**Asissted Practise Project**

**1.Writing a program in Java by extending the Thread class**

**package** multithreading;

**public** **class** Mythread **extends** Thread {

**public** **void** run()

{

System.***out***.println("concurrent thread started running..");

}

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

{

Mythread mt = **new** Mythread();

mt.start();

}

}

**Output:**

concurrent thread started running..

**Writing a program in Java by implementing the Runnable interface**

**package** multithreading;

**public** **class** MyRunnable **implements** Runnable {

**public** **static** **int** *myCount* = 0;

**public** MyRunnable(){

}

**public** **void** run() {

**while**(MyRunnable.*myCount* <= 10){

**try**{

System.***out***.println("Expl Thread: "+(++MyRunnable.*myCount*));

Thread.*sleep*(100);

} **catch** (InterruptedException iex) {

System.***out***.println("Exception in thread: "+iex.getMessage());

}

}

}

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

System.***out***.println("Starting Main Thread...");

MyRunnable mrt = **new** MyRunnable();

Thread t = **new** Thread(mrt);

t.start();

**while**(MyRunnable.*myCount* <= 10){

**try**{

System.***out***.println("Main Thread: "+(++MyRunnable.*myCount*));

Thread.*sleep*(100);

} **catch** (InterruptedException iex){

System.***out***.println("Exception in main thread: "+iex.getMessage());

}

}

System.***out***.println("End of Main Thread...");

}

}

**Output:**

Starting Main Thread...

Main Thread: 1

Expl Thread: 2

Expl Thread: 3

Main Thread: 4

Main Thread: 5

Expl Thread: 6

Main Thread: 7

Expl Thread: 8

Expl Thread: 9

Main Thread: 9

Expl Thread: 10

Main Thread: 11

End of Main Thread...

**2.Write a program in Java to demonstrate sleep() and wait()**

**package** Myclass;

**public** **class** Myclass {

**private** **static** Object *LOCK* = **new** Object();

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

{

Thread.*sleep*(1000);

System.***out***.println("Thread '" + Thread.*currentThread*().getName() +

"' is woken after sleeping for 1 second");

**synchronized** (*LOCK*)

{

*LOCK*.wait(1000);

System.***out***.println("Object '" + *LOCK* + "' is woken after" + " waiting for 1 second");

}

}

}

**Output:**

Thread 'main' is woken after sleeping for 1 second

Object 'java.lang.Object@cac736f' is woken after waiting for 1 second

1. **Write a program in Java to demonstrate synchronization**

**public synchronized void** sendMessage(String message) {

System.***out***.println("Sending " + message);

**try** {

Thread.*sleep*(1000); // Simulating some processing time

} **catch** (InterruptedException e) {

e.printStackTrace();

}

System.***out***.println(message + " sent");

}

}

**class** SenderThread **extends** Thread {

**private** MessageSender sender;

**private** String message;

**public** SenderThread(MessageSender sender, String message) {

**this**.sender = sender;

**this**.message = message;

}

@Override

**public void** run() {

sender.sendMessage(message);

}

}

**public class** Synchronisation {

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

MessageSender sender = **new** MessageSender();

SenderThread thread1 = **new** SenderThread(sender, "hi");

SenderThread thread2 = **new** SenderThread(sender, "bye");

thread1.start();

thread2.start();

**try** {

thread1.join();

thread2.join();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

**OUTPUT:**

Sending hi

hi sent

Sending bye

bye sent

1. **Writing a program in Java to demonstrate try and catch**

**package** demonstrate;

**public** **class** Myclass {

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

**int**[] array = **new** **int**[3];

**try**

{

array[7] = 3;

}

**catch** (ArrayIndexOutOfBoundsException e)

{

System.***out***.println("Array index is out of bounds!");

}

**finally**

{

System.***out***.println("The array is of size " + array.length);

}

}

}

**Outpt:**

Array index is out of bounds!

The array is of size 3

1. **Writing a program in Java to demonstrate the throw keyword**

**public** **class** throwKeyword{

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

{

**Int** a=45,b=0,rs;

**try**

{

**if**(b==0)

**throw**(**new** ArithmeticException("Can't divide by zero."));

**else**

{

rs = a / b;

System.***out***.print("\n\tThe result is : " + rs);

}

}

**catch**(ArithmeticException Ex)

{

System.***out***.print("\n\tError : " + Ex.getMessage());

}

System.***out***.print("\n\tEnd of program.");

}

}

**Output:**

Error : Can't divide by zero.

End of program.

**6.Writing a program in Java to demonstrate exception handling**

**package** demonstrate;

**public** **class** ExceptionHandling {

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

**try** {

**int**[] numbers = { 1, 2, 3 };

System.***out***.println("Accessing element at index 5: " +

numbers[5]);

}

**catch** (ArrayIndexOutOfBoundsException e) {

System.***out***.println("Array index is out of bounds!");

}

**finally** {

System.***out***.println("Exception handling completed.");

}

}

}

**Output:**

Array index is out of bounds!

Exception handling completed.

**7.Writing a program in Java to create a file**

**package** demonstrate;

**import** java.io.File;

**import** java.io.FileOutputStream;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.nio.charset.StandardCharsets;

**import** java.nio.file.Files;

**import** java.nio.file.Paths;

**import** java.nio.file.StandardOpenOption;

**import** java.util.Arrays;

**import** java.util.List;

**public** **class** NewFile {

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

{

*createFileUsingFileClass*();

*createFileUsingFileOutputStreamClass*();

*createFileIn\_NIO*();

}

**private** **static** **void** createFileUsingFileClass() **throws** IOException

{

File file = **new** File("c://temp//testFile1.txt");

//Create the file

**if** (file.createNewFile()){

System.***out***.println("File is created!");

}**else**{

System.***out***.println("File already exists.");

}

//Write Content

FileWriter writer = **new** FileWriter(file);

writer.write("Test data");

writer.close();

}

**private** **static** **void** createFileUsingFileOutputStreamClass() **throws** IOException

{

String data = "Test data";

FileOutputStream out = **new** FileOutputStream("c://temp//testFile2.txt");

out.write(data.getBytes());

out.close();

}

**private** **static** **void** createFileIn\_NIO() **throws** IOException

{

String data = "Test data";

Files.*write*(Paths.*get*("c://temp//testFile3.txt"), data.getBytes());

List<String> lines = Arrays.*asList*("1st line", "2nd line");

Files.*write*(Paths.*get*("file6.txt"),

lines,

StandardCharsets.***UTF\_8***,

StandardOpenOption.***CREATE***,

StandardOpenOption.***APPEND***);

}

}

**Output:**

File created.

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

**import** java.nio.charset.StandardCharsets;

**import** java.nio.file.\*;

**import** java.io.\*;

**public** **class** ReadFileIntoList {

**public** **static** List<String> readFileInList(String fileName)

{

List<String> lines = Collections.*emptyList*();

**try**

{

lines =

Files.*readAllLines*(Paths.*get*(fileName), StandardCharsets.***UTF\_8***);

}

**catch** (IOException e)

{

e.printStackTrace();

}

**return** lines;

}

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

{

List l = *readFileInList*("c://temp//testFile2.txt");

Iterator<String> itr = l.iterator();

**while** (itr.hasNext())

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

}

}

1. **import** java.io.BufferedReader;

**import** java.io.File;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** TextFileModification {

**static** **void** modifyFile(String filePath, String oldString, String newString)

{

File fileToBeModified = **new** File(filePath);

String oldContent = "";

BufferedReader reader = **null**;

FileWriter writer = **null**;

**try**

{

reader = **new** BufferedReader(**new** FileReader(fileToBeModified));

String line = reader.readLine();

**while** (line != **null**)

{

oldContent = oldContent + line + System.*lineSeparator*();

line = reader.readLine();

}

String newContent = oldContent.replaceAll(oldString, newString);

writer = **new** FileWriter(fileToBeModified);

writer.write(newContent);

}

**catch** (IOException e)

{

e.printStackTrace();

}

**finally**

{

**try**

{

reader.close();

writer.close();

}

**catch** (IOException e)

{

e.printStackTrace();

}

}

}

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

{

*modifyFile*("c://temp//testFile2.txt", "85", "95");

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

}

}

**Output:**

done

**5.import** java.io.IOException;

**import** java.nio.file.\*;

**public** **class** Test {

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

{

**try**

{

Files.*deleteIfExists*(Paths.*get*("c://temp//testFile2.txt"));

}

**catch**(NoSuchFileException e)

{

System.***out***.println("No such file/directory exists");

}

**catch**(DirectoryNotEmptyException e)

{

System.***out***.println("Directory is not empty.");

}

**catch**(IOException e)

{

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

}

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

}

}

**Output:**

Deletion successful.

**8.Writing a program in Java to demonstrate the uses of classes and objects**

**(1) package** createobjects;

**public class** Dog

{

String name;

String breed;

**int** age;

String color;

**public** Dog(String name, String breed, **int** age, String color)

{

**this**.name = name;

**this**.breed = breed;

**this**.age = age;

**this**.color = color;

}

**public** String getName()

{

**return** name;

}

**public** String getBreed()

{

**return** breed;

}

**public int** getAge()

{

**return** age;

}

**public** String getColor()

{

**return** color;

}

@Override

**public** String toString()

{

**return**("Hi my name is "+ **this**.getName()+ ".\nMy breed,age and color

are " + **this**.getBreed()+", " + **this**.getAge()+ ", and"+ **this**.getColor() +

".");

}

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

{

Dog scott = **new** Dog("Scott","papillon", 5, "black");

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

}

}

**OUTPUT:**

Hi my name is Scott.

My breed,age and color are papillon, 5, andblack.

(2) **package** createobjects;

**class** Sum

{

**public int** sum(**int** x, **int** y)

{

**return** (x + y);

}

**public int** sum(**int** x, **int** y, **int** z)

{

**return** (x + y + z);

}

**public double** sum(**double** x, **double** y) {

**return** (x + y);

}

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

{

Sum s = **new** Sum();

System.***out***.println(s.sum(10, 20));

System.***out***.println(s.sum(10, 20, 30));

System.***out***.println(s.sum(10.5, 20.5));

}

}

**OUTPUT:**

30

60

31.0

(3) **package** createobjects;

**class** Bicycle

{

**public int** gear;

**public int** speed;

**public** Bicycle(**int** gear, **int** speed)

{

**this**.gear = gear;

**this**.speed = speed;

}

**public void** applyBrake(**int** decrement)

{

speed -= decrement;

}

**public void** speedUp(**int** increment)

{

speed += increment;

}

**public** String toString()

{

**return**("No of gears are " + gear + "\n" + "speed of bicycle is " +

speed);

}

}

**class** MountainBike **extends** Bicycle

{

**public int** seatHeight;

**public** MountainBike(**int** gear,**int** speed,**int** startHeight)

{

**super**(gear, speed);

seatHeight = startHeight;

}

**public void** setHeight(**int** newValue)

{

seatHeight = newValue;

}

@Override

**public** String toString()

{

**return** (**super**.toString()+

"\nseat height is "+seatHeight);

}

}

**public class** Test{

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

{

MountainBike mb = **new** MountainBike(3, 100, 25);

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

}

}

**Output:**

No of gears are 3

Speed of bicycle is 100

Seat height is 25

(4)public class Encapsulate

{

private String Name;

private int Roll;

private int Age;

public int getAge()

{

return Age;

}

public String getName()

{

return Name;

}

public int getRoll()

{

return Roll;

}

public void setAge( int newAge)

{

Age = newAge;

}

public void setName(String newName)

{

Name = newName;

} public void setRoll( int newRoll)

{

Roll = newRoll;

}

}

public class TestEncapsulation

{

public static void main (String[] args)

{

Encapsulate obj = new Encapsulate();

obj.setName("Harsh");

obj.setAge(19);

obj.setRoll(51);

System.out.println("My name: " + obj.getName());

System.out.println("My age: " + obj.getAge());

System.out.println("My roll: " + obj.getRoll());

}

}

**Output:**

my name “harsh”

My age “19”

My roll “51”

5)abstract class Shape

{

String color;

abstract double area();

public abstract String toString();

public Shape(String color)

{

System.out.println("Shape constructor called");

this.color = color;

}

public String getColor()

{ return color;

}

}

class Circle extends Shape

{

double radius;

public Circle(String color,double radius)

{

super(color);

System.out.println("Circle constructor called");

this.radius = radius;

}

@Override

double area()

{

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

}

@Override

public String toString()

{

return "Circle color is " + super.color + "and area is : " + area();

}

}

class Rectangle extends Shape

{

double length;

double width;

public Rectangle(String color,double length,double width)

{

super(color);

System.out.println("Rectangle constructor called");

this.length = length;

this.width = width; }

@Override

double area()

{

return length\*width;

}

@Override

public String toString()

{

return "Rectangle color is " + super.color +

"and area is : " + area();

}

}

public class Test

{

public static void main(String[] args)

{

Shape s1 = new Circle("Red", 2.2);

Shape s2 = new Rectangle("Yellow", 2, 4);

System.out.println(s1.toString());

System.out.println(s2.toString());

}

}

**Output:** shape constructor called

Circle constructor called

Shape constructoe called

Rectangle constructor called

Circle color redand area is:15.205308443374602

Rectangle color isyellowed color area is:8.0

1. **Writing a program in Java to resolve the diamond problem using OOPs’ concept**s

**package** diamondproblem;

**interface** First

{

**default void** show()

{

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

}

}

**interface** Second

{

**default void** show()

{

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

}

}

**public class** TestClass **implements** First, Second

{

**public void** show()

{

First.**super**.show();

Second.**super**.show();

}

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

{

TestClass ob = **new** TestClass();

ob.show();

}

}