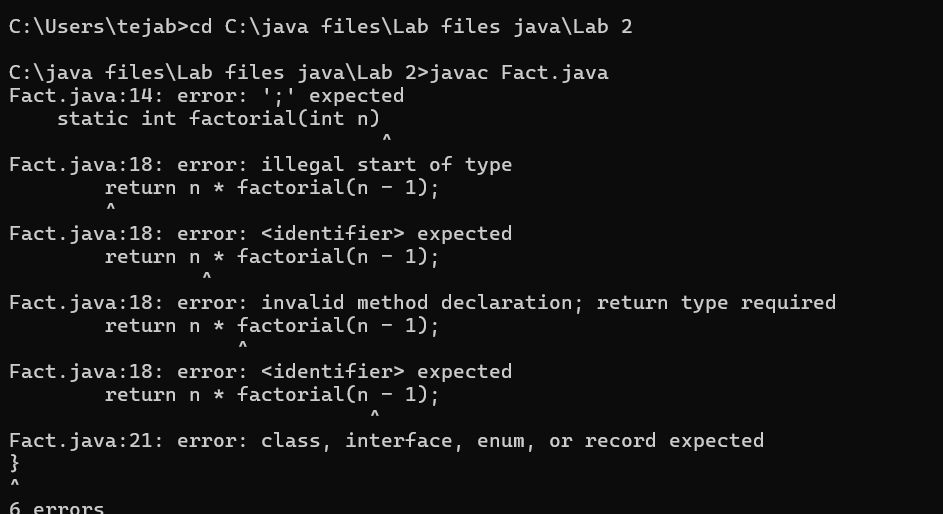
**}**

**}**

**Output:**

**Negative case:**

****

****

**Error table**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Undeclared variable error** | **Missing variable** | **Variable declared** |
| **2** | **Missing import statement** | **Not importing packages** | **Packages imported** |
| **3** | **Logical error** | **Wrong formula** | **Formula rectified** |

**3.**

**Aim: Write a program to to calculate the fibonacii sequence and take the input**

**from user**

**Code:**

**import java.util.\*;**

**class fibo**

**{**

**public static void main(String args[])**

**{**

**Scanner sc = new Scanner(System.in);**

**int num;**

**int f3;**

**int f1 = 0;**

**int f2 = 1;**

**int i = 2;**

**System.out.print("Enter a number:");**

**num = sc.nextInt();**

**System.out.println(f1);**

**System.out.println(f2);**

**while(i<num)**

**{**

**f3 = f1+f2;**

**f1 = f2;**

**f2 = f3;**

**System.out.println(f3);**

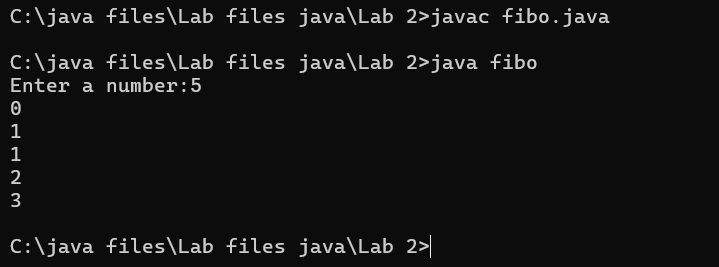
**i = i+1;**

**}**

**}**

**}**

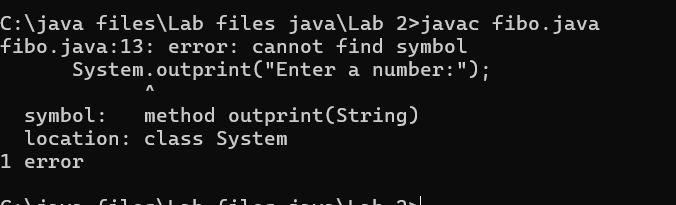
**Output:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Logical error** | **Incorrect formula** | **Formula rectified** |
| **2** | **Run-time error** | **Incorrect path** | **Added correct path** |
| **3** |  |  |  |

**Negative case:**

****

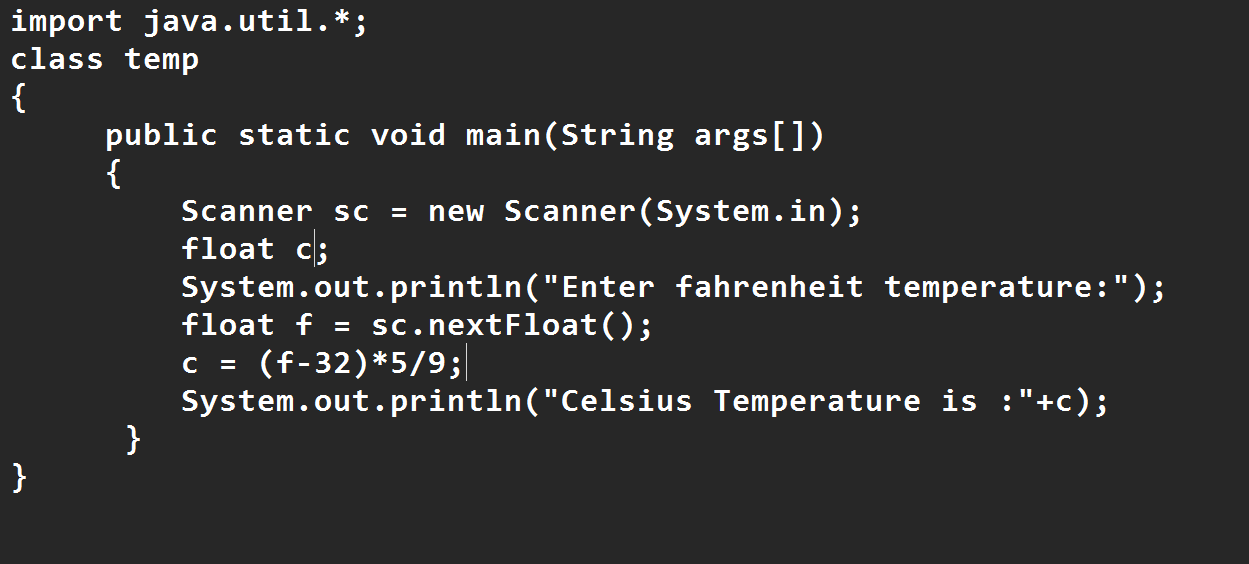
**Important points**

**Here the assignment operartion takes makes values are keep on updated for f1 and f2**

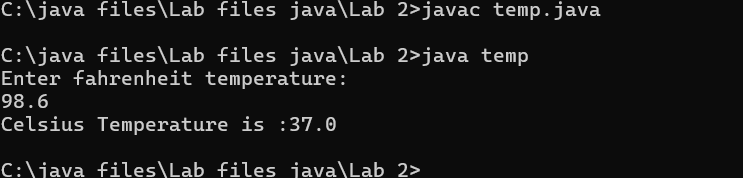
**4.A)**

**Aim: Write a java program to convert temperature from Fahrenheit to celsius**

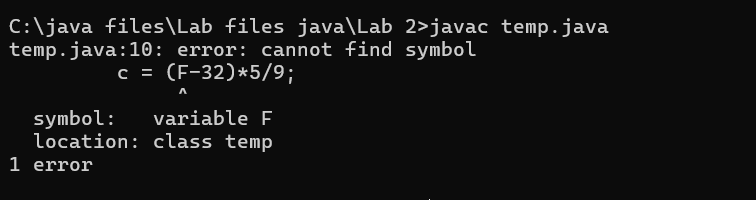
**Code:**

****

**Output:**

****

**Negative Case:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Syntax error** | **Missing ”** | **“ is added** |
| **2** | **Missing import error** | **Util package missing** | **Util package added** |
| **3** | **Logical error** | **Incorrect formula** | **Formula rectified** |

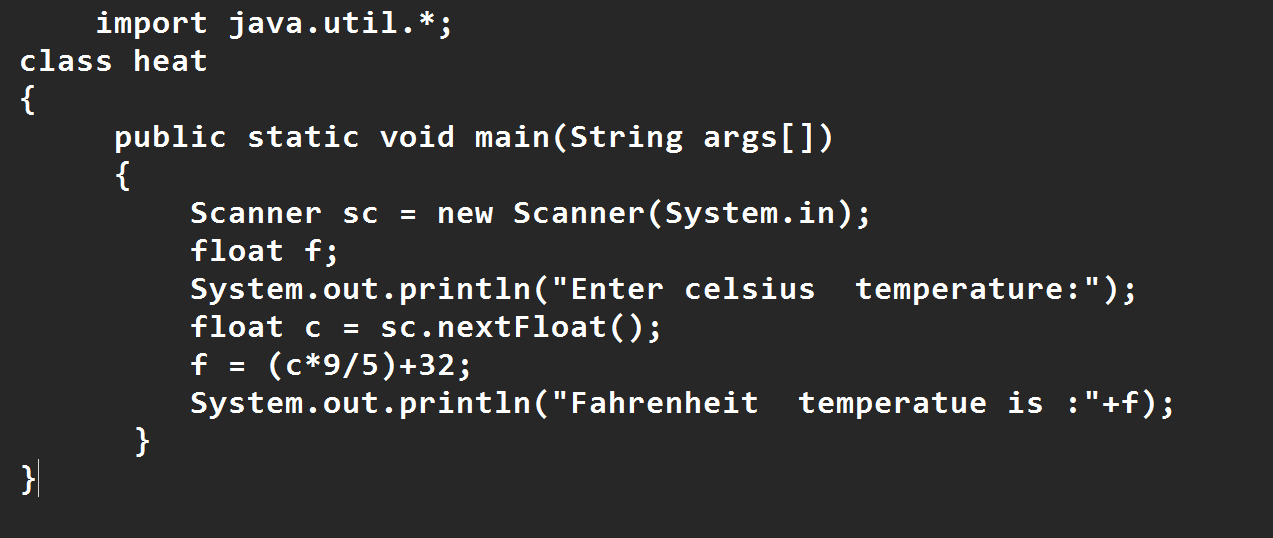
**Important Points**

**Conversion of Fahrenheit to Celsius is c = (f-32)\*5/9**

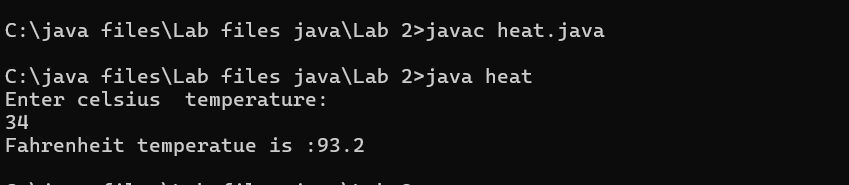
**4.b**

**Aim: Write a java program to convert temperature from Celsius to Fahrenheit**

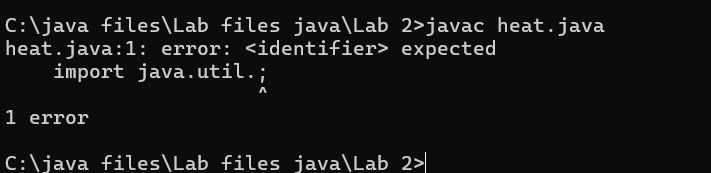
**Code:**

****

**Output:**

****

**Negative Case:**

****

**Important points**

**Conversion of celsius to Fahrenheit is (c\*9/5)+32**

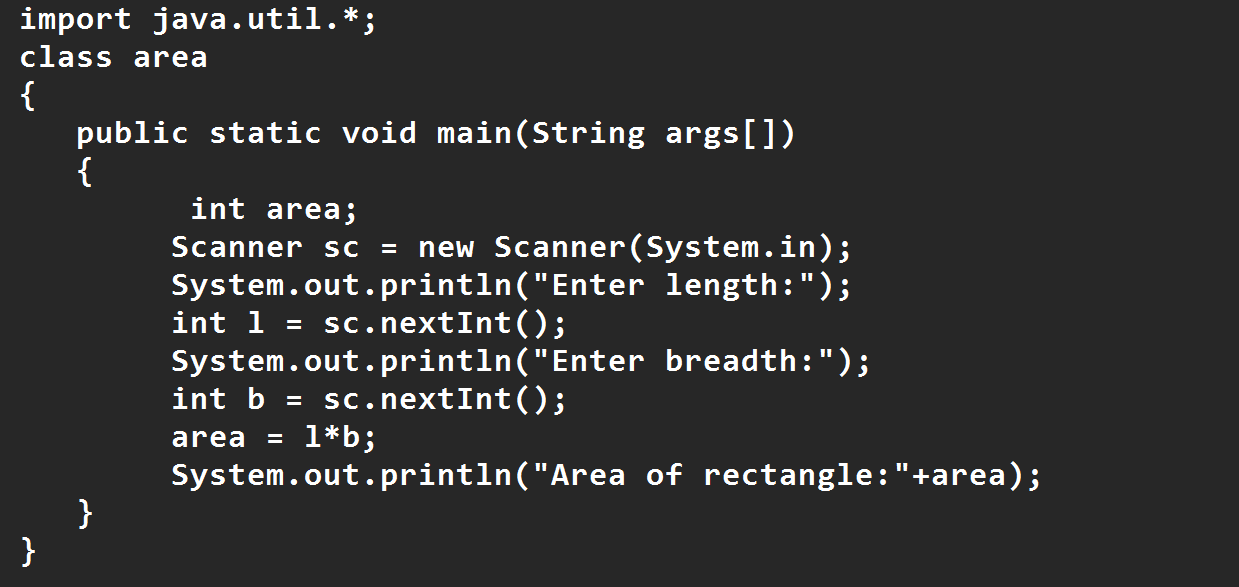
**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Runtime error** | **Incorrect path selection** | **Correct path added** |
| **2** | **Logical error** | **Incorrect formula** | **Correct formula**  **rectified** |
| **3** | **Import package error** | **Incorrect importing of**  **Packages** | **Imported util.\*;**  **Package** |

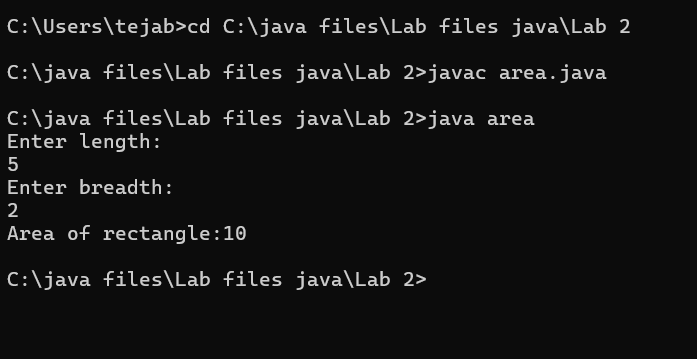
**5.**

**Aim: Write a simple java program to find the area of rectangle:**

**Code:**

****

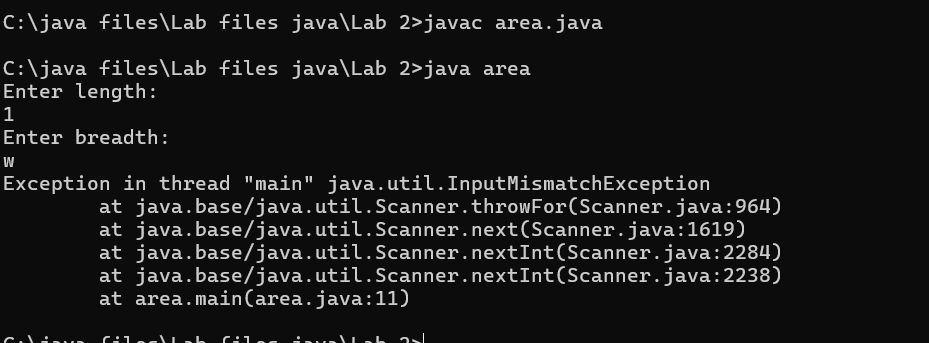
**Output:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Syntax error** | **Semi colon missing** | **Semi colon added** |
| **2** | **Missing import error** | **Import package missing** | **Import package added** |
| **3** | **Runtime error** | **Incorrect path selection** | **Rectified correct path** |

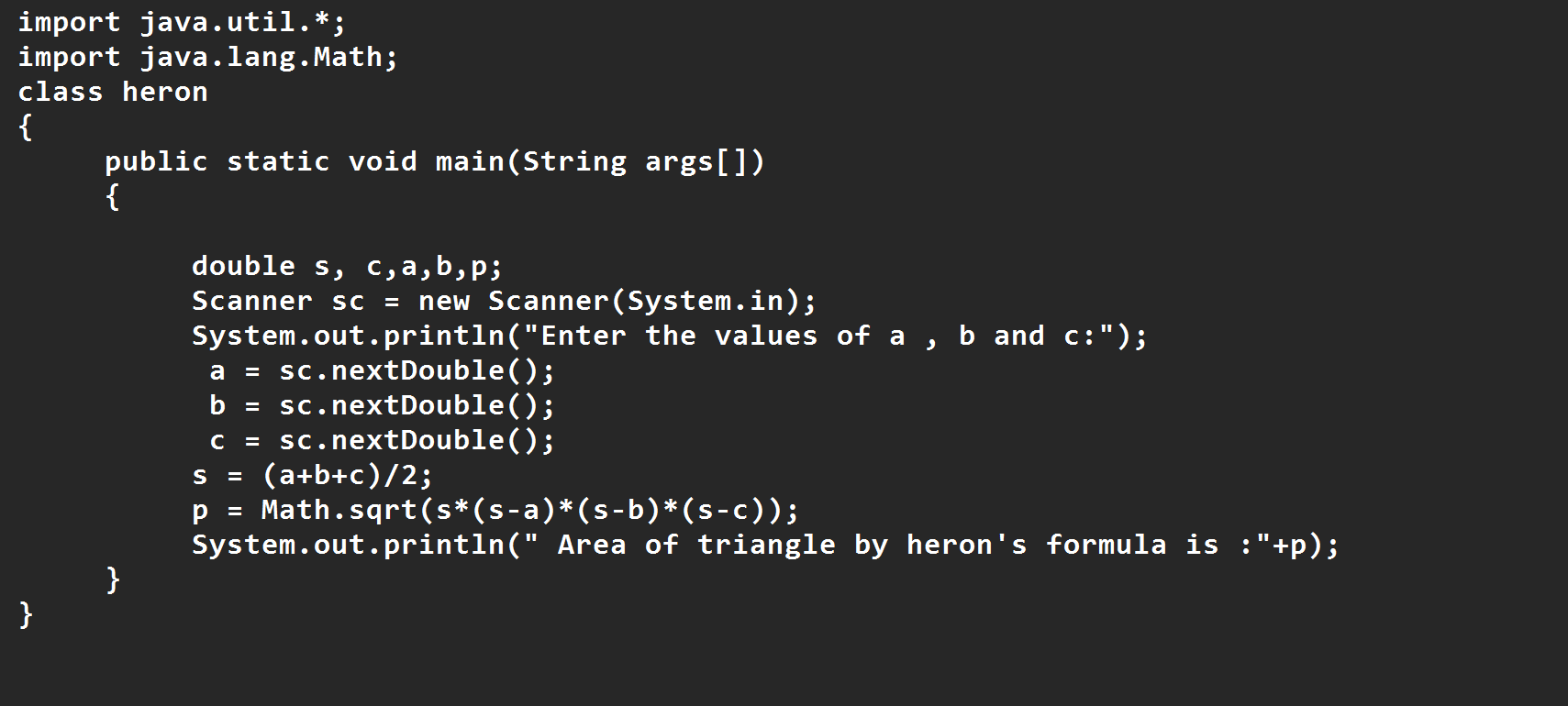
**Negative case:**

****

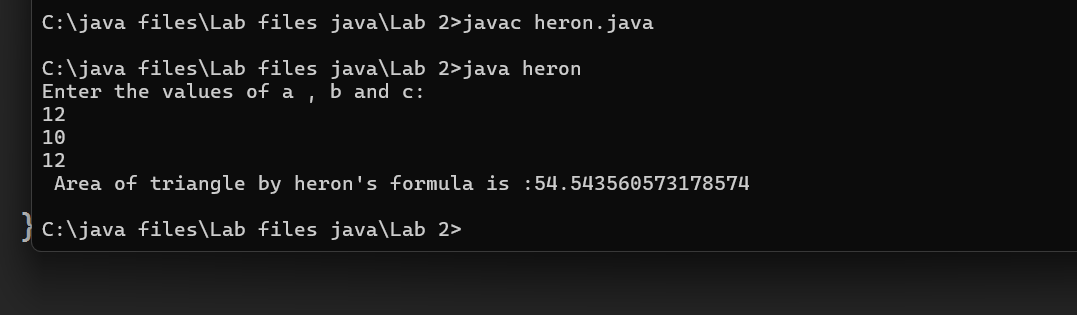
**6.Aim:**

**Write a program to find the area of triangle by using heron’s formula take the input from the user**

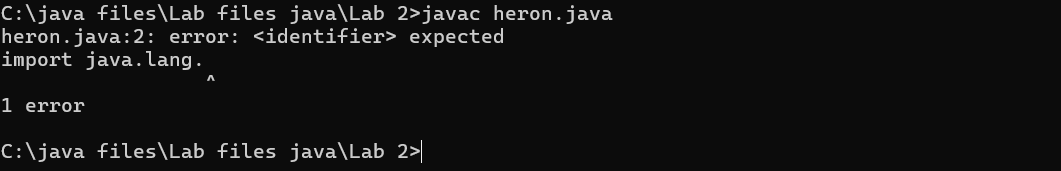
**Code:**

****

**OUTPUT:**

****

**Negative Case:**

****

**Error Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Logical error** | **Incorrect formula** | **Formula rectified** |
| **2** | **Name error** | **Undeclared variable** | **Variable declared** |
| **3** | **Import package error** | **Incorrect package** | **Package Recttified** |

**Important points**

**Import java.lang.Math is used to access the built in Math Class which provides a collection**

**of static methods**

**For performing various mathematical operations**

**Week-3**

|  |  |  |
| --- | --- | --- |
| S.No | Title | Pg no |
| 1 | Create a java program with following instructions   1. Create a class with name car 2. Create four attributes named Car\_color , Car\_brand, fuel\_type,mileage 3. Create three methods named start(),stop(),service() 4. Create three objects named Car1, Car2 and Car3 | 23-26 |
| 2 | Create a class bankAccount with elements deposit() and Withdrawl | 26-29 |

**WEEK 3**

1. **Aim:**

**To create java program with following instructions**

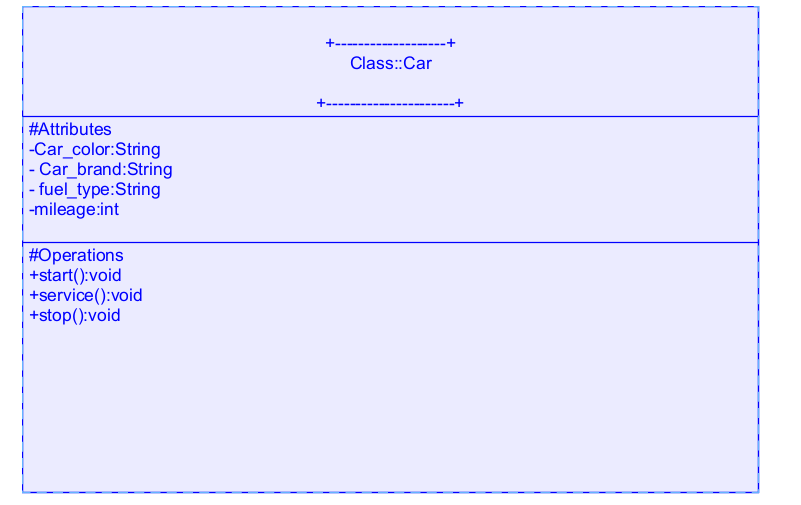
**1.Create a class with name car**

**2. Create four attributes named car\_color ,Car\_brand,fuel\_type,mileage**

**3. Create three methods named start(), stop(). Service()**

**4. Create three objects named car1,car2 and car3**

**Class Diagram:**

****

**Code:**

import java.util.\*;

class car

{

public String Car\_color;

public String Car\_brand;

public String fuel\_type;

public int mileage;

public void start()

{

System.out.println("Car Started:");

System.out.println("Car color is :"+Car\_color);

System.out.println("Car Brand is:"+Car\_brand);

System.out.println("Car fuel type is:"+fuel\_type);

System.out.println("Car mileage is:"+mileage);

}

public void service()

{

System.out.println("Car Started:");

System.out.println("Car color is :"+Car\_color);

System.out.println("Car Brand is:"+Car\_brand);

System.out.println("Car fuel type is:"+fuel\_type);

System.out.println("Car mileage is:"+mileage);

}

public void stop()

{

System.out.println("Car Started:");

System.out.println("Car color is :"+Car\_color);

System.out.println("Car Brand is:"+Car\_brand);

System.out.println("Car fuel type is:"+fuel\_type);

System.out.println("Car mileage is:"+mileage);

}

public static void main(String args[])

{ System.out.println("\nBHANU TEJA\n\n");

car car1 = new car();

car1.Car\_color = "Blue";

car1.Car\_brand = "Audi";

car1.fuel\_type = "Deisel";

car1.mileage = 100;

car1.start();

car car2 = new car();

car2.Car\_color = "Red";

car2.Car\_brand = "Tesla";

car2.fuel\_type = "EV";

car2.mileage = 200;

car2.stop();

car car3 = new car();

car3.Car\_color = "Yellow";

car3.Car\_brand = "BMW";

car3.fuel\_type = "Petrol";

car3.mileage = 300;

car3.service();

}

}

**Important points**

Methods: The set of instructions that can be called for execution using a method name.

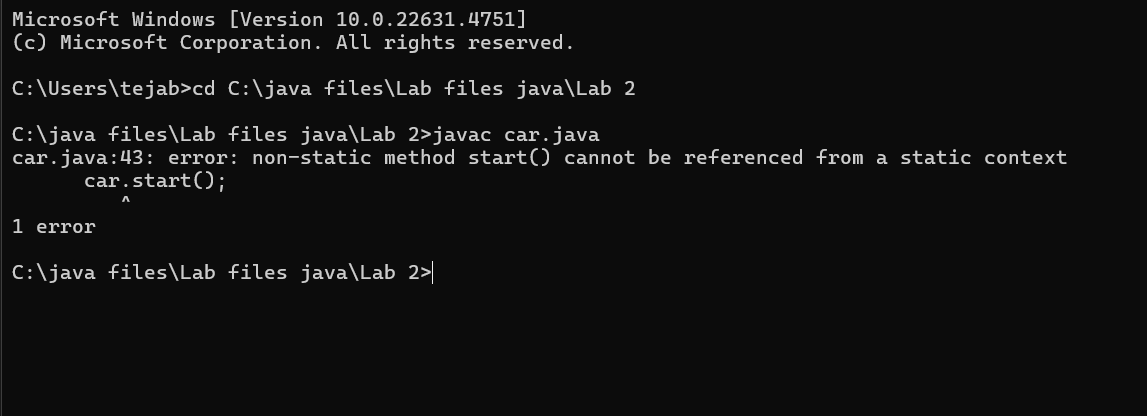
**Output:**

****

**Error Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **S No** | **Error Type** | **Cause of error** | **Rectification** |
| **1** | **Syntax Error** | **Missing ‘{‘** | **‘{‘ added** |
| **2** | **Compile time Error** | **Mispelled Variable call** | **Rectified with**  **Correct variable name** |
| **3** |  |  |  |

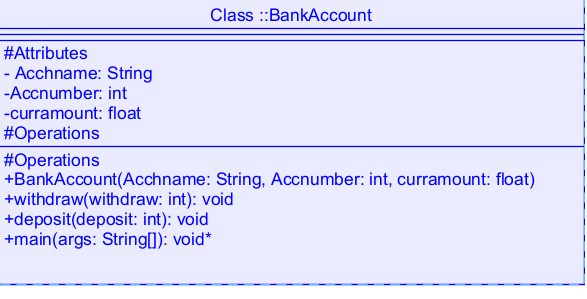
**Negative Case**

****

**2.**

**Aim: To create a class bankAccount with methods deposit() and withdrawl**

**Class Diagram**

****

**Code:**

class BankAccount

{

public String Acchname;

public int Accnumber;

public float curramount;

BankAccount(String Acchname, int Accnum,float curramount)

{

this.Acchname = Acchname;

this.Accnumber = Accnumber;

this.curramount = curramount;

System.out.println("Enter Account holder name:"+Acchname);

System.out.println("Enter Account number:"+Accnum);

System.out.println("Enter current amount:"+curramount);

}

public void withdraw(int withdraw)

{

if(withdraw>curramount)

{

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

}

else

{

curramount = curramount-withdraw;

System.out.println("withdraw amount is:"+withdraw);

System.out.println("Current amount is:"+curramount);

}

}

public void deposit(int deposit)

{

System.out.println("Deposited amount is :");

curramount = curramount+deposit;

System.out.println("Deposited amount is:"+deposit);

System.out.println("Total current amount is:"+curramount);

}

public static void main(String args[])

{

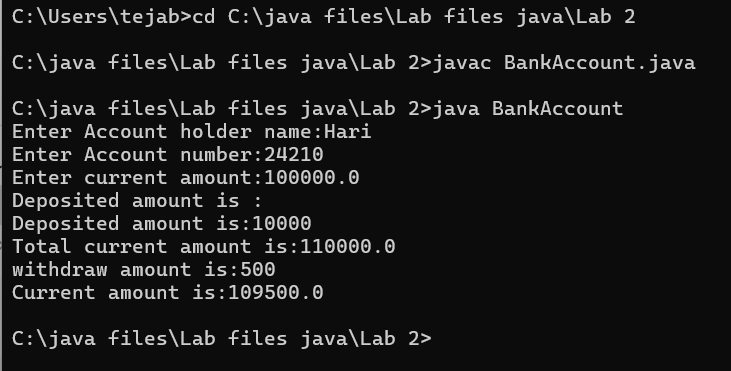
BankAccount b = new BankAccount("Hari",24210,100000);

b.deposit(10000);

b.withdraw(500);

}}

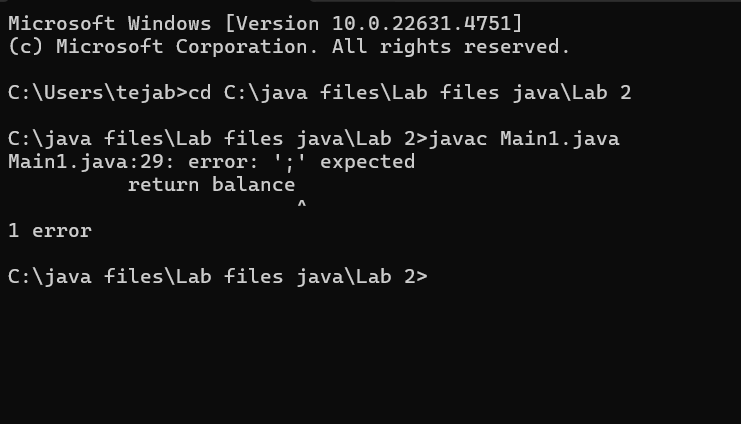
**Output:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **s.no** | **Error name** | **Cause of error** | **Rectification** |
| **1** | **Name Error** | **Undefined name** | **Correct variable**  **Name replaced** |
| **2** | **Syntax Error** | **Missing Parenthesis** | **Parenthesis Added** |
| **3** | **Logical Error** | **Incorrect Condition** | **Condition Rectified** |

**Negative Case**

****

**Important points:**

**Constructor: The Constructor creates and initializes objects of a class. They are called**

**when an object is created to a class.**

**This Keyword: The This keyword refers to the current instance of a class.It is used to**

**Access class variables and methods.**

**Class Diagram**

**WEEK 4**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Title** | **Pg no** |
| **1** | **Write a java program with class named book. The class The class should contain various**  **Attributes such as title, author, year of publication.**  **It should also contain a**  **Constructor with parameters which initializes title, author and year of**  **Publication.**  **Create a method which displays the details of the book .**  **Display**  **The details of two books** | **31-34** |
| **2** | **.Create a java program with class named “myclass” with a static variable**  **“count” of int type, initialized to zero and a constant variable “pi” of type**  **Double initialized to 3.14 as attributes of the class. Now define a constructor**  **For “myclass” that increments the count variable each time an object of**  **“myclass” is created**  **Finally Print the final values of count and pi variables. Create three objects** | **34-37** |

**Aim: Write a java program with class named book. The class should contain various**

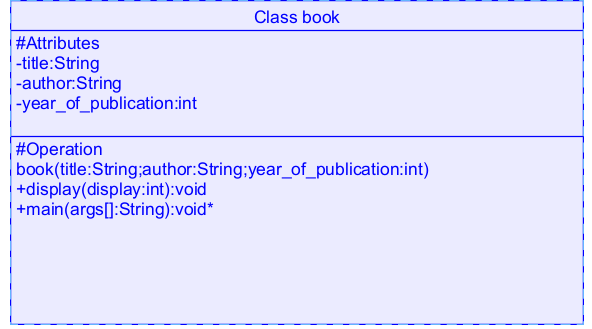
**Attributes such as title, author, year of publication. It should also contain a**

**Constructor with parameters which initializes title, author and year of**

**Publication. Create a method which displays the details of the book .Display**

**The details of two books**

**Class Diagram**

****

**Code**

**class book**

**{**

**public String title;**

**public String author;**

**public int year\_of\_publication;**

**book(String title, String author, int year\_of\_publication)**

**{**

**this.title = title;**

**this.author = author;**

**this.year\_of\_publication = year\_of\_publication;**

**}**

**public void display()**

**{**

**System.out.println("Title of book is:"+title);**

**System.out.println("Author of book is:"+author);**

**System.out.println("Year of publication is:"+year\_of\_publication);**

**}**

**public static void main(String args[])**

**{**

**book b1 = new book("Python","Bhanu",2023);**

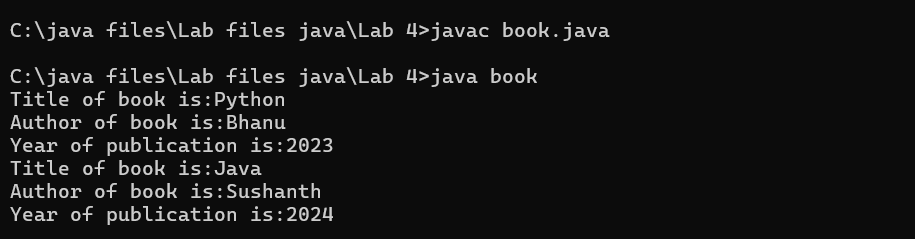
**b1.display();**

**book b2 = new book("Java","Sushanth",2024);**

**b2.display();**

**}}**

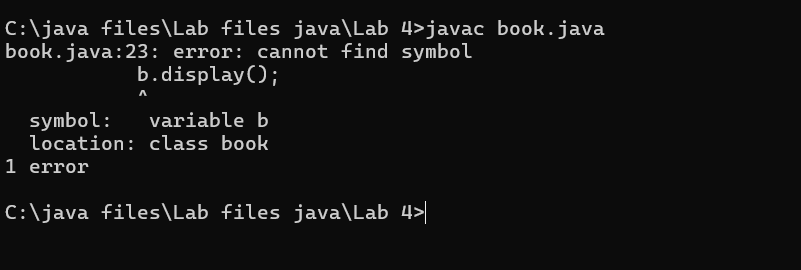
**Output:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error Type** | **Reason** | **Rectification** |
| **1** | **Syntax error** | **Missing {** | **Added {** |
| **2** | **Static method called non statically** | **Fault in calling the method** | **Rectified method**  **Calling** |
| **3** | **Run-time error** | **Incorrect Selection of**  **path** | **Correct path**  **Selected** |

**Negative Case**

****

**2.Create a java program with class named “myclass” with a static variable**

**“count” of int type, initialized to zero and a constant variable “pi” of type**

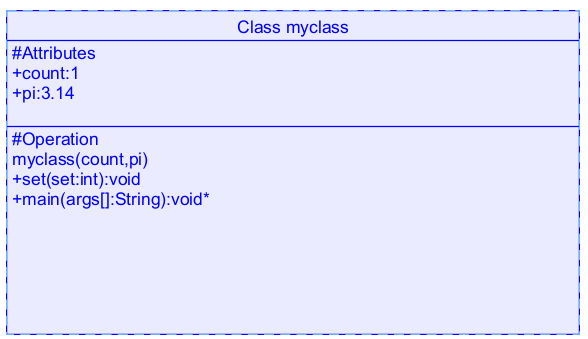
**Double initialized to 3.14 as attributes of the class. Now define a constructor**

**For “myclass” that increments the count variable each time an object of**

**“myclass” is created**

**Finally Print the final values of count and pi variables. Create three objects**

**Class Diagram:**



**Code:**

**class myclass**

**{**

**static int count=0;**

**static double pi=3.14;**

**myclass()**

**{**

**count = count+1;**

**}**

**public void set()**

**{**

**System.out.println("Count is:"+count);**

**System.out.println("Pi value is:"+pi);**

**}**

**public static void main(String args[])**

**{**

**myclass m = new myclass();**

**m.set();**

**myclass m1 = new myclass();**

**m1.set();**

**myclass m2 = new myclass();**

**m2.set();**

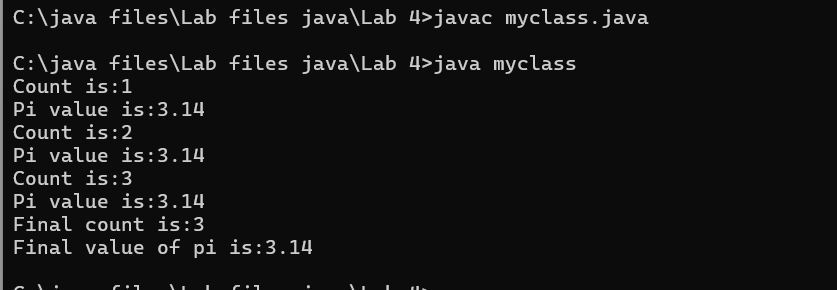
**System.out.println("Final count is:"+count);**

**System.out.println("Final value of pi is:"+pi);**

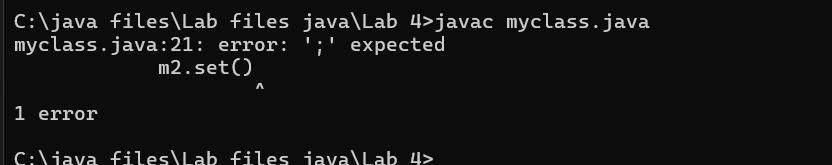
**}**

**}**

**Output:**

****

**Negative Case:**

****

**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Error Type | Cause | Rectification |
| 1 | Name error | Incorrect variable called | Rectified with correct variable |
| 2 | Syntax error | Missing semi-colon | Semi-colon added |
| 3 | Run time error | Incorrect path | Selected correct path |

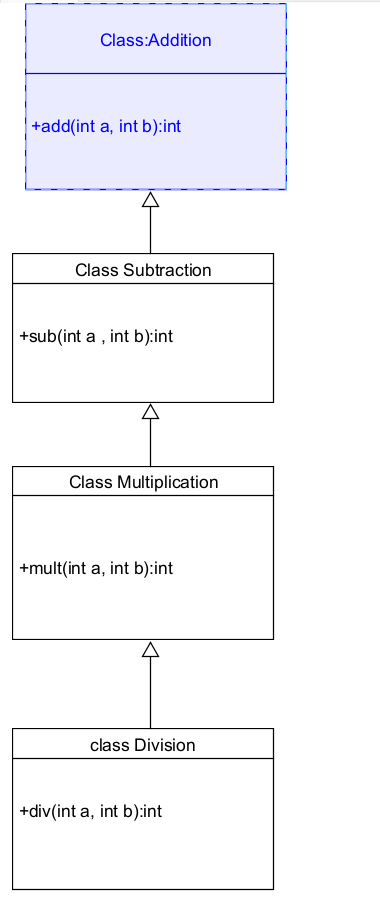
WEEK-5

AIM: Create a calculator using the operations including addition, subtraction

Multiplication and division using multilevel inheritance and display the desired

Output

Class Diagram:



Code:

class addition

{

public int add(int a, int b)

{

int addition = a+b;

return addition;

}

}

class subtraction extends addition

{

public int sub(int a, int b)

{

int subtraction = a-b;

return subtraction;

}

}

class multiplication extends subtraction

{

public int mult(int a, int b)

{

int multiplication = a\*b;

return multiplication;

}

}

class division extends multiplication

{

public int div(int a,int b)

{

int division = a/b;

return division;

}

}

class calculator

{

public static void main(String args[])

{

division obj = new division();

System.out.println("Addition is:"+ obj.add(10,2));

System.out.println ("Subtraction is:"+obj.sub(8,4));

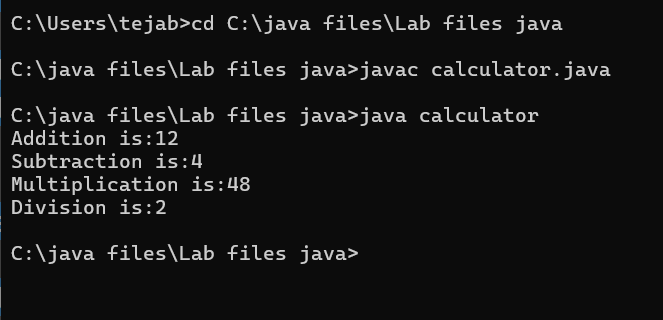
System.out.println("Multiplication is:"+obj.mult(12,4));

System.out.println("Division is:"+obj.div(8,4));

  }

}

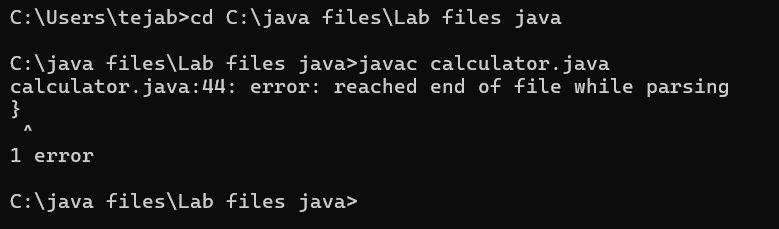
Output



Error Table

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Error Type | Cause | Rectification |
| 1 | Constructor error | Invalid name to method | Defined class name |
| 2 | Syntax error | Expected ‘(‘ | Added parenthesis |
| 3 | Logical error | Incorrect arithmetic  operation | Correct operation  rectified |

Negative Case:



Important Points

Inheritence:

The concept of OOP where a class inherits the properties and behaviours from

Another class (parent class) which promotes code reusability and hieratchical relationships

Multilevel Inheritence:

This is a type of inheritance in which a class inherited from another class, and

That superclass, in turn, inherits from yet another class, creating a chain of

inheritence

extends:

The extends keyword defines the relation of child class with the parent class

2.

Aim: A vehicle rental company wants to develop a system that maintains

Information about different types of vehicles available for rent

The Company rents out cars, bikes and truck and they need a program to

Store details about each vehicle, such as brand and speed

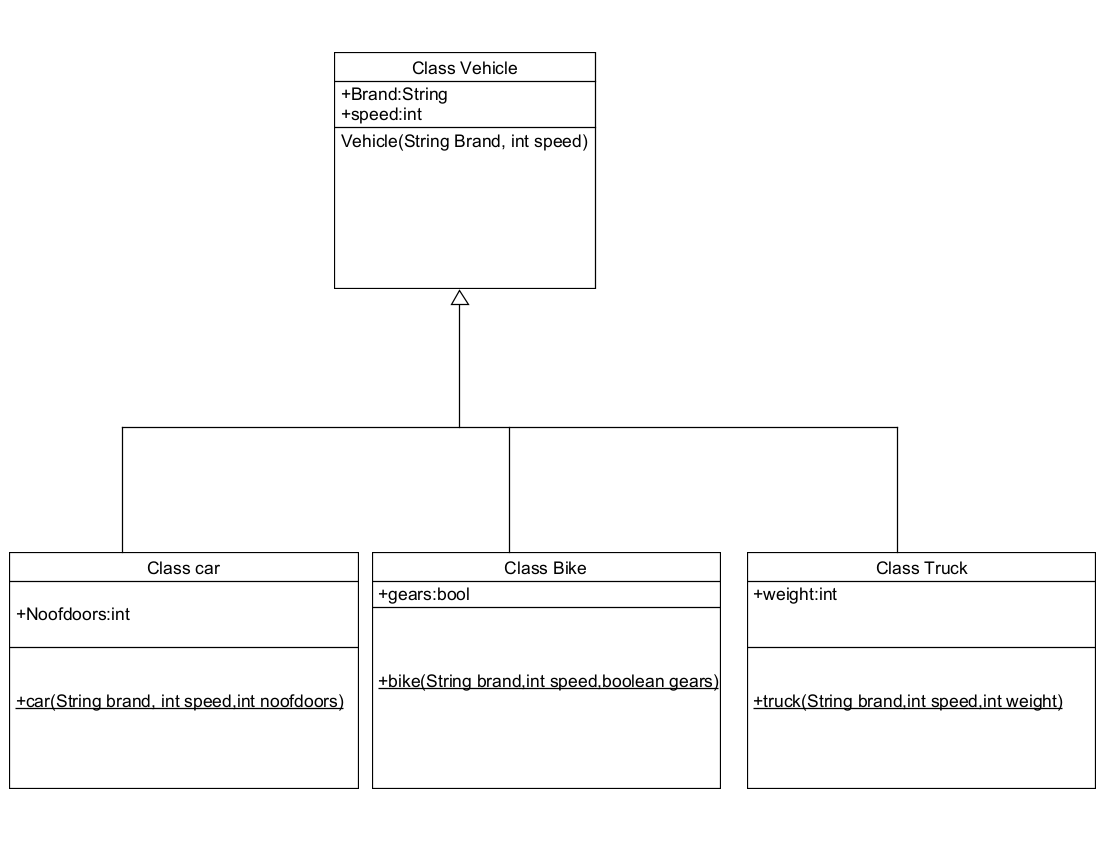
Cars should have an additional property: number of doors

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

Class diagram



Code:

class vehicle{

String brand;

int speed;

public vehicle(String brand,int speed){

this.brand=brand;

this.speed=speed;

}

public static void main(String[] args) {

car obj1=new car("ford",34,4);

bike obj2=new bike("hero",100,true);

truck obj3=new truck("tata",60,40);

}

}

class car extends vehicle{

int noofdoors;

public car(String brand, int speed,int noofdoors) {

super(brand, speed);

this.noofdoors=noofdoors;

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

System.out.println("Speed of car is:"+speed);

System.out.println("no of doors of car:"+noofdoors);

}

}

class bike extends vehicle{

boolean gears;

public bike(String brand,int speed,boolean gears){

super(brand, speed);

this.gears=gears;

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

System.out.println("Speed of bike is:"+speed);

System.out.println("Gears of bike:"+gears);

}

}

class truck extends vehicle{

int weight;

public truck(String brand,int speed,int weight){

super(brand,speed);

this.weight=weight;

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

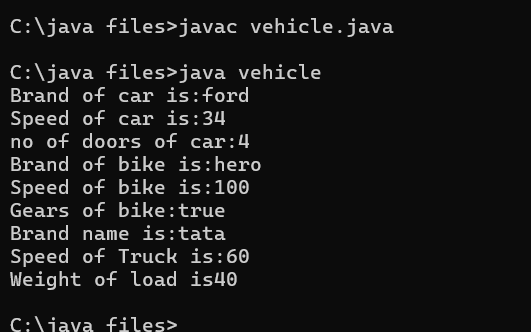
System.out.println("Speed of Truck is:"+speed);

System.out.println("Weight of load is"+weight);

}

}

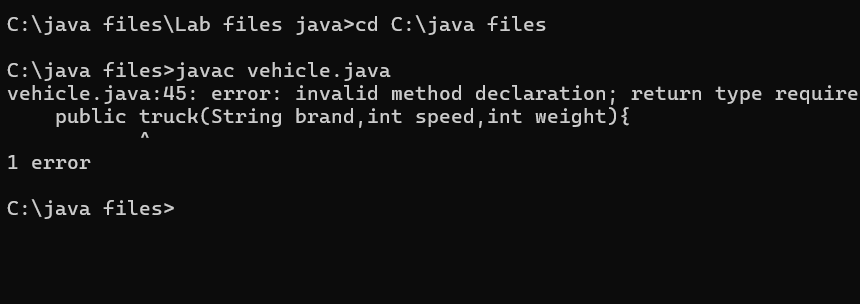
Output:



Error Table:

|  |  |  |  |
| --- | --- | --- | --- |
| S No | Error Type | Cause | Rectification |
| 1 | Syntax Error | Semicolon missing | Added ; |
| 2 |  |  |  |
| 3 |  |  |  |

Negative Case:



Important Points

Hierarchical Inheritence:

This is a type of inheritance occurs when multiple subclasses inherit from a

Single parent class

v

**WEEK-6**

1) **Write a Java program to create a vehicle class with a method displayInfo(). Override this method in the car subclass to provide specific information about a car, model, fuel type, and color using the constructor**

- Write your code in VS CODE and execute

**- Important Points:**

* + - 1. Understand the calling of a Constructor
      2. Giving class name correctly
      3. Give the parameters Correctly

**CODE:**class Vehicle {

String Brand;

String model;

String fuel;

String color;

int capacity;

Vehicle(String Brand, String model, String fuel, int capacity, String color) {

this.Brand = Brand;

this.model = model;

this.fuel = fuel;

this.capacity = capacity;

this.color = color;

}

void displayInfo(String Brand, String model, String fuel, int capacity, String color) {

System.out.println("Vehicle Details: ");

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

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

System.out.println("Fuel: " + fuel);

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

System.out.println("Color: " + color);

}

}

class Car extends Vehicle {

Car(String Brand, String model, String fuel, int capacity, String color) {

super(Brand, model, fuel, capacity, color);

}

void displayInfo() {

System.out.println("Car Details: ");

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

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

System.out.println("Fuel: " + fuel);

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

System.out.println("Color: " + color);

}

}

class Week6\_1 {

public static void main(String[] args) {

// Creating an instance of Car

Car car1 = new Car("BMW", "X5", "Petrol", 6, "Red");

car1.displayInfo(); // Display car details

}

}

**OUTPUT:**

A screen shot of a black background

AI-generated content may be incorrect.

**Errors:**

|  |  |  |
| --- | --- | --- |
| S.NO | Error Name | Error Rectification |
| 1 | Syntax/ Compilation Error | Absence of Semicolon |
| 2 | Closing Brackets | Need to Close the brackets |
| 3 | Class Name Error | Give the class name correctly |
| 4 | Constructor Calling | Call the constructor correctly |

**2) Create a Java program for the scenario.**

**A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate(PG) programs. Each program has different eligibility criteria based on the student's percentage in their previous qualification.**

**i) UG admissions require a minimum of 60%**

**ii) PG admissions require a minimum of 70%**

- Write your code in VS CODE and execute

**- Important Points:**

* + - 1. Understand the calling of a Constructor
      2. Giving class name correctly
      3. Give the parameters Correctly

**CODE:**

class College{

String name;

int percentage;

void geteligibility(String name,int percentage){

this.name=name;

this.percentage=percentage;

}

}

class UG extends College{

void geteligibility(String name,int percentage){

if (percentage>=60){

System.out.println(name+" is eligible");

}

else{

System.out.println(name+" is not eligible");

}

}

}

class PG extends College{

void geteligibility(String name,int percentage){

if (percentage>=70){

System.out.println(name+" is eligible");

}

else{

System.out.println(name+" is not eligible");

}

}

}

class week6\_2{

public static void main(String[] args){

UG ug=new UG();

ug.geteligibility("Person-1",40);

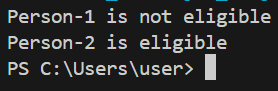
PG pg=new PG();

pg.geteligibility("Person-2",80);

}

}

**OUTPUT:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| S.NO | Error Name | Error Rectification |
| 1 | Syntax/ Compilation Error | Absence of Semicolon |
| 2 | Closing Brackets | Need to Close the brackets |
| 3 | Class Name Error | Give the class name correctly |
| 4 | Constructor Calling | Call the constructor correctly |

**3) Write a Java Program to create a Calculator class with overloaded methods to perform addition: Take the integer values a and b from the user.**

**i) Add two integers**

**ii) Add two doubles**

**iii) Add three integers**

- Write your code in VS CODE and execute

**- Important Points:**

* + - 1. Understand the calling of a Constructor
      2. Giving class name correctly
      3. Give the parameters Correctly

**CODE:**

class Calc{

public int add(int a,int b){

return a+b;

}

public double add(double a,double b){

return a+b;

}

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

return a+b+c;

}

}

class week6\_3{

public static void main(String[] args){

Calc C1=new Calc();

System.out.println("Sum of 6 and 9 is: "+C1.add(6,9));

System.out.println("Sum of 7.6 and 8.6 is: "+C1.add(7.6,8.6));

System.out.println("Sum of 2,4 and 6 is: "+C1.add(2,4,6));

}

}

**OUTPUT:**

A black background with white text

AI-generated content may be incorrect.

**Errors:**

|  |  |  |
| --- | --- | --- |
| S.NO | Error Name | Error Rectification |
| 1 | Syntax/ Compilation Error | Absence of Semicolon |
| 2 | Closing Brackets | Need to Close the brackets |
| 3 | Class Name Error | Give the class name correctly |
| 4 | Constructor Calling | Call the constructor correctly |

**4) Write a Java Program to 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 a circle.**

- Write your code in VS CODE and execute

**- Important Points:**

* + - 1. Understand the calling of a Constructor
      2. Giving class name correctly
      3. Give the parameters Correctly

**CODE:**class Shape {

double calculateArea(double side) {

return side \* side;

}

double calculateArea(double width, double height) {

return width \* height;

}

}

class Circle extends Shape {

double calculateArea(double radius) {

return 3.14 \* radius \* radius;

}

}

class Week6\_4 {

public static void main(String[] args) {

Shape S1 = new Shape();

System.out.println("Area of square: " + S1.calculateArea(5));

System.out.println("Area of rectangle: " + S1.calculateArea(2, 5));

Circle C1 = new Circle();

System.out.println("Area of circle: " + C1.calculateArea(3));

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**Errors:**

|  |  |  |
| --- | --- | --- |
| S.NO | Error Name | Error Rectification |
| 1 | Syntax/ Compilation Error | Absence of Semicolon |
| 2 | Closing Brackets | Need to Close the brackets |
| 3 | Class Name Error | Give the class name correctly |
| 4 | Constructor Calling | Call the constructor correctly |

**WEEK-7**

**1Q)Write a java program to create an abstract class Animal with an abstract method sound().Create Subclass Tiger and Lion extends the Animal class and implement the sound() method to make a specific sound for each animal**

**Program:**

**abstract class Animal {**

**public abstract void sound();**

**}**

**class Lion extends Animal {**

**@Override**

**public void sound() {**

**System.out.println("Lion: Roar!");**

**}**

**}**

**class Tiger extends Animal {**

**@Override**

**public void sound() {**

**System.out.println("Tiger: Growl!");**

**}**

**}**

**public class Q1labw7 {**

**public static void main(String[] args) {**

**Animal lion = new Lion();**

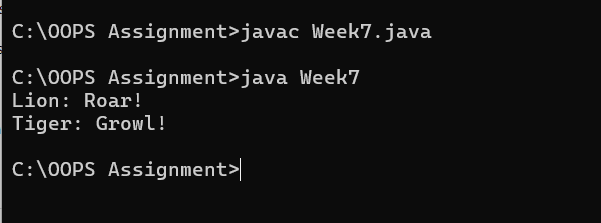
**Animal tiger = new Tiger();**

**lion.sound();**

**tiger.sound();**

**}**

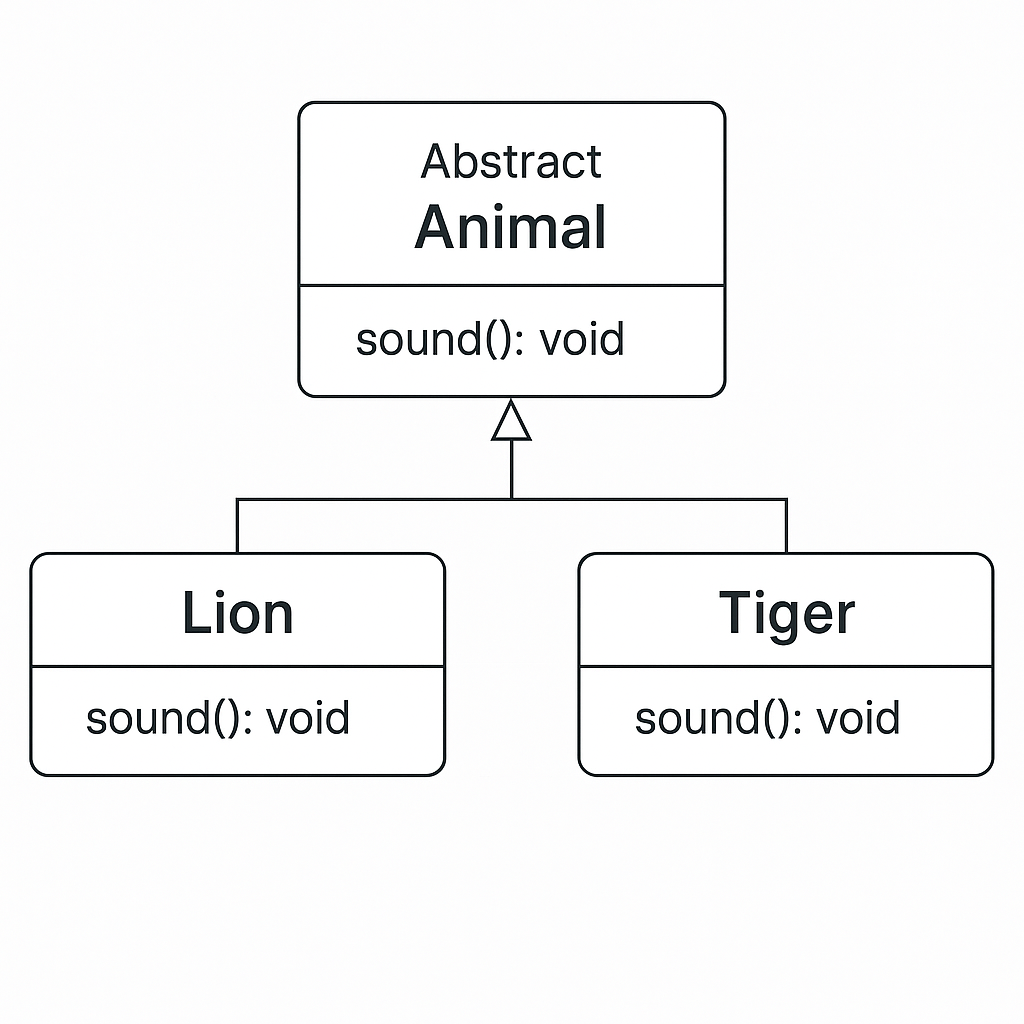
**}**

**OUTPUT:**

**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**Class Diagram:**

****

**2Q)Write a java program to create an abstract class Shape3D with an abstract methods Calculate\_volume() and Calculate\_Surface\_area.Create Subclass Sphere and Cube extends the Shape3D class and implement the respective methods to calculate the volume and surface\_area of each shape.**

**Program:**

**abstract class Shape3D {**

**public abstract void calculate\_volume();**

**public abstract void calculate\_surf\_a();**

**}**

**class Sphere extends Shape3D {**

**private double radius;**

**public Sphere(double radius) {**

**this.radius = radius;**

**}**

**@Override**

**public void calculate\_surf\_a() {**

**double surfaceArea = 4 \* Math.PI \* Math.pow(radius, 2);**

**System.out.printf("Surface Area of Sphere: %.2f%n", surfaceArea);**

**}**

**@Override**

**public void calculate\_volume() {**

**double volume = (4.0 / 3) \* Math.PI \* Math.pow(radius, 3);**

**System.out.printf("Volume of Sphere: %.2f%n", volume);**

**}**

**}**

**class Cube extends Shape3D {**

**private double side;**

**public Cube(double side) {**

**this.side = side;**

**}**

**@Override**

**public void calculate\_surf\_a() {**

**double surfaceArea = 6 \* Math.pow(side, 2);**

**System.out.printf("Surface Area of Cube: %.2f%n", surfaceArea);**

**}**

**@Override**

**public void calculate\_volume() {**

**double volume = Math.pow(side, 3);**

**System.out.printf("Volume of Cube: %.2f%n", volume);**

**}**

**}**

**public class Q2labw7 {**

**public static void main(String[] args) {**

**Shape3D sphere = new Sphere(5);**

**Shape3D cube = new Cube(3);**

**sphere.calculate\_surf\_a();**

**sphere.calculate\_volume();**

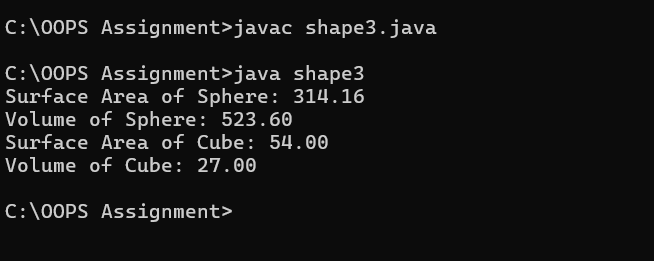
**cube.calculate\_surf\_a();**

**cube.calculate\_volume();**

**}**

**}**

****

**Output: **

**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**3Q)Write a java program using an abstract class to define a method for pattern printing**

**-->create an abstract class named patternprinting with an abstract method print pattern (int n) and a concrete method to display the pattern title**

**-->impletment two sub classes**

**1)star pattern**

**Prints a right angled triangle of stars**

**2)Number pattern**

**Prints a right angled triangle of increasing numbers**

**-->in the main()method create objects of both sub classes and print the patterns for a given number of rows**

**Program:**

**import java.util.Scanner;**

**abstract class Pattern {**

**public abstract void printPattern(int n);**

**}**

**class RightTrianglePattern extends Pattern {**

**@Override**

**public void printPattern(int n) {**

**System.out.println("Right Triangle Pattern:");**

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

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

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

**}**

**System.out.println();**

**}**

**}**

**}**

**class NumberPattern extends Pattern {**

**@Override**

**public void printPattern(int n) {**

**System.out.println("number pattern:");**

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

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

**System.out.print( j);**

**}**

**System.out.println();**

**}**

**}**

**}**

**public class Q3labw7 {**

**public static void main(String[] args) {**

**Scanner input= new Scanner(System.in);**

**System.out.println("enter the n value to select number of rows");**

**int n=input.nextInt();**

**Pattern rightTriangle = new RightTrianglePattern();**

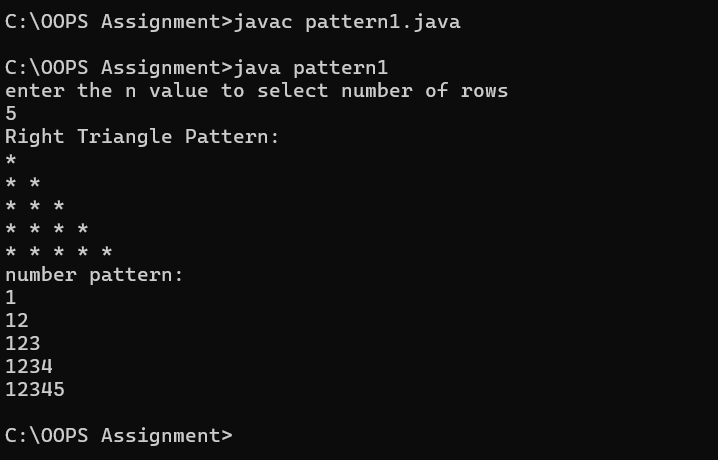
**Pattern numberpattern = new NumberPattern();**

**rightTriangle.printPattern(n);**

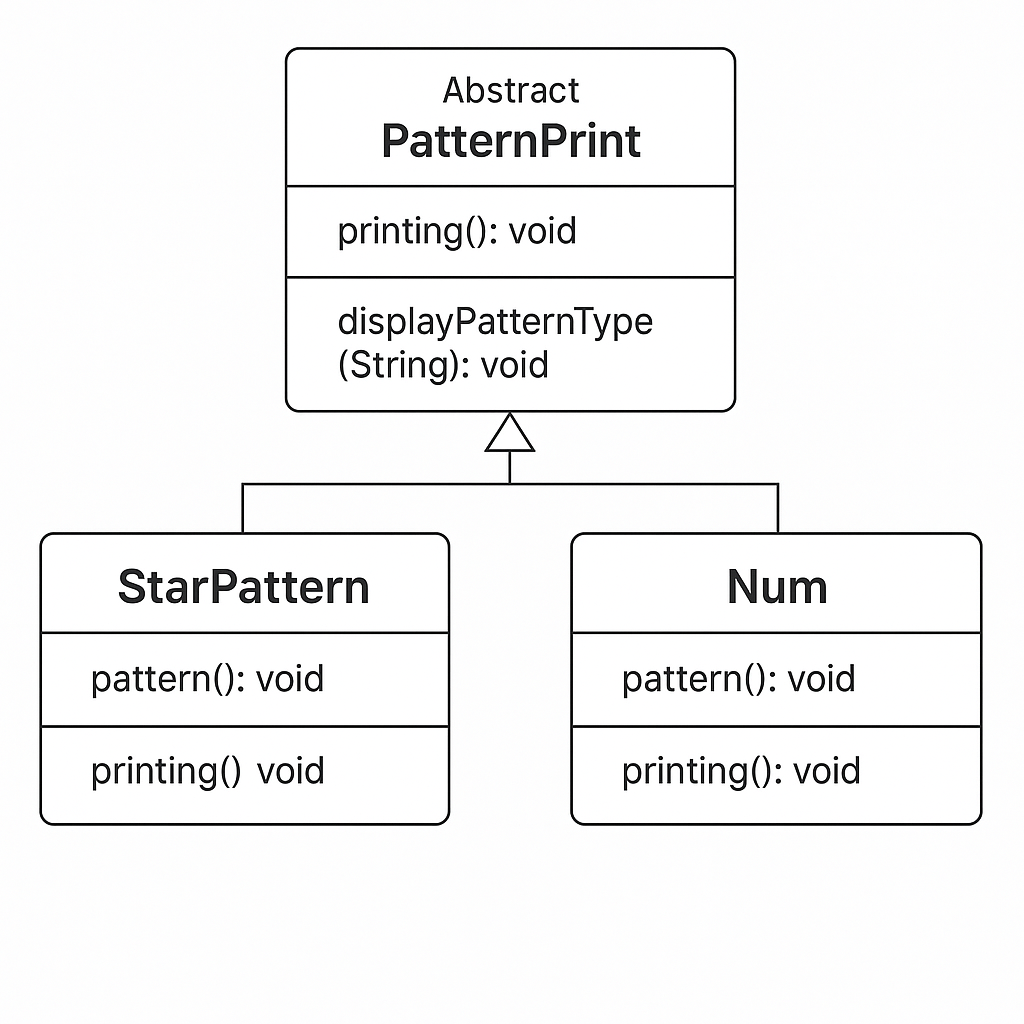
**numberpattern.printPattern(n);**

**}**

**}**

OUTPUT: 

Class diagram:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

**WEEK-8**

**1.Write a java program to create an interface shape with getPerimeter() method. Create three classes Rectangle, Circle, and Triangle that implement the shape interface implement the getPerimeter() method for each of the three classes.**

**Program:**

**interface Shapee {**

**float pi = 3.14f;**

**int getperimeter();**

**}**

**class Rectangle implements Shapee {**

**int length;**

**int breadth;**

**public Rectangle(int length, int breadth) {**

**this.length = length;**

**this.breadth = breadth;**

**}**

**public int getperimeter() {**

**return 2 \* (length + breadth);**

**}**

**}**

**class Circle implements Shapee {**

**private int radius;**

**public Circle(int radius) {**

**this.radius = radius;   }**

**public int getperimeter() {**

**return (int) (2 \* pi \* radius);**

**}}**

**class Triangle implements Shapee {**

**private int side1;**

**private int side2;**

**private int side3;**

**public Triangle(int side1, int side2, int side3) {**

**this.side1 = side1;**

**this.side2 = side2;**

**this.side3 = side3;**

**}**

**public int getperimeter() {**

**return (side1 + side2 + side3);**

**}}**

**class InShape {**

**public static void main(String[] args) {**

**System.out.println("pvskrishnakanth");**

**System.out.println("24304");**

**System.out.println("CSE-B");**

**Rectangle r = new Rectangle(5, 6);**

**System.out.println("Rectangle perimeter: " + r.getperimeter());**

**Circle c = new Circle(7);**

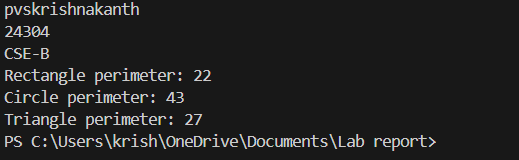
**System.out.println("Circle perimeter: " + c.getperimeter());**

**Triangle t = new Triangle(8, 9, 10);**

**System.out.println("Triangle perimeter: " + t.getperimeter());**

**}}**

**Output:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Method not implemented** | |  | | --- | |  |  |  | | --- | | **Ensure all classes implement the getperimeter() method from the interface.** | |
| **2** | |  | | --- | | **Incorrect method name** | | |  | | --- | | **Use the exact method name getperimeter() as declared in the interface.** | |
| **3** | **Missing cast in Circle calculation** | **Cast the result to int: (int)(2 \* pi \* radius).** |
| **4** | **Access modifier confusion** | |  | | --- | |  |  |  | | --- | | **Clearly define public, private, or default where appropriate.** | |

**Important Points:**

1. **Interfaces can have constants: pi in Shapee is implicitly public static final.**
2. **All classes implement the same interface, so they must define the getperimeter() method.**
3. **Casting is done in Circle: (int)(2 \* pi \* radius) — this truncates the result to an integer.**
4. **Access Modifiers: Notice mixed use of public/default (e.g., length is package-private in Rectangle)**

**2.Write a java program to create an interface playable with a method play that takes no arguments and returns play that takes no arguments and returns void. Create 3 classes Football, Volleyball, Basketball that implements the playable interface and override the play method to play the respective sports.**

**Program:**

**public interface playable {**

**void play();**

**}**

**class Football implements playable {**

**public void play() {**

**System.out.println("Football is being played...");**

**}}**

**class Vollyball implements playable {**

**public void play() {**

**System.out.println("Vollyball is being played...");**

**}}**

**class Basketball implements playable {**

**public void play() {**

**System.out.println("Basketball is being played...");**

**}}**

**class played {**

**public static void main(String[] args) {**

**System.out.println("My name is : pvskrishnakanth");**

**System.out.println("My roll.no : 24304");**

**System.out.println("My class-sec : CSE-B");**

**System.out.println("..............................");**

**Football fot = new Football();**

**fot.play();**

**Vollyball voly = new Vollyball();**

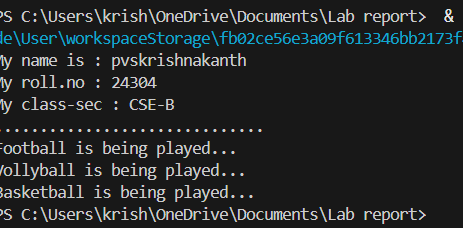
**voly.play();**

**Basketball bask = new Basketball();**

**bask.play();**

**}}**

**Output:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | |  | | --- | | **Interface not fully implemented** | | |  | | --- | | **Implement the play() method in all classes.** |   **.** |
| **2** | |  | | --- | | **Class/interface name mismatch** |  |  | | --- | |  | | **Make sure class and interface names match exactly (e.g., intplayable).** |
| **3** | |  | | --- | | **Method name spelling error** |  |  | | --- | |  | | **Use correct syntax: System.out.println("...");.** |
| **4** | |  | | --- | | **Object reference typo** |  |  | | --- | | **.** | | **Maintain consistent spelling in class names and object references** |

**Important Points:**

1. **Polymorphism is possible with intplayable but not used here (all objects are called separately).**
2. **Each class provides its own implementation of play().**
3. **Could be enhanced by using an array or list of intplayable objects and a loop.**

**3.Write a java program to implement a login system using interfaces**

**Program:**

**interface loginSystem{**

**public boolean  login(String username,String password);**

**}**

**class Universitylogin implements loginSystem {**

**public boolean login(String username, String password) {**

**if(username == "ashish1" && password == "cse1111") {**

**System.out.println("Successfully logged in..");**

**return true;**

**}**

**else {**

**System.out.println("Invalid username or password");**

**return false;**

**}**

**}}**

**class login{**

**public static void main(String[] args) {**

**System.out.println("pvskrishnakanth");**

**System.out.println("24304");**

**System.out.println("CSE-B");**

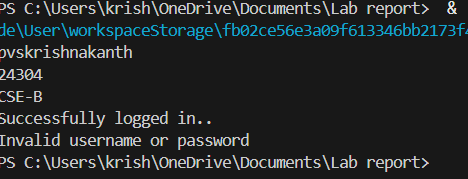
**Universitylogin ul = new Universitylogin();**

**ul.login("ashish1", "cse1111");**

**ul.login("ashih","ashish01");**

**}}**

**Output:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | |  | | --- | | **String comparison using ==** |  |  | | --- | | **Use** | | **username.equals("ashish1") instead of username == "ashish1".** |
| **2** | |  | | --- | | **Missing return statement** |  |  | | --- | |  | | **Ensure login() returns true or false as required.** |
| **3** | |  | | --- | | **Poor class naming convention** |  |  | | --- | |  | | **Rename class login to Login (PascalCase) to follow Java conventions.** |
| **4** | |  | | --- | | **Incorrect method call in main** |  |  | | --- | |  | | **Call ul.login("username", "password") correctly with matching arguments.** |

**Important Points:**

1. **Use .equals() for string comparison in Java to compare values.**
2. **Interface method is correctly implemented by Universitylogin.**
3. **The method prints a success or failure message based on input.**
4. **Proper boolean return allows flexible control in further development.**