**VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES**

**VIVEKANANDA SCHOOL OF INFORMATION TECHNOLOGY**

Logo

Description automatically generated with medium confidence

**BACHELOR OF COMPUTER APPLICATION**

**Practical-VII JAVA Lab File**

**BCA-272**

**Guru Gobind Singh Indraprastha University Sector - 16C Dwarka, Delhi – 110078**

Logo

Description automatically generated

**SUBMITTED TO: SUBMITTED BY:**

Ms. Sakshi Khullar Pranjal Tiwari

Assistant Professor 07629802021

VSIT BCA 4EB

Q1 WAP to find out factorial of a number through recursion

Code.

package pranjal\_tiwari.javafile;

import java.util.*\**;

public class ques1 {

    public static int fact(int n) {

        if(n==1){

            return 1;

        }

        else{

            return n\*fact(n-1);

        }

    }

    public static void main(String[] args) {

        Scanner sc= new Scanner(System.in);

        System.out.print("enter a number = ");

        int n=sc.nextInt();

        int ans=fact(n);

        System.out.println("factorial of "+n+" is = "+ans);

    }

}

Output

Text

Description automatically generated

Q2 WAP to print Fibonacci series. WAP to print Fibonacci series.

Code.

package pranjal\_tiwari.javafile;

import java.util.*\**;

public class ques2 {

    public static int fib(int n) {

        if(n==1 ){

            return 0;

        }

        if(n==2){

            return 1;

        }

        else{

            return fib(n-1) + fib(n-2);

        }

    }

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);

        System.out.print("enter a number = ");

        int num=sc.nextInt();

        for(int i=1;i<=num+1;i++){

            System.out.println(fib(i));

        }

    }

}

Output

A picture containing text

Description automatically generated

Q3 WAP to accept Command line arguments & print them.

Code.

package pranjal\_tiwari.javafile;

public class ques3 {

    public static void main(String args[]){

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

        System.out.println(args[i]);

        }

}

Output

A screenshot of a computer

Description automatically generated with medium confidence

Q4 WAP to obtain a number by a user & check if it’s prime or not.

Code.

package pranjal\_tiwari.javafile;

import java.util.*\**;

public class ques4 {

    public static boolean isPrime(int n) {

        boolean isPrime = true;

        for(int i=2;i<=n-1;i++){

            if(n%i==0){

                isPrime=false;

                break;

            }

        }

        return isPrime;

    }

    public static void main(String[] args) {

        Scanner sc= new Scanner(System.in);

        System.out.print("enter a number to check for prime = ");

        int n=sc.nextInt();

       System.out.println( isPrime(n));

    }

}

Output

Text

Description automatically generated

Q5 WAP that creates a class Accounts with following details:Instance variables: ac\_no., name, ac\_name, balance .Methods: withdrawal(), deposit(),display().Use constructors to initialize members.

Code.

package pranjal\_tiwari.javafile;

class Accounts{

    long account\_number;

    String name;

    String account\_name;

    double balance;

    Accounts(long account\_number, String name, String account\_name, double balance){

*this*.account\_number = account\_number;

*this*.name = name;

*this*.account\_name = account\_name;

*this*.balance = balance;

    }

    void withdrawal(int amount){

        if(amount>balance){

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

        }else{

            balance -= amount;

            System.out.println("Successfull withdrawl");

        }

    }

    void deposit(int amount){

        balance += amount;

        System.out.println("Added Successfully");

    }

    void display(){

        System.out.println("Account no: "+account\_number);

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

        System.out.println("Account type: "+account\_name);

        System.out.println("Balance: "+balance);

    }

}

public class ques5 {

    public static void main(String[] args) {

        Accounts obj = new Accounts(076, "Pranjal", "Current", 120000);

        obj.withdrawal(12000);

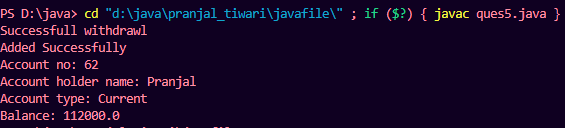
        obj.deposit(4000);

        obj.display();

    }

}

Output



Q6 WAP to implement constructor overloading.

Code.

class overload{

    overload(){

        System.out.println("performing function overloading");

    }

    overload(int a){

        System.out.println("enrollment no. is = "+a);

    }

}

 class ques6 {

    public static void main(String[] args) {

        overload obj=new overload();

        overload obj1 =new overload(76);

    }

}

Output



Q7 WAP to count the no. of objects created in a program.

Code.

package pranjal\_tiwari.javafile;

class counter{

    static int count;

    counter(){

        count++;

        System.out.println("object number = "+count);

    }

}

public class ques7 {

    public static void main(String[] args) {

        counter obj1=new counter();

        counter obj2=new counter();

        counter obj3=new counter();

        counter obj4=new counter();

        counter obj5=new counter();

    }

}

Output

Text

Description automatically generated

Q8 WAP to show call by value & call by reference.

Code.

package pranjal\_tiwari.javafile;

public class ques8 {

        public static void call\_by\_value(int number){

            number = number+1;  *// increment variable by 1*

            System.out.println("value in method: "+number);

        }

        int number=15;

        public static void call\_by\_reference(ques8 obj){

            obj.number = obj.number+1;  *// increment variable by 1*

            System.out.println("value in method: "+obj.number);

        }

        public static void main(String[] args) {

            int number=10;

            System.out.println("value before method call : "+number);

            call\_by\_value(number);

            System.out.println("value after method call: "+number);

            ques8 obj=new ques8();

            System.out.println("value before method call: "+obj.number);

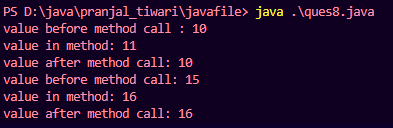
            call\_by\_reference(obj);

            System.out.println("value after method call: "+obj.number);

        }

}

Output



Q9. WAP to implement method over ridding & method overloading.

Code.

package pranjal\_tiwari.javafile;

class overload{

    public void show(){

        System.out.println("implementing overloading function in java");

    }

    public void show(String Name){

        System.out.println("student name = "+Name);

    }

}

class override extends overload{

    public void show(){

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

        System.out.println("implementing overriding function in java" );

    }

}

public class ques9 {

    public static void main(String[] args) {

        overload obj=new overload();

        obj.show();

        obj.show("Pranjal");

        override obj1=new override();

        obj1.show();

    }

}

Output

Text

Description automatically generated

Q 10. WAP that demonstrates all the usages of “super” keyword.

Code.

package pranjal\_tiwari.javafile;

class base{

    public void name(){

    System.out.println("this is base class");

    }

}

class derived extends base{

    public void name(){

*super*.name();

        System.out.println("derived class");

    }

}

public class ques10 {

    public static void main(String[] args) {

        derived obj=new derived();

        obj.name();

    }

}

Output

Graphical user interface

Description automatically generated with medium confidence

Q 11. Create a class box having height, width , depth as the instance variables & calculate its volume. Implement constructor overloading in it. Create a subclass named box\_new that has weight as an instance variable. Use super in the box\_new class to initialize members of the base class.

Code.

class Box {

    int height, width, depth;

    Box(){

        System.out.println("using default Constructor ");

    }

    Box(int height, int width, int depth) {

*this*.height = height;

*this*.width = width;

*this*.depth = depth;

    }

    void volume() {

        System.out.println("The volume of box is " + (height \* width \* depth));

    }

}

class Box\_new extends Box{

    int weight;

    Box\_new(){

*super*(16,10,12);

        weight = 10;    }

}

public class ques11 {

    public static void main(String[] args) {

        Box obj1 = new Box();

        Box obj2 = new Box(13,9,11);

        Box\_new obj3 = new Box\_new();

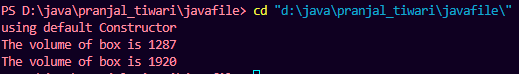
        obj2.volume();

        obj3.volume();

    }

}

Output



Q 12. WAP that implements multilevel inheritance.

Code.

package pranjal\_tiwari.javafile;

class base{

    public void show(){

        System.out.println("this is base class \n implementing mutlilevel inheritance");

    }

}

class child1 extends base{

    public void show1(){

        System.out.println("this is child 1");

    }

}

class child2 extends child1{

    public void show2(){

        System.out.println("this is child 2");

    }

}

public class ques12 {

    public static void main(String[] args) {

        child2 obj = new child2();

        obj.show();

    }

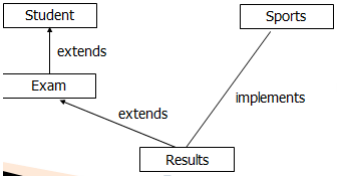
}

Output

Text

Description automatically generated with low confidence

Q 13. Consider a university where students who participate in the national games or Olympics are given some grace marks. Therefore, the final marks awarded = Exam\_Marks + Sports\_Grace\_Marks. A class diagram representing this scenario is as follow;



Code.

package pranjal\_tiwari.javafile;

import java.util.Scanner;

class Student {

    String name;

    long rollNo;

    void getData(String name, long rollNo) {

*this*.name = name;

*this*.rollNo = rollNo;

    }

}

class Exam extends Student {

    int exam\_marks;

    void getData(String name, long rollNo, int exam\_marks) {

*super*.getData(name, rollNo);

*this*.exam\_marks = exam\_marks;

    }

}

interface Sports {

    int marks = 10;

}

class Result extends Exam implements Sports {

    int resultExam() {

        return marks + exam\_marks;

    }

    void getData(String Name, long rollNo, int Exam\_Marks) {

*super*.getData(name, rollNo, exam\_marks);

    }

}

public class ques13 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        Result res = new Result();

        String name = sc.next();

        int rollNo = sc.nextInt();

        int exam\_marks = sc.nextInt();

        res.getData(name, rollNo, exam\_marks);

        System.out.println(res.resultExam());

        sc.close();

    }

}

Output

A picture containing application

Description automatically generated

Q 14. WAP to implement Run time polymorphism.

Code.

package pranjal\_tiwari.javafile;

class Shape {

    public void draw() {

        System.out.println("Drawing a shape");

    }

}

class Circle extends Shape {

    public void draw() {

        System.out.println("Drawing a circle");

    }

}

class Square extends Shape {

    public void draw() {

        System.out.println("Drawing a square");

    }

}

public class ques14 {

    public static void main(String[] args) {

        Shape obj1 = new Circle();

        Shape obj2 = new Square();

        obj1.draw();

        obj2.draw();

    }

}

Output



Q 15 . WAP to implement interface. Create an interface named Shape having area() & perimeter() as its methods. Create three classes circle, rectangle & square that implement this interface.

Code.

interface shape {

    void area();

    void perimeter();

}

class circle implements shape{

    double pi;

    int rad;

    circle(double pi, int rad){

*this*.pi = pi;

*this*.rad = rad;

    }

    public void print(){

        System.out.println("area and perimeter of circle");

    }

    public void area() {

        System.out.println( "area = "+pi\*rad);

    }

    public void perimeter(){

        System.out.println("perimeter = "+2\*pi\*rad);

    }

}

class rectangle implements shape{

    int len,wid;

    rectangle(int len, int wid){

*this*.len = len;

*this*.wid = wid;

    }

    public void print(){

        System.out.println("area and perimeter of rectangle");

    }

    public void area() {

        System.out.println("area = "+len\*wid);

    }

    public void perimeter(){

        System.out.println("perimeter = "+2\*len\*wid);

    }

}

class square implements shape{

    int side;

    square(int side){

*this*.side = side;

    }

    public void print(){

        System.out.println("area and perimeter of square");

    }

    public void area() {

        System.out.println("area = "+side\*side);

    }

    public void perimeter(){

        System.out.println("perimeter = "+2\*side);

    }

}

public class ques15 {

    public static void main(String[] args) {

        circle obj1 = new circle(3.14, 9);

        obj1.print();

        obj1.area();

        obj1.perimeter();

        rectangle obj2 = new rectangle(7, 4);

        obj2.print();

        obj2.area();

        obj2.perimeter();

        square obj3 = new square(5);

        obj3.print();

        obj3.area();

        obj3.perimeter();

}

}

Output

Graphical user interface, text, application

Description automatically generated

Q 16. WAP to show multiple inheritance.

Code

package pranjal\_tiwari.javafile;

interface A

{

    public void base1();

}

interface B

{

    public void base2();

}

class C implements A,B

{

    public void base1()

    {

        System.out.println("calling function base1");

    }

    public void base2()

    {

        System.out.println("calling function base2");

    }

}

public class ques16

{

    public static void main(String[] args)

    {

        C obj = new C();

        obj.base1();

        obj.base2();

    }

}

Output

Text

Description automatically generated

Q17. WAP to implement exception handling. The program should accept two numbers from the user & divide the first no. by the second. It should throw a Arithmetic Exception if an attempt is made to divide the no. by zero. Use try, catch & finally .Implement multi-catch statements also .

Code.

package pranjal\_tiwari.javafile;

import java.util.Scanner;

public class ques17 {

    public static void main(String[] args) {

        try{

            int numerator=76,denominator=0;

            int quotient = numerator/denominator;

            System.out.println(quotient);

        }

        catch(ArithmeticException e){

            System.out.println("Not defined");

        }

        finally{

            System.out.println("All exception handled");

        }

    }

}

Output

Text

Description automatically generated