

Object Oriented Programming (23CSE111)

Assignment

|  |  |
| --- | --- |
| **Submitted by** | |
| Name | P.Likitha Lekhya |
| Roll No | AV.SC.U4CSE24240 |
| Year/Sem/Section | 1st YEAR/SEM-2/CSE-C |
| Date of Submission |  |
| **Submitted to** | |
| Name | Dr. B Raj Kumar |
| Department | CSE |
| Designation | Asst. Professor |

|  |  |
| --- | --- |
| Marks |  |

|  |  |  |
| --- | --- | --- |
| S.NO | Q.No | PAGE No. |
| 1. | 1 |  |
| 2. | 2 |  |
| 3. | 3,19,23,26,29,30 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**PROGRAM-1**

**AIM:**

Write a java program with class named “book”. The class should contain various attributes such as “title, author, yearofpublication”. It should also contain a “constructor” with parameters which initializes “title”, ”author”, and “yearofpublication”.Create a method which displays the details of the book i.e. “author, title, yearofpublication”.(Display the details of two books i.e. create 2 objects and display their details).

**CODE:**

public class Book {

String title;

String author;

int year\_of\_publication;

public Book(String title, String author, int year\_of\_publication ) {

this.title = title;

this.author = author;

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

}

public void displayDetails() {

System.out.println(this.title);

System.out.println(this.author);

System.out.println(this.year\_of\_publication);

System.out.println();

}

}

public class Main {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("CSE-C");

System.out.println("AV.SC.U4CSE24240");

Book book\_one = new Book("The Magic of the lost Story", "Sudha Murthy", 2022);

Book book\_two = new Book("Three thousand stiches", "Sudha Murthy", 2021);

System.out.println("Book one details: ");

book\_one.displayDetails();

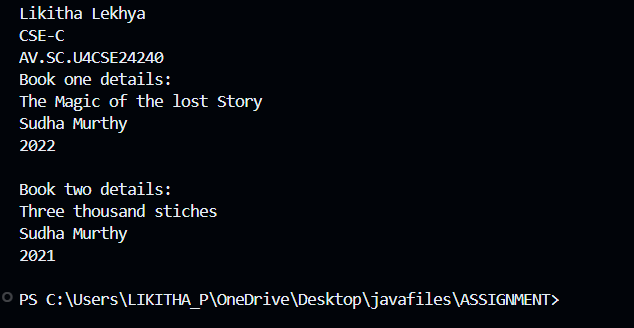
System.out.println("Book two details: ");

book\_two.displayDetails();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Not putting the semi-colon; after calling the function. 2. After Start, Stop, Service not giving the parenthesis ( ). | 1. Put the semi-colon after the writing the code. 2. After every method, put the parenthesis ( ). |

**IMPORTANT POINTS**

1. While defining two classes for a code, we must be sure that we save both the classes in separate files.
2. While defining a method we should also define a function to call that method.

**PROGRAM -2**

**AIM:**

Write a java program with class named “MyClass”, with a static variable “count” of “int” type, initialized to “0” and a constant variable “PI” of type “double” initialized to 3.14159 as attributes of that 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.

**CODE:**

public class myclass {

static int count = 0;

final double pi = 3.14;

public myclass() {

count++;

}

public static void main(String[] args) {

myclass obj1 = new myclass();

myclass obj2 = new myclass();

myclass obj3 = new myclass();

System.out.println("Likitha Lekhya");

System.out.println("CSE-C");

System.out.println("AV.SC.U4CSE24240");

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

System.out.println("Value of pi: " + obj1.pi);

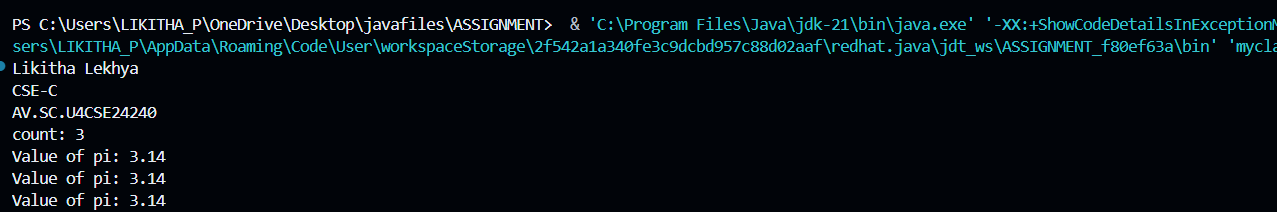
System.out.println("Value of pi: "+ obj2.pi);

System.out.println("Value of pi: " + obj3.pi);

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Incorrect datatype entered. 2. Not giving the indentation properly. | 1. Enter the correct input during the run-time. 2. All the indentation must be correct to run the code correct. |

**IMPORTANT POINTS:**

1. We must declare the initial value of the variable before declaring the final one.
2. Here the main objective is to increase the count according to the number of objects we make, i.e the count increases when the no.of objects are increasing.

**PROGRAM-4**

**AIM:**

Write a Java program that takes a number from the user and generates an integer between 1 and 7. It displays the weekday name (Use Conditional Statements).

**CODE:**

import java.util.Scanner;

public class weekdays {

public static void main(String[]args) {

Scanner input = new Scanner(System.in);

System.out.println("Enter a number(1-7): " );

int n = input.nextInt();

String day = null;

if (n==1) {

day = "Monday";

} else if (n==2) {

day = "Tuesday";

} else if (n==3) {

day = "Wednesday";

} else if (n==4) {

day = "Thursday";

} else if (n==5) {

day = "Friday";

} else if (n==6) {

day = "Saturday";

} else if (n==7) {

day = "Sunday";

} else {

System.out.println("invalid number");

}

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

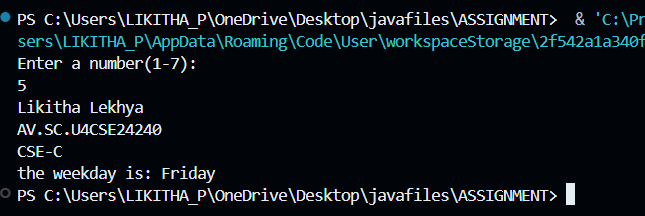
System.out.println("the weekday is: " + day );

input.close();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. While printing the variable not giving + sign. 2. Not closing the scanner. | 1. We should give correct indentation. 2. Closing the scanner is must. |

**IMPORTANT POINTS:**

1. While declaring a variable, first we must initialize the variable,
2. Here, we used the Scanner package to take the inputs from the user instead of declaring it beforehand.
3. After using the package we close the scanner package.

**PROGRAM-5**

**AIM:** Write a Java program to display the multiplication table of a given integer.

**CODE:**

import java.util.Scanner;

public class Multiplication {

public static void main(String[]args) {

Scanner input = new Scanner(System.in);

System.out.println("Table to be calculated: " );

int x = input.nextInt();

System.out.println("Enter the number: ");

int n = input.nextInt();

for (int i= 0; i<=n; i++) {

System.out.println( x + "X" + i + "=" + (x\*i) );

}

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

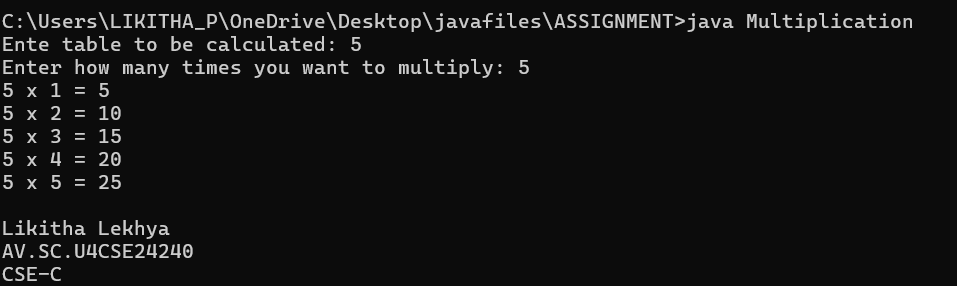
System.out.println("CSE-C");

input.close();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error while printing the variables. 2. Incorrect declaration of integer. | 1. Give the plus sign while printing. 2. Give input.nextInt(), where I should be capital. |

**IMPORTANT POINTS:**

1. Here, we used the Scanner package to take the inputs from the user instead of declaring it beforehand.
2. After using the package we close the scanner package.

**PROGRAM-6**

**AIM:**Write a Java program that reads two floating-point numbers and tests whether they are the same up to three decimal places (Use Conditional Statements).

**CODE:**

import java.util.Scanner;

public class decimal1 {

public static void main(String[]args) {

Scanner input = new Scanner(System.in);

System.out.println("Enter a number: ");

double a = input.nextDouble();

System.out.println("Enter b number: ");

double b = input.nextDouble() ;

if (a == b) {

System.out.println("They are the same upto three decimal places.");

} else {

System.out.println("They are not the same upto three decimal places.");

}

System.out.println("Likitha Lekhya");

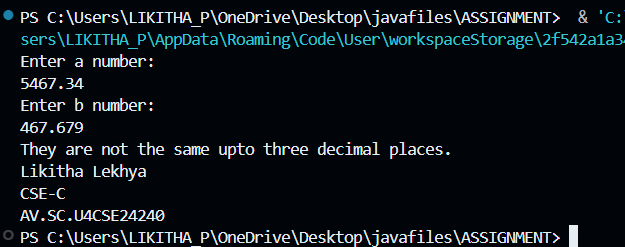
System.out.println("CSE-C");

System.out.println("AV.SC.U4CSE24240");

input.close();

}

}

**OUTPUT:** ****

**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Invalid datatype. 2. The print statement should be closed at last. | 1. Decalre double instead of int. 2. Add the ‘;’ after each and every statement. |

**IMPORTANT POINTS:**

1. The ‘double’ data type in java states that it can calculate upto any decimal places.
2. Here, the if and else statement is used as if both the floating-point numbers are equal it prints the if statement or else prints the else statement.

**PROGRAM-7**

**AIM:**Write a program that accepts three numbers from the user and prints "increasing" if the numbers are in increasing order, "decreasing" if the numbers are in decreasing order, and "Neither increasing or decreasing order" otherwise (Use Conditional Statements).

**CODE:**

import java.util.Scanner;

public class incordec {

public static void main(String[]args) {

Scanner input = new Scanner(System.in);

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

int a = input.nextInt();

System.out.print("Enter b number: ");

int b = input.nextInt();

System.out.print("Enter c number: ");

int c = input.nextInt();

System.out.println("Likitha Lekhya");

System.out.println("CSE-C");

System.out.println("AV.SC.U4CSE24240");

if (a<b) {

if (b<c) {

System.out.println("Increasing order");

} else {

System.out.println("Neither inreasing nor decreasing");

}

} else {

System.out.println("Decreasing order");

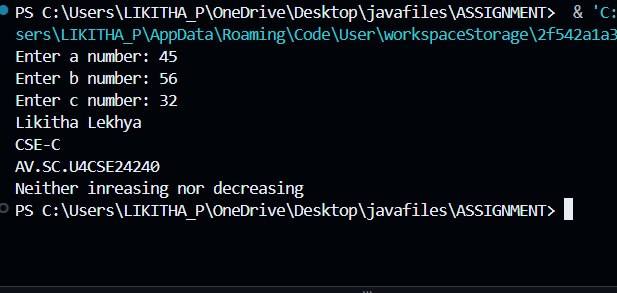
}

input.close();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Incorrect data type. 2. Didn’t close the input. | 1. Declare int instead of Stirng. 2. Close the input at last like ‘input.close();’ |

**IMPORTANT POINTS**:

1. Here, we are finding whether the given numbers are in the increasing order or decreasing order.
2. We do it by using the if and else statements,.
3. According to the given condition in the question, if a<b and b<c, then it prints in increasing order or it prints in decreasing order or it prints neither increasing nor decreasing.

**PROGRAM-8**

**AIM:**

Write a Java program that reads a positive integer and count the number of digits the number (less than ten billion) has (Use Conditional Statements)

**CODE:**

import java.util.Scanner;

public class noofdigits {

public static void main(String[]args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter a number less than ten billion: ");

int a = input.nextInt();

int b = a;

int i = 0;

while (a>0) {

a = a/10;

i++;

}

if ( b == 0) {

i = 1;

}

System.out.println("the number of digits in the number is: " +i );

System.out.println("Likitha Lekhya");

System.out.println("CSE-C");

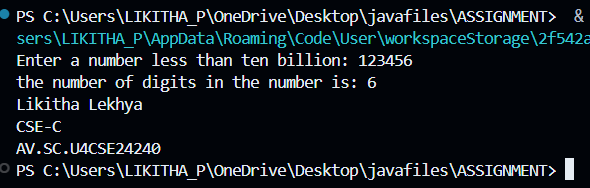
System.out.println("AV.SC.U4CSE24240");

input.close();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Variable I not found. 2. Not giving the value of a to b. | 1. Initialize variable I to 0. 2. Giving the value of a to b prints the correct output. |

**IMPORTANT POINTS:**

1. Here, we are using the while loop to execute our program.
2. In the while loop, we are dividing the number by 10 and then increasing the count number.
3. The loop continues until the value of a becomes zero.

**PROGRAM -9**

**AIM:**

Write a Java program to display Pascal's triangle.

**CODE:**

public class pascaltriangle {

public static void main(String[]args) {

int r = 5;

for(int i = 0; i<r; i++) {

for(int j = 0; j<(r-i); j++) {

System.out.print(" ");

}

int n = 1;

for(int j = 0; j <=i; j++) {

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

n = n\* (i - j)/(j + 1);

}

System.out.println( );

}

System.out.println("Likitha Lekhya");

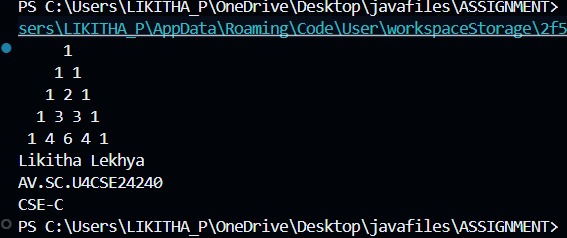
System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Wrong comma in the for loop. 2. Using println instead of using print. | 1. Put the ‘;’ istead of ‘,’ in the for loop. 2. Thought it runs the code it doesn’t give the right output. |

**IMPORTANT POINTS:**

1. The first inner loop controls the spaces before the numbers in each row to ensure the triangle is properly aligned.
2. R – I calculates the number of spaces needed for the current row.
3. The formula n = n\*(i-j)/(j+1) is a way of calculating the binomial coefficient c(i,j) which represents the value at the jth position in the ith row.

**PROGRAM – 10**

**AIM**

Write a Java program to display the following character rhombus structure.

**CODE:**

import java.util.Scanner;

public class rhombus {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Input: ");

int n = input.nextInt();

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

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

System.out.print(" ");

}

for (char c = 'A'; c < 'A' + i; c++) {

System.out.print(c);

}

for (char c = (char) ('A' + i - 2); c >= 'A'; c--) {

System.out.print(c);

}

System.out.println();

}

for (int i = n - 1; i >= 1; i--) {

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

System.out.print(" ");

}

for (char c = 'A'; c < 'A' + i; c++) {

System.out.print(c);

}

for (char c = (char) ('A' + i - 2); c >= 'A'; c--) {

System.out.print(c);

}

System.out.println();

}

input.close();

System.out.println("Likitha Lekhya");

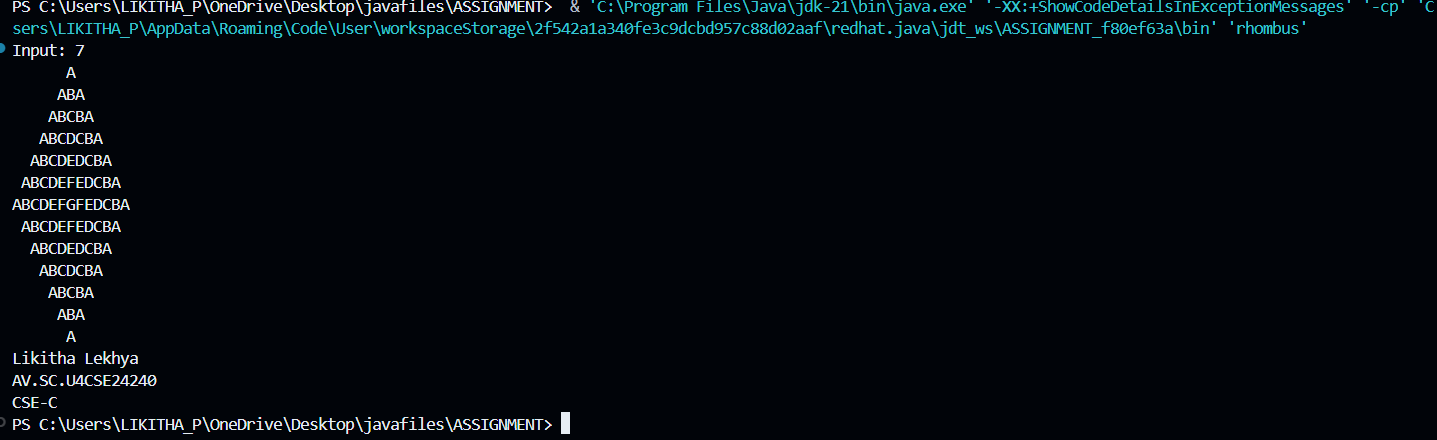
System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Declaring the int type instead of char. 2. Sometimes printing ‘println’ instead of print. | 1. Declare char instead of int type. 2. Though it gives the output, it gives the wrong answer. |

**IMPORTANT POINTS:**

1. The program starts by taking an integer n as input, which represents the size of the rhombus.
2. The first for loop iterates from 1 to n. It prints the upper half of the rhombus.
3. Second and third loops handle the printing of the alphabet pattern. The second loop prints from A to the i-th letter, and the third loop prints the decreasing sequence of letters to completer the row.

**PROGRAM-11**

**AIM**

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

**CODE:**

abstract class vehicle {

public String make;

public String model;

public int year;

public String fueltype;

public vehicle(String make, String model, int year, String fueltype) {

this.make = make;

this.model = model;

this.year = year;

this.fueltype = fueltype;

}

abstract double fuelefficiency();

abstract double maxspeed();

public double distancetravelled(double fuelused) {

return fuelefficiency()\*fuelused;

}

public String getmake() {

return make;

}

public String getmodel() {

return model;

}

public int getyear() {

return year;

}

public String getfueltype() {

return fueltype;

}

}

class truck extends vehicle {

public truck(String make, String model, int year, String fueltype) {

super(make, model, year, fueltype);

}

public double fuelefficiency() {

return 5.5;

}

public double maxspeed() {

return 145.0;

}

}

class car extends vehicle {

public car(String make, String model, int year, String fueltype) {

super(make, model, year, fueltype);

}

public double fuelefficiency() {

return 6.0;

}

public double maxspeed() {

return 150.0;

}

}

class motorcycle extends vehicle {

public motorcycle(String make, String model, int year, String fueltype) {

super(make, model, year, fueltype);

}

public double fuelefficiency() {

return 7.0;

}

public double maxspeed() {

return 155.0;

}

}

public class vtest {

public static void main(String[] args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

vehicle truck = new truck("Ford", "F-150", 2023, "Diesel");

vehicle car = new car("Toyota", "Corolla", 2022, "Petrol");

vehicle motorcycle = new motorcycle("Yamaha", "YZF-R1", 2021, "Petrol");

System.out.println("truck details:");

System.out.println("Make: " + truck.getmake() + ", Model: " + truck.getmodel());

System.out.println("Fuel Efficiency: " + truck.fuelefficiency() + " km/l");

System.out.println("Max Speed: " + truck.maxspeed() + " km/h");

System.out.println("Distance travelled:" + truck.distancetravelled(10) + " km");

System.out.println();

System.out.println("car details:");

System.out.println("Make: " + car.getmake() + ", Model: " + car.getmodel());

System.out.println("Fuel Efficiency: " + car.fuelefficiency() + " km/l");

System.out.println("Max Speed: " + car.maxspeed() + " km/h");

System.out.println("Distance traveled with 10 liters of fuel: " + car.distancetravelled(10) + " km");

System.out.println();

System.out.println("motorcycle details:");

System.out.println("Make: " + motorcycle.getmake() + ", Model: " + motorcycle.getmodel());

System.out.println("Fuel Efficiency: " + motorcycle.fuelefficiency() + " km/l");

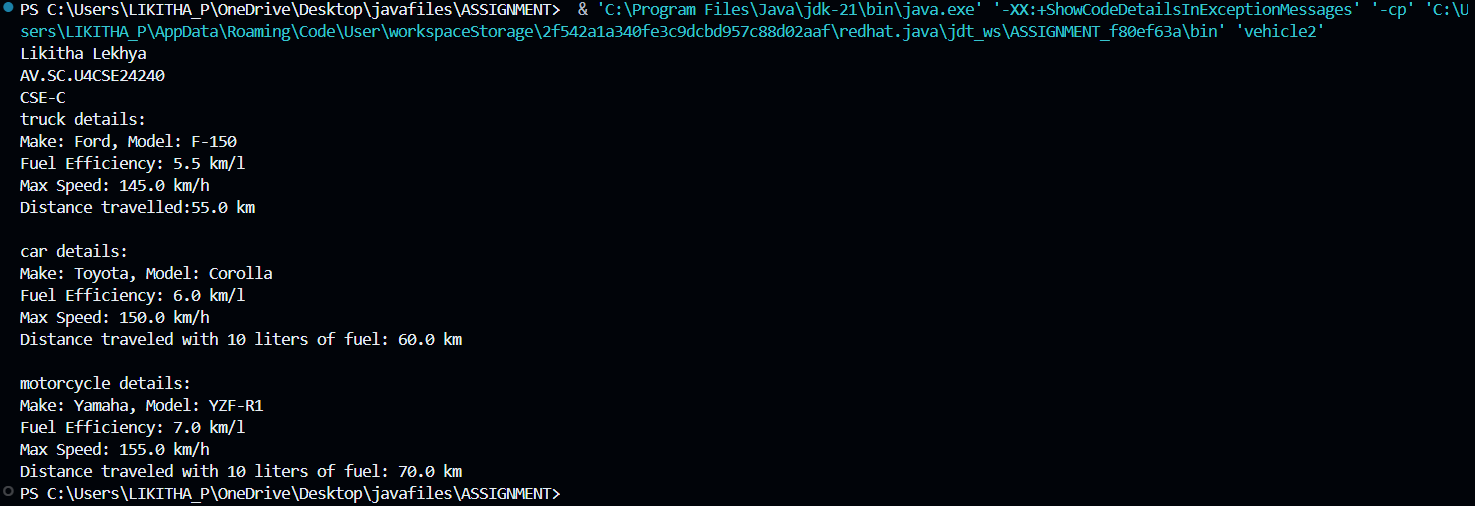
System.out.println("Max Speed: " + motorcycle.maxspeed() + " km/h");

System.out.println("Distance traveled with 10 liters of fuel: " + motorcycle.distancetravelled(10) + " km");

}

}

OUTPUT:



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error while printing the print statement. 2. no values in the constructor. | 1. Give + sign when you want to print methods in the print statement. 2. Initialize the value of variables in the constructor. |

**IMPORTANT POINTS:**

1. Here, we declared the class abstract because there are some methods which do not have any return method.
2. Generally, abstract classes are used to define a common structure and behavior for a group of related classes.
3. We created a constructor in the super class and linked it with the subclasses so that the details of the subclasses can also be printed.
4. For that, we use the ‘super’ keyword to access the constructor of the super class.
5. We also override certain methods to access the information.
6. Here, even after initializing values in the constructor, we still declared gettermethods and give a return type for it to access them individually.

**PROGRAM-12**

**AIM**

Write a Java program to create a class called Employee with methods called work () and getSalary(). Create a subclass called HRManager that overrides the work () method and adds a new method called addEmployee().

**CODE:**

public class Employee {

String name;

int salary;

public Employee(String name, int salary) {

this.name = name;

this.salary = salary;

}

public String getname() {

return name;

}

public int getsalary() {

return salary;

}

public void work() {

System.out.println(name + " is working under ABC technologies.");

}

}

class HRmanager extends Employee {

public HRmanager(String name, int salary) {

super(name,salary);

}

public void work() {

System.out.println(name + " is working as a HRmanager at ABC technologies.");

}

public void addEmployee(String employee) {

System.out.println(name + " added " + employee + " to the Employee list.");

}

}

public class employeetest {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

Employee e = new Employee("Aryan", 2400000);

System.out.println(e.getname() + "'s basic salary is: " + e.getsalary());

e.work();

HRmanager hr = new HRmanager("prem", 100000);

System.out.println(hr.getname() + "'s basic salary is: " + hr.getsalary());

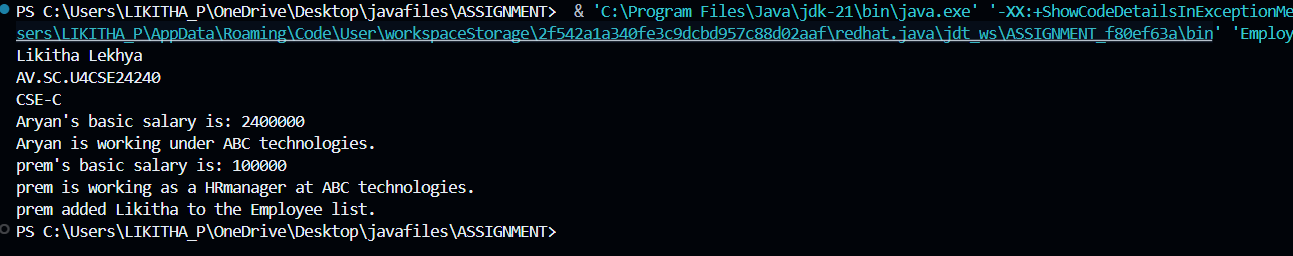
hr.work();

hr.addEmployee("Likitha");

}

}

**OUTPUT:**

**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error while printing the print statement. 2. no values in the constructor. | 1. Give + sign when you want to print methods in the print statement. 2. Initialize the value of variables in the constructor. |

**IMPORTANT POINTS:**

1. Here, we declared two variables and then initialized them through a constructor.
2. Then created two methods to access them separately.
3. To create a new method in the subclass we declared a new argument within the parenthesis in the subclass.

**PROGRAM- 13**

**AIM**

Create a calculator using the operations including addition, subtraction, multiplication and division using multi-level inheritance and display the desired output.

**CODE:**

class calculator {

    public double a, b;

    public calculator(double a, double b) {

        this.a = a;

        this.b = b;

    }

}

class Addition extends calculator {

    public Addition(double a, double b) {

        super(a, b);

    }

    public double add() {

        return a + b;

    }

}

class Subtraction extends Addition {

    public Subtraction(double a, double b) {

        super(a, b);

    }

    public double subtract() {

        return a - b;

    }

}

class Multiplication extends Subtraction {

    public Multiplication(double a, double b) {

        super(a, b);

    }

    public double multiply() {

        return a \* b;

    }

}

class Division extends Multiplication {

    public Division(double a, double b) {

        super(a, b);

    }

    public double divide() {

        if (b != 0) {

            return a / b;

        } else {

            System.out.println("Error");

            return Double.NaN;

        }

    }

}

class Final extends Division {

    public Final(double a, double b) {

        super(a, b);

    }

    public void displayResults() {

        System.out.println("Addition: " + add());

        System.out.println("Subtraction: " + subtract());

        System.out.println("Multiplication: " + multiply());

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

    }

}

import java.util.Scanner;

public class allcalculator {

public static void main(String[] args) {

System.out.println("Likitha lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-c");

Scanner input = new Scanner(System.in);

System.out.println("Enter a number: ");

double a = input.nextDouble();

System.out.println("Enter b number: ");

double b = input.nextDouble();

Final calc = new Final( a, b);

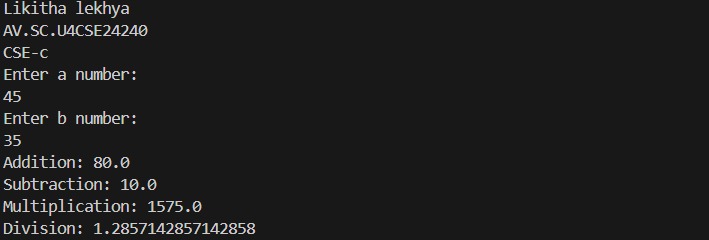
calc.displayResults();

input.close();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error, is showing zero. 2. Incorrect datatype entered. | 1. Declare the Nan after the double. 2. Declare int instead of double. |

**IMPORTANT POINTS:**

1. To get the inputs from the user we use import java.util.Scanner; this is a package.
2. Scanner class is used to get the user input.
3. in java.util.Scanner, the java.util is a package while Scanner is a class of the java.util package.
4. to import a whole package, end the sentence with an asterisk sign(\*).
5. Here, Nan stands for ‘Not a Number’ which mostly used while dividing two numbers as if it equals to zero, instead of printing an error it just passes declaring it as not a number.

**PROGRAM-14**

**AIM**

Consider a software system for a company that manages its employees. The company categorizes its employees into two primary types: RegularEmployee and Manager. Both types of employees share common attributes such as name and employee ID, but managers have attributes such as a bonus. You are tasked with designing the Java classes for this scenario and add up the salary for each type.

**CODE:**

public class software {

public String name;

public int employeeId;

public software(String name, int employeeId) {

this.name = name;

this.employeeId = employeeId;

}

public String getname() {

return name;

}

public int getemployeeId() {

return employeeId;

}

}

class RegularEmployee extends software {

public RegularEmployee(String name, int employeeId) {

super(name,employeeId);

}

}

class Manager extends software {

public Manager(String name, int employeeId) {

super(name,employeeId);

}

public void bonus(int bonus) {

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

}

}

public class softwaretest {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

software s = new software("likitha", 123456);

RegularEmployee re = new RegularEmployee("Lekhya", 34567);

Manager m = new Manager("honey", 3456788);

System.out.println("Boss " + s.getname());

System.out.println(re.getname() + " is a regular employee.");

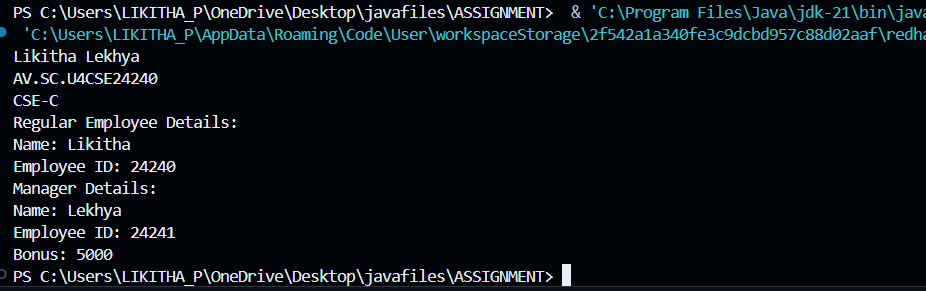
System.out.println(m.getname() + " is their manager.");

m.bonus(50000);

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Subclass doesn’t have a super() constructor. | 1. Every subclass must implement a super constructor. |

**IMPORTANT POINTS:**

1. Here, we declared two variables and then initialized them through a constructor.
2. Then created two methods to access them separately.
3. To create a new method in the subclass we declared a new argument within the parenthesis in the subclass.

**PROGRAM – 15**

**AIM**

A superclass named “Shapes” has a method called “area()”. Subclasses of “Shapes” can be “Triangle”, “circle”, “Rectangle”, etc. Each subclass has its own way of calculating area. Using base class as Shapes with subclasses triangle, circle and rectangle, use overriding polymorphism and find the area for each shape.

**CODE:**

abstract class shapes {

abstract double area();

}

class triangle extends shapes {

public double base;

public double height;

public triangle(double base, double height) {

this.base = base;

this.height = height;

}

double area() {

return (0.5\*base\*height);

}

}

class circle extends shapes {

public double r;

public circle(double r) {

this.r = r;

}

double area() {

return (3.14\*r\*r);

}

}

class rectangle extends shapes {

public double l;

public double b;

public rectangle(double l, double b) {

this.l = l;

this.b = b;

}

double area() {

return(l\*b);

}

}

public class shapestest {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

triangle t = new triangle(4,5);

circle c = new circle(2);

rectangle r = new rectangle(2,4);

System.out.println("Area of a Triangle is: " + t.area());

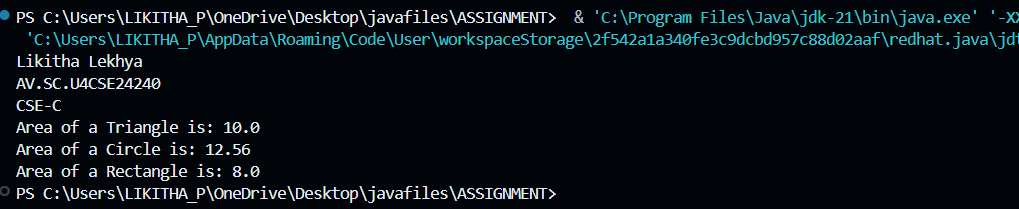
System.out.println("Area of a Circle is: " + c.area());

System.out.println("Area of a Rectangle is: " + r.area());

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error while printing the variables. 2. Incorrect declaration of integer. | 1. Give the plus sign while printing. 2. Give input.nextInt(), where I should be capital. |

**IMPORTANT POINTS:**

1. Here we used the abstract to declare an abstract class.
2. Abstract classes and methods help us to declare the methods without declaring the return type in them.
3. To get the values, we declared a constructor for each subclass and initialized values for them.

**PROGRAM – 16**

**AIM**

creating one superclass Animal and three subclasses, Herbivores, Carnivores, and Omnivores. Subclasses extend the superclass and override its eat() method. Returning the method for the required type of animals.

**CODE:**

public class Animal {

public String example(String name) {

System.out.println("Example of an Animal: " + name);

return name;

}

public void eat() {

System.out.println("All the animals follow the food chain.");

}

}

class Herbivores extends Animal {

public void eat() {

System.out.println("Herbivores eat grass.");

}

public String example(String name) {

System.out.println("Example of a Herbivore: " + name);

return name;

}

}

class Carnivores extends Animal {

public void eat() {

System.out.println("Carnivores eat Herbivores.");

}

public String example(String name) {

System.out.println("Example of a Carnivore: " + name);

return name;

}

}

class Omnivores extends Animal {

public void eat() {

System.out.println("Omnivores eat both Herbivores and Carnivores.");

}

public String example(String name) {

System.out.println("Example of an Omnivore: " + name);

return name;

}

}

public class Animaltest {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

Animal a = new Animal();

Herbivores h = new Herbivores();

Carnivores c = new Carnivores();

Omnivores o = new Omnivores();

a.eat();

h.eat();

h.example("goat");

c.eat();

c.example("lion");

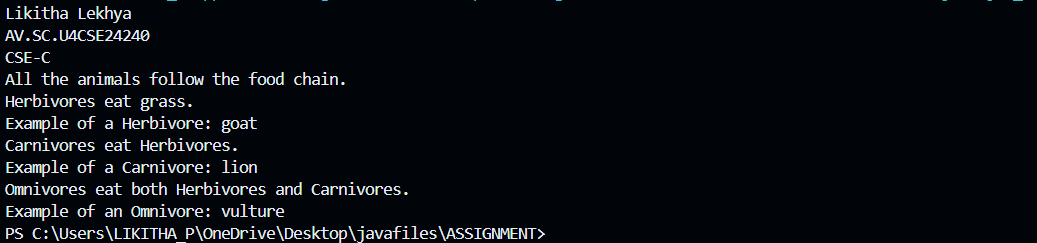
o.eat();

o.example("vulture");

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. No return method after declaring a variable. 2. Void doesn’t support return method. | 1. After declaring a variable in the method,we must return that variable.   2.Void itself says that it doesn’t return any method, it just prints the statement. |

**IMPORTANT POINTS:**

1. Here, we initialized a variable inside a method in the form of a parameter.
2. At last, after creating the object, while calling the method, if the parameter is passed inside the method then It prints the value.

**PROGRAM -17**

**AIM**

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.

**CODE:**

abstract class Animalnext {

abstract void sound();

}

class Lion extends Animalnext {

void sound() {

System.out.println("Lion are cute");

}

}

class Tiger extends Animalnext {

void sound() {

System.out.println("Tigers are cute");

}

}

public class Animal2next {

public static void main(String[]args) {

System.out.println("Likitha Lekhya");

System.out.println("AV.SC.U4CSE24240");

System.out.println("CSE-C");

Lion l = new Lion();

l.sound();

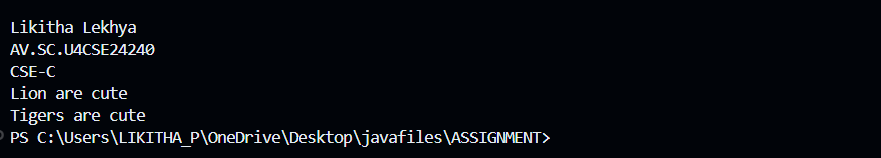
Tiger t = new Tiger();

t.sound();

}

}

**OUTPUT:**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Error while printing the variables. 2. Incorrect declaration of integer. | 1. Give the plus sign while printing. 2. Give input.nextInt(), where I should be capital. |

**IMPORTANT POINTS:**

1. We override the methods in the superclass.
2. Here we are using the heirarchial inheritance.

**PROGRAM- 18**

**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.

**CODE:**

abstract class Shape3D {

abstract double calculateVolume();

abstract double calculateSurfaceArea();

}

class Sphere extends Shape3D {

int radius;

Sphere(int 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);

}

}

class Cube extends Shape3D {

int edge;

Cube(int edge) {

this.edge = edge;

}

@Override

public double calculateVolume() {

return Math.pow(edge, 3);

}

@Override

public double calculateSurfaceArea() {

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

}

}

public class Main {

public static void main(String[] args) {

Sphere s = new Sphere(4);

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

System.out.println("Sphere Surface Area: " + s.calculateSurfaceArea());

Cube c = new Cube(3);

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

System.out.println("Cube Surface Area: " + c.calculateSurfaceArea());

System.out.println("Likitha lekhya,Roll no : 24240,CSE-C");

}

**}**

**OUTPUT:**

