

**OOPS PRACTICAL LAB FILE**

**UNIT 2**

WRITE A JAVA PROGRAM TO CREATE A CLASS NAMED SHAPE AND CREATE THREE SUB CLASSSES CIRCLE ,TRIANGLE AND SQUARE EACH CLASS HAS TWO MEMBER FUNCTION NAMED DRAW() AND ERASE(). IMPLEMENT THIS CONCEPTS USING POLYMORPHISM

class Shape {

void draw() {

System.out.println("Drawing Shape");

}

void erase() {

System.out.println("Erasing Shape");

}

}

class Circle extends Shape {

@Override

void draw() {

System.out.println("Drawing Circle");

}

@Override

void erase() {

System.out.println("Erasing Circle");

}

}

class Triangle extends Shape {

@Override

void draw() {

System.out.println("Drawing Triangle");

}

@Override

void erase() {

System.out.println("Erasing Triangle");

}

}

class Square extends Shape {

@Override

void draw() {

System.out.println("Drawing Square");

}

@Override

void erase() {

System.out.println("Erasing Square");

}

}

public class Main {

public static void main(String[] args) {

// TODO Auto-generated method stub

Shape c = new Circle();

Shape t = new Triangle();

Shape s = new Square();

c.draw(); c.erase();

t.draw(); t.erase();

s.draw(); s.erase();

}

}

Output

Drawing Circle

Erasing Circle

Drawing Triangle

Erasing Triangle

Drawing Square

Erasing Square

WRITE A JAVA PRGRAM TO GIVE A SIMPLE EXAMPLE FOR ABSTRACT CLASS

abstract class Shape{

abstract void draw();

}

//In real scenario, implementation is provided by others i.e. unknown by end user

class Rectangle extends Shape{

void draw(){System.out.println("drawing rectangle");}

}

class Circle1 extends Shape{

void draw(){System.out.println("drawing circle");}

}

//In real scenario, method is called by programmer or user

class Main{

public static void main(String args[]){

Shape s=new Circle1();//In a real scenario, object is provided through method, e.g., getShape() method

s.draw();

}

}

Output

Drawing circle

WRITE A JAVA PROGRAM TO GIVE EXAMPLE FOR MULTIPLE INHERTTANCE IN JAVA

interface Backend {

// abstract class

public void connectServer();

}

class Frontend {

public void responsive(String str) {

System.out.println(str + " can also be used as frontend.");

}

}

// Language extends Frontend class

// Language implements Backend interface

class Main extends Frontend implements Backend {

String language = "Java";

// implement method of interface

public void connectServer() {

System.out.println(language + " can be used as backend language.");

}

public static void main(String[] args) {

// create object of Language class

Main java = new Main();

java.connectServer();

// call the inherited method of Frontend class

java.responsive(java.language);

}

}

Output

Java can be used as backend language.

Java can also be used as frontend.

import java.io.\*;

class SimpleObject {

int num;

String name;

// this would be invoked while an object

// of that class is created.

SimpleObject () { System.out.println("Constructor called"); }

}

class Main {

public static void main(String[] args)

{

// this would invoke default constructor.

Main main1 = new Main();

// Default constructor provides the default

// values to the object like 0, null

System.out.println(main1.name);

System.out.println(main1.num);

}

}

Output

Constructor called

null

0

Question 5

**package oops;**

**public class Number1 {**

**private double x;**

**Number1(double x)**

**{**

**this.x=x;**

**}**

**boolean isZero()**

**{**

**if(x==0)**

**return true;**

**else**

**return false;**

**}**

**boolean isPositive()**

**{**

**if(x>0)**

**return true;**

**else**

**return false;**

**}**

**boolean isNegative()**

**{**

**if(x<0)**

**return true;**

**else**

**return false;**

**}**

**boolean isOdd()**

**{**

**if(x%2==0)**

**return false;**

**else**

**return true;**

**}**

**boolean isEven()**

**{**

**if(x%2==0)**

**return true;**

**else**

**return false;**

**}**

**boolean isPrime()**

**{**

**for(int i=2; i<x; i++)**

**{**

**if(x%i==0)**

**return false;**

**return true;**

**}**

**}**

**public class Number{**

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

**{**

**Number n=new Numer(23.9);**

**System.out.println("number is prime or not="+n.isPrime());**

**System.out.println("number is prime or not="+n.isZero());**

**System.out.println("number is prime or not="+n.isOdd());**

**System.out.println("number is prime or not="+n.isEven());**

**System.out.println("number is prime or not="+n.isPositive());**

**System.out.println("number is prime or not="+n.isNegative());**

**}**

**}**