**Type Casting**

**public** **class** typeCasting {

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

System.***out***.println("Implicit Type Casting");

**char** a='A';

System.***out***.println("Value of a: "+a);

**int** b=a;

System.***out***.println("Value of b: "+b);

**float** c=a;

System.***out***.println("Value of c: "+c);

**long** d=a;

System.***out***.println("Value of d: "+d);

**double** e=a;

System.***out***.println("Value of e: "+e);

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

System.***out***.println("Explicit Type Casting");

**double** x=45.5;

**int** y=(**int**)x;

System.***out***.println("Value of x: "+x);

System.***out***.println("Value of y: "+y);

}

}

1. **Access Modifiers**

**class** defAccessSpecifier

{

**void** display()

{

System.***out***.println("You are using defalut access specifier");

}

}

**public** **class** accessSpecifiers1 {

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

System.***out***.println("Dafault Access Specifier");

defAccessSpecifier obj = **new** defAccessSpecifier();

obj.display();

}

}

//2. using private access specifiers

**class** priaccessspecifier

{

**private** **void** display()

{

System.***out***.println("You are using private access specifier");

}

}

**public** **class** accessSpecifiers2 {

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

System.***out***.println("Private Access Specifier");

priaccessspecifier obj = **new** priaccessspecifier();

}

}

//3. using protected access specifiers

**package** pack1;

**public** **class** proaccessspecifiers {

**protected** **void** display()

{

System.***out***.println("This is protected access specifier");

}

}

//create another package

**package** pack2;

**import** pack1.\*;

**public** **class** accessSpecifiers3 **extends** proaccessspecifiers {

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

accessSpecifiers3 obj = **new** accessSpecifiers3 ();

obj.display();

}

}

//4. using public access specifiers

**package** pack1;

**public** **class** pubaccessspecifiers {

**public** **void** display()

{

System.***out***.println("This is Public Access Specifiers");

}

}

//create another package

**package** pack2;

**import** pack1.\*;

**public** **class** accessSpecifiers4 {

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

pubaccessspecifiers obj = **new** pubaccessspecifiers();

obj.display();

}

}

1. **Methods**

**public** **class** methodExecution {

**public** **int** multipynumbers(**int** a,**int** b) {

**int** z=a\*b;

**return** z;

}

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

methodExecution b=**new** methodExecution();

**int** ans= b.multipynumbers(10,3);

System.***out***.println("Multipilcation is :"+ans);

}

//call by value

**public** **class** callMethod {

**int** val=150;

**int** operation(**int** val) {

val =val\*10/100;

**return**(val);

}

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

callMethod d = **new** callMethod();

System.***out***.println("Before operation value of data is "+d.val);

d.operation(100);

System.***out***.println("After operation value of data is "+d.val);

}

}

//method overloading

**public** **class** overloadMethod {

**public** **void** area(**int** b,**int** h)

{

System.***out***.println("Area of Triangle : "+(0.5\*b\*h));

}

**public** **void** area(**int** r)

{

System.***out***.println("Area of Circle : "+(3.14\*r\*r));

}

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

{

overloadMethod ob=**new** overloadMethod();

ob.area(10,12);

ob.area(5);

}

}

1. **Constructors**

**class** EmpInfo{

**int** id;

String name;

**void** display() {

System.***out***.println(id+" "+name);

}

}

**public** **class** constructorDemo {

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

EmpInfo emp1=**new** EmpInfo();

EmpInfo emp2=**new** EmpInfo();

emp1.display();

emp2.display();

}

}

//parameterized constructor

**class** Std{

**int** id;

String name;

Std(**int** i,String n)

{

id=i;

name=n;

}

**void** display() {

System.***out***.println(id+" "+name);

}

}

**public** **class** paramConstrDemo {

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

Std std1=**new** Std(2,"Alex");

Std std2=**new** Std(10,"Annie");

std1.display();

std2.display();

}

}

1. **Collections**

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

**public** **class** collectionAssisted {

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

//creating arraylist

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

ArrayList<String> city=**new** ArrayList<String>();

city.add("Bangalore");//

city.add("Delhi");

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

//creating vector

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

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

Vector<Integer> vec = **new** Vector();

vec.addElement(15);

vec.addElement(30);

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

//creating linkedlist

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

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

LinkedList<String> names=**new** LinkedList<String>();

names.add("Alex");

names.add("John");

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

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

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

//creating hashset

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

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

HashSet<Integer> set=**new** HashSet<Integer>();

set.add(101);

set.add(103);

set.add(102);

set.add(104);

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

//creating linkedhashset

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

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

LinkedHashSet<Integer> set2=**new** LinkedHashSet<Integer>();

set2.add(11);

set2.add(13);

set2.add(12);

set2.add(14);

System.***out***.println(set2);

}

}

}

1. **Map**

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

**public** **class** mapDemo {

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

// map

//Hashmap

HashMap<Integer,String> hm=**new** HashMap<Integer,String>();

hm.put(1,"Tim");

hm.put(2,"Mary");

hm.put(3,"Catie");

System.***out***.println("\nThe elements of Hashmap are ");

**for**(Map.Entry m:hm.entrySet()){

System.***out***.println(m.getKey()+" "+m.getValue());

}

//HashTable

Hashtable<Integer,String> ht=**new** Hashtable<Integer,String>();

ht.put(4,"Ales");

ht.put(5,"Rosy");

ht.put(6,"Jack");

ht.put(7,"John");

System.***out***.println("\nThe elements of HashTable are ");

**for**(Map.Entry n:ht.entrySet()){

System.***out***.println(n.getKey()+" "+n.getValue());

}

//TreeMap

TreeMap<Integer,String> map=**new** TreeMap<Integer,String>();

map.put(8,"Annie");

map.put(9,"Carlotte");

map.put(10,"Catie");

System.***out***.println("\nThe elements of TreeMap are ");

**for**(Map.Entry l:map.entrySet()){

System.***out***.println(l.getKey()+" "+l.getValue());

}

}

}

1. **INNER Class**

**public** **class** innerClassAssisted1 {

**private** String msg="Welcome to Java";

**class** Inner{

**void** hello(){System.***out***.println(msg+", Let us start learning Inner Classes");}

}

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

innerClassAssisted1 obj=**new** innerClassAssisted1();

innerClassAssisted1.Inner in=obj.**new** Inner();

in.hello();

}

}

**public** **class** innerClassAssisted2 {

**private** String msg="Inner Classes";

**void** display(){

**class** Inner{

**void** msg(){

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

}

}

Inner l=**new** Inner();

l.msg();

}

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

innerClassAssisted2 ob=**new** innerClassAssisted2 ();

ob.display();

}

}

//anonymous inner class

**abstract** **class** AnonymousInnerClass {

**public** **abstract** **void** display();

}

**public** **class** innerClassAssisted3 {

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

AnonymousInnerClass i = **new** AnonymousInnerClass() {

**public** **void** display() {

System.***out***.println("Anonymous Inner Class");

}

};

i.display();

}

}

1. **Strings**

**public** **class** stringDemo {

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

//methods of strings

System.***out***.println("Methods of Strings");

String sl=**new** String("Hello World");

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

//substring

String sub=**new** String("Welcome");

System.***out***.println(sub.substring(2));

//String Comparison

String s1="Hello";

String s2="Heldo";

System.***out***.println(s1.compareTo(s2));

//IsEmpty

String s4="";

System.***out***.println(s4.isEmpty());

//toLowerCase

String s5="Hello";

System.***out***.println(s1.toLowerCase());

//replace

String s6="Heldo";

String replace=s2.replace('d', 'l');

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

//equals

String x="Welcome to Java";

String y="WeLcOmE tO JaVa";

System.***out***.println(x.equals(y));

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

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

//Creating StringBuffer and append method

StringBuffer s=**new** StringBuffer("Welcome to Java!");

s.append("Enjoy your learning");

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

//insert method

s.insert(0, 'w');

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

//replace method

StringBuffer sb=**new** StringBuffer("Hello");

sb.replace(0, 2, "hEl");

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

//delete method

sb.delete(0, 1);

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

//StringBuilder

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

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

StringBuilder sb1=**new** StringBuilder("Happy");

sb1.append("Learning");

System.***out***.println(sb1);

System.***out***.println(sb1.delete(0, 1));

System.***out***.println(sb1.insert(1, "Welcome"));

System.***out***.println(sb1.reverse());

//conversion

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

System.***out***.println("Conversion of Strings to StringBuffer and StringBuilder");

String str = "Hello";

// conversion from String object to StringBuffer

StringBuffer sbr = **new** StringBuffer(str);

sbr.reverse();

System.***out***.println("String to StringBuffer");

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

// conversion from String object to StringBuilder

StringBuilder sbl = **new** StringBuilder(str);

sbl.append("world");

System.***out***.println("String to StringBuilder");

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

}

}

1. **Arrays**

**public** **class** arrayAssisted {

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

//single-dimensional array

**int** a[]= {10,20,30,40,50};

**for**(**int** i=0;i<5;i++) {

System.***out***.println("Elements of array a: "+a[i]);

}

//multidimensional array

**int**[][] b = {

{2, 4, 6, 8},

{3, 6, 9} };

System.***out***.println("\nLength of row 1: " + b[0].length);

}

}

**10 . Regular Expressions**

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

**public** **class** regularExpnAssisted {

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

String pattern = "[a-z]+";

String check = "Regular Expressions";

Pattern p = Pattern.*compile*(pattern);

Matcher c = p.matcher(check);

**while** (c.find())

System.***out***.println( check.substring( c.start(), c.end() ) );

}

}

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

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();

}

}

1. **Writing the program in Java to demonstrate sleep() and wait()**

**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");

}

}

}

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

import java.io.\*;

import java.util.\*;

class Sender

{

public void send(String msg)

{

System.out.println("Sending\t" + msg );

try

{

Thread.sleep(1000);

}

catch (Exception e)

{

System.out.println("Thread interrupted.");

}

System.out.println("\n" + msg + "Sent");

}

}

class ThreadedSend extends Thread

{

private String msg;

private Thread t;

Sender sender;

ThreadedSend(String m, Sender obj)

{

msg = m;

sender = obj;

}

public void run()

{

synchronized(sender)

{

sender.send(msg);

}

}

}

class SyncDemo

{

public static void main(String args[])

{

Sender snd = new Sender();

ThreadedSend S1 =

new ThreadedSend( " Hi " , snd );

ThreadedSend S2 =

new ThreadedSend( " Bye " , snd );

S1.start();

S2.start();

try

{

S1.join();

S2.join();

}

catch(Exception e)

{

System.out.println("Interrupted");

}

}

}

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

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);

}

}

}

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

public class ThrowDemo

{

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.");

}

}

16 . **Writing a program in Java to demonstrate the finally keyword**

public class FinallyBlockDemo

{

public static void main(String[] args)

{

int a=45,b=0,rs=0;

try

{

rs = a / b;

}

catch(ArithmeticException Ex)

{

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

}

finally

{

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

}

}

}

1. **Writing a program in Java to demonstrate custom exceptions**

class MyException extends Exception

{

public MyException(String s)

{

super(s);

}

}

public class Main

{

public static void main(String args[])

{

try

{

throw new MyException("temp");

}

catch (MyException ex)

{

System.out.println("Caught");

System.out.println(ex.getMessage());

}

}

}

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

class MyException extends Exception{

String str1;

MyException(String str2) {

str1=str2;

}

public String toString(){

return ("MyException Occurred: "+str1) ;

}

}

class Example1{

public static void main(String args[]){

try{

System.out.println("Starting of try block");

// I'm throwing the custom exception using throw

throw new MyException("This is My error Message");

}

catch(MyException exp){

System.out.println("Catch Block") ;

System.out.println(exp) ;

}

}

}

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

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 CreateNewFile

{

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);

}

}

1. **Writing a program in Java to read a file**

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. **Writing a program in Java to update a file**

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class TextFileModificationProgram

{

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");

}

}

**22 . Writing a program in Java to delete a file**

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.");

}

}

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

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());

}

}

24 . **Writing a program in Java to demonstrate the uses of polymorphism**

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));

}

}

25 .  **Writing a program in Java to demonstrate the uses of inheritance**

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());

}

}

**26 . Writing a program in Java to demonstrate the uses of encapsulation**

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());

}

}

**27 . Writing a program in Java to demonstrate the uses of abstraction**

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());

}

}

**28 . Writing a program in Java to resolve the diamond problem using OOPs’ concepts**

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();

}

}

1. **Writing a program in Java for array rotation**

class RotateArray {

public void rotate(int[] nums, int k) {

if(k > nums.length)

k=k%nums.length;

int[] result = new int[nums.length];

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

result[i] = nums[nums.length-k+i];

}

int j=0;

for(int i=k; i<nums.length; i++){

result[i] = nums[j];

j++;

}

System.arraycopy( result, 0, nums, 0, nums.length );

}

}

public class Main

{

public static void main(String[] args) {

RotateArray r = new RotateArray();

int arr[] = { 1, 2, 3, 4, 5, 6, 7 };

r.rotate(arr, 5);

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

System.out.print(arr[i]+" ");

}

}

}

**34 . Writing a program in Java to understand order statistics**

class KthSmallst

{

int kthSmallest(int arr[], int l, int r, int k)

{

if (k > 0 && k <= r - l + 1)

{

int pos = randomPartition(arr, l, r);

if (pos-l == k-1)

return arr[pos];

if (pos-l > k-1)

return kthSmallest(arr, l, pos-1, k);

return kthSmallest(arr, pos+1, r, k-pos+l-1);

}

return Integer.MAX\_VALUE;

}

void swap(int arr[], int i, int j)

{

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

int partition(int arr[], int l, int r)

{

int x = arr[r], i = l;

for (int j = l; j <= r - 1; j++)

{

if (arr[j] <= x)

{

swap(arr, i, j);

i++;

}

}

swap(arr, i, r);

return i;

}

int randomPartition(int arr[], int l, int r)

{

int n = r-l+1;

int pivot = (int)(Math.random()) \* (n-1);

swap(arr, l + pivot, r);

return partition(arr, l, r);

}

}

public class Main

{

public static void main(String[] args) {

KthSmallst ob = new KthSmallst();

int arr[] = {12, 3, 5, 7, 4, 19, 26};

int n = arr.length,k = 4;

System.out.println("K'th smallest element is "+ ob.kthSmallest(arr, 0, n-1, k));

}

}

**35 .Writing the program in Java to understand range queries**

public class RangeQueries

{

static int k = 16;

static int N = 100000;

static long table[][] = new long[N][k + 1];

static void buildSparseTable(int arr[], int n)

{

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

table[i][0] = arr[i];

for (int j = 1; j <= k; j++)

for (int i = 0; i <= n - (1 << j); i++)

table[i][j] = table[i][j - 1] + table[i + (1 << (j - 1))][j - 1];

}

static long query(int L, int R)

{

long answer = 0;

for (int j = k; j >= 0; j--)

{

if (L + (1 << j) - 1 <= R)

{

answer = answer + table[L][j];

L += 1 << j;

}

}

return answer;

}

public static void main(String args[])

{

int arr[] = { 3, 7, 2, 5, 8, 9 };

int n = arr.length;

buildSparseTable(arr, n);

System.out.println(query(0, 5));

System.out.println(query(3, 5));

System.out.println(query(2, 4));

}

}

1. **Writing the program in Java to multiply two matrices**

public class MultiplyMatrices

{

public static void main(String[] args)

{

int r1 = 2, c1 = 3;

int r2 = 3, c2 = 2;

int[][] firstMatrix = { {3, -2, 5}, {3, 0, 4} };

int[][] secondMatrix = { {2, 3}, {-9, 0}, {0, 4} };

int[][] product = multiplyMatrices(firstMatrix, secondMatrix, r1, c1, c2);

displayProduct(product);

}

public static int[][] multiplyMatrices(int[][] firstMatrix, int[][] secondMatrix, int r1, int c1, int c2)

{

int[][] product = new int[r1][c2];

for(int i = 0; i < r1; i++)

{

for (int j = 0; j < c2; j++)

{

for (int k = 0; k < c1; k++)

{

product[i][j] += firstMatrix[i][k] \* secondMatrix[k][j];

}

}

}

return product;

}

public static void displayProduct(int[][] product)

{

System.out.println("Product of two matrices is: ");

for(int[] row : product)

{

for (int column : row)

{

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

}

System.out.println();

}

}

1. **Writing the program in Java to understand the working of the singly linked list**

import java.io.\*;

public class LinkedList

{

Node head; // head of list

static class Node

{

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

// Method to insert a new node

public static LinkedList insert(LinkedList list, int data)

{

// Create a new node with given data

Node new\_node = new Node(data);

new\_node.next = null;

// If the Linked List is empty, then make the new node as head

if (list.head == null)

{

list.head = new\_node;

}

else

{

// Else traverse till the last node and insert the new\_node there

Node last = list.head;

while (last.next != null)

{

last = last.next;

}

// Insert the new\_node at last node

last.next = new\_node;

}

return list;

}

public static void printList(LinkedList list)

{

Node currNode = list.head;

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

// Traverse through the LinkedList

while (currNode != null)

{

// Print the data at current node

System.out.print(currNode.data + " ");

// Go to next node

currNode = currNode.next;

}

System.out.println();

}

// Method to delete a node in the LinkedList by KEY

public static LinkedList deleteByKey(LinkedList list, int key)

{

// Store head node

Node currNode = list.head, prev = null;

If (currNode != null && currNode.data == key)

{

list.head = currNode.next; // Changed head

System.out.println(key + " found and deleted");

return list;

}

while (currNode != null && currNode.data != key)

{

prev = currNode;

currNode = currNode.next;

}

if (currNode != null)

{

prev.next = currNode.next;

System.out.println(key + " found and deleted");

}

if (currNode == null)

{

System.out.println(key + " not found");

}

return list;

}

// method to create a Singly linked list with n nodes

public static void main(String[] args)

{

/\* Start with the empty list. \*/

LinkedList list = new LinkedList();

// Insert the values

list = insert(list, 1);

list = insert(list, 2);

list = insert(list, 3);

list = insert(list, 4);

list = insert(list, 5);

list = insert(list, 6);

list = insert(list, 7);

list = insert(list, 8);

// Print the LinkedList

printList(list);

// Delete node with value 1

deleteByKey(list, 1);

// Print the LinkedList

printList(list);

// Delete node with value 4

deleteByKey(list, 4);

// Print the LinkedList

printList(list);

// Delete node with value 10

deleteByKey(list, 10);

// Print the LinkedList

printList(list);

}

}

1. **Writing the program in Java . You need to now write the program to insert a new element in a sorted circular linked list.**

public class LinkedList

{

static class Node

{

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

Node head;

LinkedList()

{

head = null;

}

void sortedInsert(Node new\_node)

{

Node current = head;

if (current == null)

{

new\_node.next = new\_node;

head = new\_node;

}

else if (current.data >= new\_node.data)

{

while (current.next != head)

current = current.next;

current.next = new\_node;

new\_node.next = head;

head = new\_node;

}

else

{

while (current.next != head && current.next.data < new\_node.data)

current = current.next;

new\_node.next = current.next;

current.next = new\_node;

}

}

void printList()

{

if (head != null)

{

Node temp = head;

do

{

System.out.print(temp.data + " ");

temp = temp.next;

} while (temp != head);

}

}

public static void main(String[] args)

{

LinkedList list = new LinkedList();

int arr[] = new int[] {12, 56, 2, 11, 1, 90};

Node temp = null;

for (int i = 0; i < 6; i++)

{

temp = new Node(arr[i]);

list.sortedInsert(temp);

}

list.printList();

}

}

**39. Writing the program in Java . You need to write the program to traverse a doubly linked list in the forward and backward directions.**

public class DLL

{

     Node head;

class Node

{

         int data;

         Node prev;

         Node next;

Node(int d)

{

data = d;

}

     }

public void push(int new\_data)

     {

Node new\_Node = new Node(new\_data);

new\_Node.next = head;

         new\_Node.prev = null;

if (head != null)

             head.prev = new\_Node;

head = new\_Node;

     }

public void InsertAfter(Node prev\_Node, int new\_data)

     {

if (prev\_Node == null)

{

             System.out.println("The given previous node cannot be NULL ");

             return;

         }

Node new\_node = new Node(new\_data);

new\_node.next = prev\_Node.next;

prev\_Node.next = new\_node;

new\_node.prev = prev\_Node;

if (new\_node.next != null)

             new\_node.next.prev = new\_node;

     }

     void append(int new\_data)

     {

Node new\_node = new Node(new\_data);

   Node last = head;

new\_node.next = null;

if (head == null)

{

             new\_node.prev = null;

             head = new\_node;

             return;

         }

while (last.next != null)

             last = last.next;

last.next = new\_node;

new\_node.prev = last;

     }

public void printlist(Node node)

     {

         Node last = null;

         System.out.println("Traversal in forward Direction");

         while (node != null)

{

             System.out.print(node.data + " ");

             last = node;

             node = node.next;

         }

         System.out.println();

         System.out.println("Traversal in reverse direction");

         while (last != null)

{

             System.out.print(last.data + " ");

             last = last.prev;

         }

     }

public static void main(String[] args)

     {

DLL dll = new DLL

dll.append(6);

dll.push(7);

dll.push(1);

dll.append(4);

dll.InsertAfter(dll.head.next, 8);

   System.out.println("Created DLL is: ");

         dll.printlist(dll.head);

     }

}

**40. Writing a program in Java implementing the quick sort algorithm**

**There are two ways you can perform this step; you can create a new Java project, or you can create a new Java class in the existing project. It is preferable to create a new Java class in the existing project but feel free to explore the first option. The steps mentioned below will work once you create a project in Java.**

class QuickSort  
{  
   
 int partition(int arr[], int low, int high)  
 {  
 int pivot = arr[high];  
 int i = (low-1); // index of smaller element  
 for (int j=low; j<high; j++)  
 {   
 if (arr[j] <= pivot)  
 {  
 i++;  
  
 // swap arr[i] and arr[j]  
 int temp = arr[i];  
 arr[i] = arr[j];  
 arr[j] = temp;  
 }  
 }  
  
 // swap arr[i+1] and arr[high] (or pivot)  
 int temp = arr[i+1];  
 arr[i+1] = arr[high];  
 arr[high] = temp;  
  
 return i+1;  
 }  
  
  
  
 void sort(int arr[], int low, int high)  
 {  
 if (low < high)  
 {  
  
 int pi = partition(arr, low, high);  
  
   
 sort(arr, low, pi-1);  
 sort(arr, pi+1, high);  
 }  
 }  
 static void printArray(int arr[])  
 {  
 int n = arr.length;  
 for (int i=0; i<n; ++i)  
 System.*out*.print(arr[i]+" ");  
 System.*out*.println();  
 }  
  
 // Driver program  
 public static void main(String args[])  
 {  
 int arr[] = {10, 7, 8, 9, 1, 5};  
 int n = arr.length;  
  
 QuickSort ob = new QuickSort();  
 ob.sort(arr, 0, n-1);  
  
 System.*out*.println("sorted array");  
 *printArray*(arr);  
 }  
}