**Experiment 35**

**Aim:**

Create a Graphics package that has classes and interfaces for figures Rectangle, Triangle, Square and Circle. Test the package by finding the area of these figures.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

Graphics.java:-

package graphics;

import java.util.\*;

interface shapes{

   public double ArRec();

   public double ArCir();

   public double ArSqr();

   public double ArTri();

}

public class Graphics implements shapes {

    Scanner Snr = new Scanner(System.in);

    int r,l,b,s;

    double pi = 3.14, area;

public  double ArRec(){

System.out.print("Enter the Length of Rectangle: ");

        l=Snr.nextInt();

        System.out.print("Enter the Breadth of Rectangle: ");

        b=Snr.nextInt();

area=l\*b;

return area;

}

    public double ArCir(){

        System.out.print("Enter the Radius of Circle: ");

        r = Snr.nextInt();

        area = pi \* r \* r;

        return area;

}

    public double ArSqr(){

        System.out.print("Enter the Side of the Square: ");

        s = Snr.nextInt();

        area = s \* s;

return area;

    }

    public double ArTri(){

System.out.print("Enter the Width of the Triangle: ");

        double base = Snr.nextDouble();

        System.out.print("Enter the Height of the Triangle: ");

        double height = Snr.nextDouble();

        double area = (base\* height)/2;

        return area;

    }

}

ShapeArea.java:-

import graphics.Graphics;

import java.util.\*;

public class ShapeArea{

    public static void main(String []args){

Scanner Snr = new Scanner(System.in);

Graphics Obj = new Graphics();

int choice = 0;

while(choice != 5){

System.out.println(" ");

System.out.println(":: AREA OF SHAPES ::");

System.out.println("1. Rectangle");

System.out.println("2. Circle");

System.out.println("3. Square");

System.out.println("4. Triangle");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

System.out.println("Area of Rectangle: " + Obj.ArRec());

break;

case 2:

System.out.println("Area of Circle: " + Obj.ArCir());

break;

case 3:

System.out.println("Area of Square: " + Obj.ArSqr());

break;

case 4:

System.out.println("Area of Triangle: " + Obj.ArTri());

break;

case 5:

System.exit(0);

break;

default:

System.out.println("Select a valid option!");

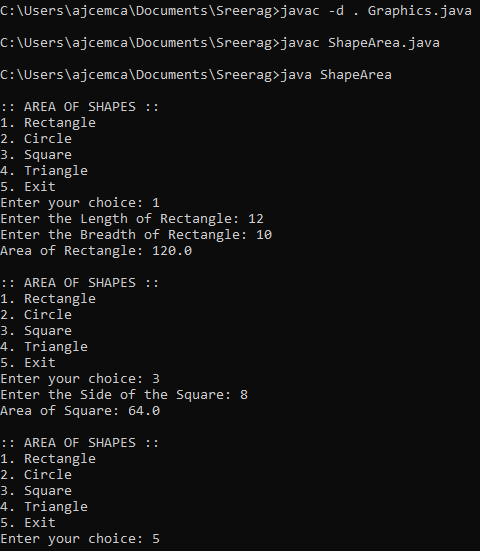
}

}

}

}

**Output:**



**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 36**

**Aim:**

Create an Arithmetic package that has classes and interfaces for the 4 basic arithmetic operations. Test the package by implementing all operations on two given numbers

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

Calculate.java:-

package arithmetic;

import java.util.\*;

interface operations{

   public double Sum();

   public double Difference();

   public double Product();

   public double Quotient();

}

public class Calculate implements operations {

    Scanner Snr = new Scanner(System.in);

    double num1,num2;

    double cal;

public  double Sum(){

System.out.print("Enter the First Number: ");

        num1=Snr.nextDouble();

        System.out.print("Enter the Second Number: ");

        num2=Snr.nextDouble();

cal=num1+num2;

return cal;

}

    public double Difference(){

        System.out.print("Enter the First Number: ");

        num1=Snr.nextDouble();

        System.out.print("Enter the Second Number: ");

        num2=Snr.nextDouble();

cal=num1-num2;

return cal;

}

    public double Product(){

        System.out.print("Enter the First Number: ");

        num1=Snr.nextDouble();

        System.out.print("Enter the Second Number: ");

        num2=Snr.nextDouble();

cal=num1\*num2;

return cal;

    }

    public double Quotient(){

System.out.print("Enter the First Number: ");

        num1=Snr.nextDouble();

        System.out.print("Enter the Second Number: ");

        num2=Snr.nextDouble();

cal=num1/num2;

return cal;

    }

}

ArthOpr.java

import arithmetic.Calculate;

import java.util.\*;

public class ArthOpr{

    public static void main(String []args){

Scanner Snr = new Scanner(System.in);

Calculate Obj = new Calculate();

int choice = 0;

while(choice != 5){

System.out.println(" ");

System.out.println(":: AREA OF SHAPES ::");

System.out.println("1. Sum");

System.out.println("2. Difference");

System.out.println("3. Product");

System.out.println("4. Quotient");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

System.out.println("Sum: " + Obj.Sum());

break;

case 2:

System.out.println("Difference: " + Obj.Difference());

break;

case 3:

System.out.println("Product: " + Obj.Product());

break;

case 4:

System.out.println("Quotient: " + Obj.Quotient());

break;

case 5:

System.exit(0);

break;

default:

System.out.println("Select a valid option!");

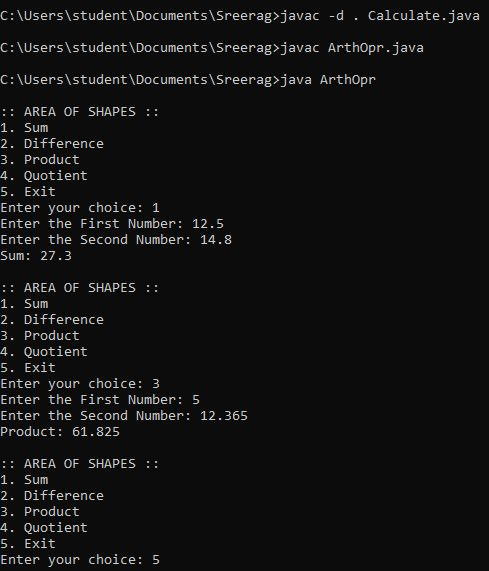
}

}

}

}

**Output**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 37**

**Aim:**

Write a user defined exception class to authenticate the user name and password.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

class user\_excep extends Exception{

public user\_excep(String str){

super(str);passw

}

}

class pass\_excep extends Exception{

public pass\_excep(String str){

super(str);

}

}

public class validate{

public static void main(String[] args){

Scanner Snr = new Scanner(System.in);

String user, pass;

System.out.print("Enter the UserName: ");

user = Snr.next();

System.out.print("Enter the Password: ");

pass = Snr.next();

int uslen = user.length();

try{

if(uslen < 8){

throw new user\_excep("User Name should be more than 8 characters");

}

if(!pass.equals("root")){

throw new pass\_excep("Incorrect Password");

}

else{

System.out.println(":: LOGIN SUCCESSFUL ::");

}

}

catch (user\_excep u){

u.printStackTrace();

}

catch (pass\_excep p){

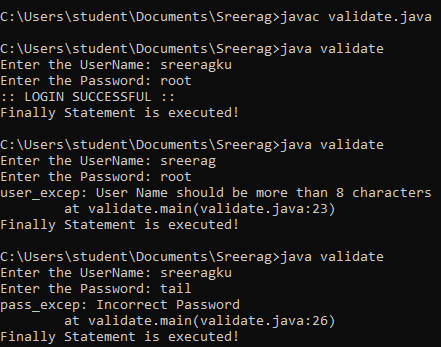
p.printStackTrace();

}

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 38**

**Aim:**

Find the average of N positive integers, raising a user defined exception for each negative input.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.Scanner;

class nIntExcep extends Exception{

    public nIntExcep(String str){

             super(str);

    }

}

public class average{

    public static void main(String[] args){

        Scanner Snr=new Scanner(System.in);

int arr[];

int sz, total=0, avg, count=0;

        System.out.print("Enter the limit: ");

sz = Snr.nextInt();

        arr = new int[sz];

        for(int i=0;i<sz;i++)

        {

            System.out.print("Enter the value: ");

int val = Snr.nextInt();

arr[i] = val;

        }

        try {

            for(int i=0;i<sz;i++) {

                if(arr[i]<0){

                    throw new nIntExcep("Numbers must be positive");

                }

                else{

                    total += arr[i];

                    count++;

}

}

            avg=total/count;

            System.out.println("Average :"+avg);

        }

catch(nIntExcep e){

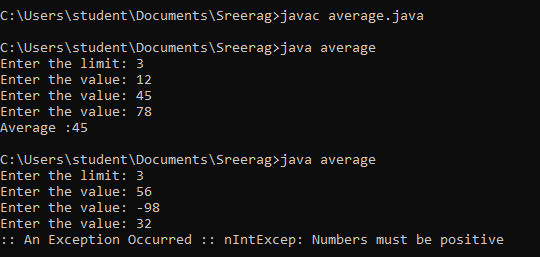
System.out.println(":: An Exception Occurred :: "+ e);

       }

    }

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 39**

**Aim:**

Define 2 classes; one for generating a multiplication table of 5 and other for displaying first N prime numbers. Implement using threads. (Thread class)

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

class MultiT extends Thread{

public void run(){

Scanner Snr = new Scanner(System.in);

System.out.println(":: MULTIPLICATION TABLE ::");

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

int num = Snr.nextInt();

System.out.println("Multiplication table of " + num + " : ");

for(int i=1;i<=10;i++){

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

}

}

}

class PrimeNo extends Thread{

public void run(){

Scanner Snr = new Scanner(System.in);

System.out.println(":: PRIME NUMBERS ::");

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

int sz = Snr.nextInt();

System.out.println("Required Prime Numbers: ");

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

if(i==1 || i==0){

continue;

}

else{

int flag= 1;

for(int j=2;j<=i/2;j++){

if(i%j == 0){

flag=0;

break;

}

}

if(flag == 1){

System.out.println(i);

}

}

}

}

}

public class ThreadClass{

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

MultiT m = new MultiT();

m.start();

m.sleep(2500);

PrimeNo p = new PrimeNo();

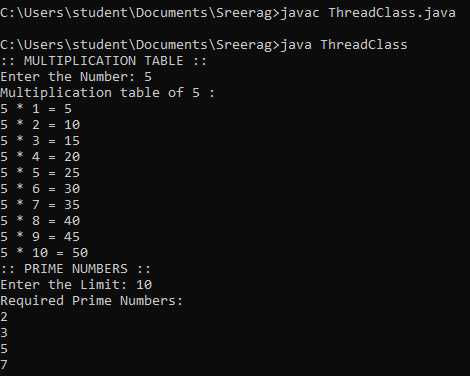
p.start();

p.sleep(2500);

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained.

**Experiment 40**

**Aim:**

Define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using threads. (Runnable Interface)

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

class Fibonacci implements Runnable{

public void run(){

Scanner Snr = new Scanner(System.in);

int n1 = 0, n2 = 1, n3;

System.out.println(":: FIBONACCI NUMBERS ::");

System.out.print("Enter the limit: ");

int sz = Snr.nextInt();

System.out.println("The Required Fibonacci Number(s): ");

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

System.out.print(n1+ " ");

n3=n1+n2;

n1=n2;

n2=n3;

}

}

}

class EvenNo implements Runnable{

public void run(){

int uplmt, lwlmt;

Scanner Snr = new Scanner(System.in);

System.out.println("\n:: EVEN NUMBERS ::");

System.out.println("Enter the Lower Limit: ");

lwlmt = Snr.nextInt();

System.out.println("Enter the Upper Limit: ");

uplmt = Snr.nextInt();

System.out.println("The Required Even Number(s): ");

for(int i=lwlmt; i<=uplmt; i++){

if(i%2 == 0){

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

}

}

}

}

public class ThreadRunner{

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

Fibonacci obj1 = new Fibonacci();

Thread a = new Thread(obj1);

a.start();

a.sleep(2500);

EvenNo obj2 = new EvenNo();

Thread b = new Thread(obj2);

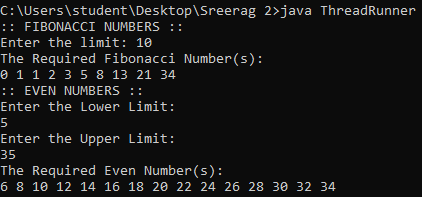
b.start();

b.sleep(2500);

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 41**

**Aim:**

Program to create a generic stack and do the Push and Pop operations.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

class Fibonacci implements Runnable{

public void run(){

Scanner Snr = new Scanner(System.in);

int n1 = 0, n2 = 1, n3;

System.out.println(":: FIBONACCI NUMBERS ::");

System.out.print("Enter the limit: ");

int sz = Snr.nextInt();

System.out.println("The Required Fibonacci Number(s): ");

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

System.out.print(n1+ " ");

n3=n1+n2;

n1=n2;

n2=n3;

}

}

}

class EvenNo implements Runnable{

public void run(){

int uplmt, lwlmt;

Scanner Snr = new Scanner(System.in);

System.out.println("\n:: EVEN NUMBERS ::");

System.out.println("Enter the Lower Limit: ");

lwlmt = Snr.nextInt();

System.out.println("Enter the Upper Limit: ");

uplmt = Snr.nextInt();

System.out.println("The Required Even Number(s): ");

for(int i=lwlmt; i<=uplmt; i++){

if(i%2 == 0){

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

}

}

}

}

public class ThreadRunner{

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

Fibonacci obj1 = new Fibonacci();

Thread a = new Thread(obj1);

a.start();

a.sleep(2500);

EvenNo obj2 = new EvenNo();

Thread b = new Thread(obj2);

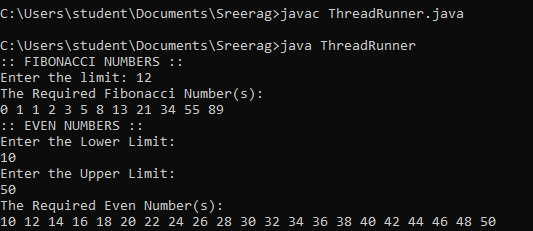
b.start();

b.sleep(2500);

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 42**

**Aim:**

Using a generic method, perform Bubble sort.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

public class BubbleSort{

int sz;

int[] Arr;

public BubbleSort(int n){

sz = n;

Arr = new int[sz];

}

public void insert(int i, int e){

Arr[i] = e;

}

public void display(int i){

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

}

public void Sort(int n){

int temp;

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

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

if(Arr[i] > Arr[j]){

temp = Arr[i];

Arr[i] = Arr[j];

Arr[j] = temp;

}

}

}

}

public static void main(String[] args){

Scanner Snr= new Scanner(System.in);

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

int size = Snr.nextInt();

BubbleSort arr = new BubbleSort(size);

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

System.out.print("Enter the element: ");

int val = Snr.nextInt();

arr.insert(i, val);

}

System.out.print("Before sorting: ");

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

arr.display(i);

}

System.out.print("\nAfter sorting: ");

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

arr.Sort(size);

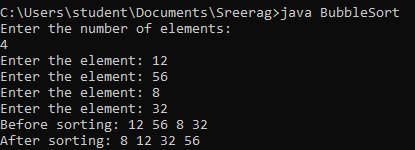
arr.display(i);

}

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 43**

**Aim:**

Maintain a list of Strings using ArrayList from the collection framework, perform built-in operations.

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.ArrayList;

import java.util.Comparator;

import java.util.Scanner;

import java.util.Collections;

public class ArrayListDemo{

public static void insert(ArrayList<String> list){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the string to be added: ");

String val = Snr.next();

list.add(val);

System.out.println("Value inserted successfully");

}

public static void deleteAll(ArrayList<String> list){

list.clear();

System.out.println("ArrayList successfully cleared");

}

public static void find(ArrayList<String> list){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the index to get: ");

int val = Snr.nextInt();

System.out.println(list.get(val));

}

public static void delete(ArrayList<String> list){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the index to delete: ");

int val = Snr.nextInt();

list.remove(val);

System.out.println("Value successfully removed");

}

public static void len(ArrayList<String> list){

System.out.println("Length of Array List: " + list.size());

}

public static void sortlist(ArrayList<String> list){

list.sort(Comparator.naturalOrder());

System.out.println("Sorted Array List: " + list);

}

public static void sortRev(ArrayList<String> list){

list.sort(Comparator.reverseOrder());

System.out.println("Reverse Order Sorted Array List: " + list);

}

public static void max(ArrayList<String> list){

String large = list.get(0);

int len = large.length();

for(int i=1; i<list.size(); i++){

String cand = list.get(i);

if(len < cand.length()){

large = list.get(i);

len = large.length();

}

}

System.out.println("Largest String: " + large);

}

public static void display(ArrayList<String> list){

System.out.println(list);

}

public static void main(String[] args){

ArrayList<String> list = new ArrayList<String>();

Scanner Snr = new Scanner(System.in);

int choice=0;

while(choice!=10){

System.out.println("");

System.out.println(":: ARRAYLIST ::");

System.out.println("1. Insert String");

System.out.println("2. Clear List");

System.out.println("3. Get Specific Element by Index");

System.out.println("4. Remove Specific Element by Index");

System.out.println("5. Length of List");

System.out.println("6. Sort Array List");

System.out.println("7. Reverse Order Sort of Array List");

System.out.println("8. Get Longest value from List");

System.out.println("9. Display");

System.out.println("10. Exit");

System.out.print("Enter your choice: ");

choice= Snr.nextInt();

switch(choice){

case 1:

insert(list);

break;

case 2:

deleteAll(list);

break;

case 3:

find(list);

break;

case 4:

delete(list);

break;

case 5:

len(list);

break;

case 6:

sortlist(list);

break;

case 7:

sortRev(list);

break;

case 8:

max(list);

break;

case 9:

display(list);

break;

case 10:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice!");

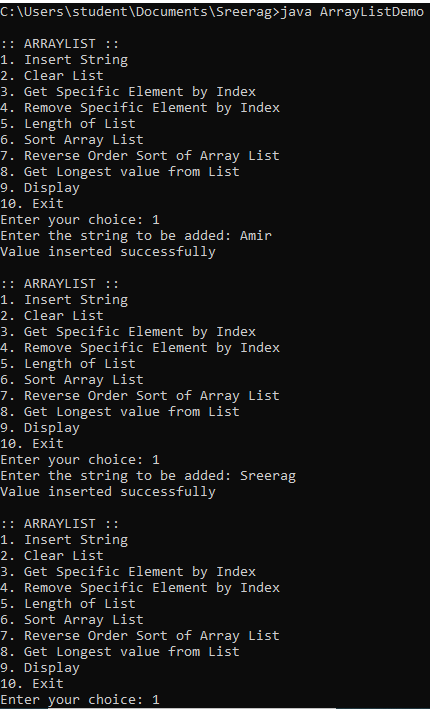
}

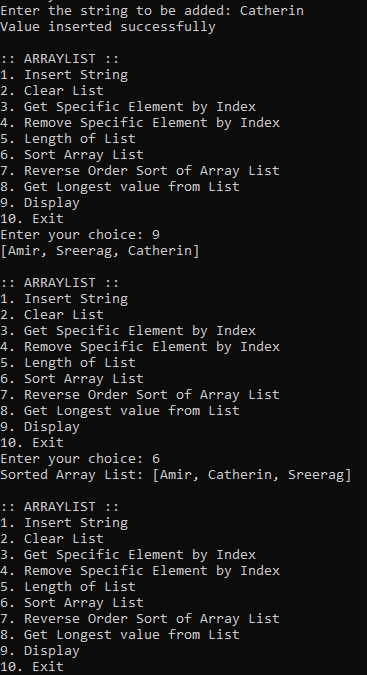
}

}

}

**Output:**

****

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 44**

**Aim:**

Program to remove all the elements from a linked list

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.LinkedList;

public class linkedList{

public static void insertBeg(LinkedList<String> list){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the string: ");

String line = Snr.next();

list.addFirst(line);

}

public static void insertEnd(LinkedList<String> list){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the string: ");

String line = Snr.next();

list.addLast(line);

}

public static void delBeg(LinkedList<String> list){

System.out.println(list.removeFirst() + " removed");

}

public static void delEnd(LinkedList<String> list){

System.out.println(list.removeLast() + " removed");

}

public static void gtfr(LinkedList<String> list){

System.out.println(list.getFirst());

}

public static void gtend(LinkedList<String> list){

System.out.println(list.getLast());

}

public static void delAll(LinkedList<String> list){

list.clear();

}

public static void main(String[] args){

Scanner Snr =new Scanner(System.in);

LinkedList<String> list = new LinkedList<String>();

int choice = 1;

while(choice != 9){

System.out.println("");

System.out.println(":: LINKED LIST ::");

System.out.println("1. Insert at Beginning");

System.out.println("2. Insert at End");

System.out.println("3. Delete from Beginning");

System.out.println("4. Delete from End");

System.out.println("5. Get the First Element");

System.out.println("6. Get the Last Element");

System.out.println("7. Delete all elements");

System.out.println("8. Display");

System.out.println("9. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

insertBeg(list);

break;

case 2:

insertEnd(list);

break;

case 3:

delBeg(list);

break;

case 4:

delEnd(list);

break;

case 5:

gtfr(list);

break;

case 6:

gtend(list);

break;

case 7:

delAll(list);

break;

case 8:

System.out.println(list);

break;

case 9:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice!");

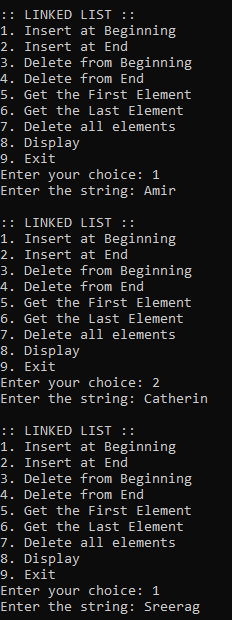
}

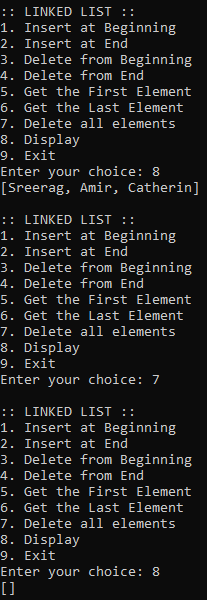
}

}

}

**Output:**

****

****

**Result:**

Output displayed successfully and CO4 was obtained’

**Experiment 45**

**Aim:**

Program to remove an object from the Stack when the position is passed as parameter

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.Stack;

public class stack{

public static void insert(Stack<String> stk){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the string: ");

String line = Snr.next();

stk.add(line);

}

public static void delete(Stack<String> stk){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the position: ");

int pos = Snr.nextInt();

pos = pos-1;

System.out.println(stk.remove(pos) + " removed");;

}

public static void display(Stack<String> stk){

System.out.println("Stack: " + stk);

}

public static void main(String[] args){

Scanner Snr =new Scanner(System.in);

Stack<String> stk = new Stack<String>();

int choice = 0;

while(choice != 4){

System.out.println(" ");

System.out.println(":: STACK OPERATIONS ::");

System.out.println("1. Insert");

System.out.println("2. Delete");

System.out.println("3. Display");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

insert(stk);

break;

case 2:

delete(stk);

break;

case 3:

display(stk);

break;

case 4:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice!");

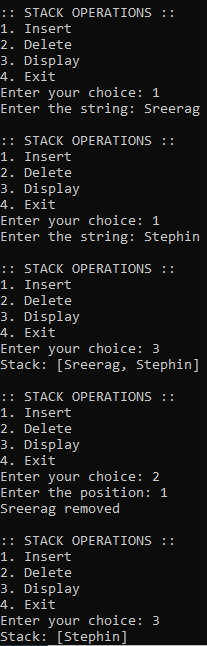
}

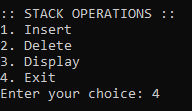
}

}

}

**Output:**

****

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 46**

**Aim:**

Program to demonstrate the creation of queue object using the PriorityQueue class

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.PriorityQueue;

public class priorqueue{

public static void insert(PriorityQueue<Integer> pq){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int num = Snr.nextInt();

pq.add(num);

}

public static void last(PriorityQueue<Integer> pq){

System.out.println(pq.peek());

}

public static void del(PriorityQueue<Integer> pq){

System.out.println(pq.poll() + " removed");

}

public static void display(PriorityQueue<Integer> pq){

System.out.println("Priority Queue: " + pq);

}

public static void main(String[] args){

Scanner Snr =new Scanner(System.in);

PriorityQueue<Integer> pq = new PriorityQueue<Integer>();

int choice = 0;

while(choice != 5){

System.out.println(" ");

System.out.println(":: PRIORITY QUEUE OPERATIONS ::");

System.out.println("1. Insert");

System.out.println("2. Print top element");

System.out.println("3. Print and Delete top element");

System.out.println("4. Display");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

insert(pq);

break;

case 2:

last(pq);

break;

case 3:

del(pq);

break;

case 4:

display(pq);

break;

case 5:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice!");

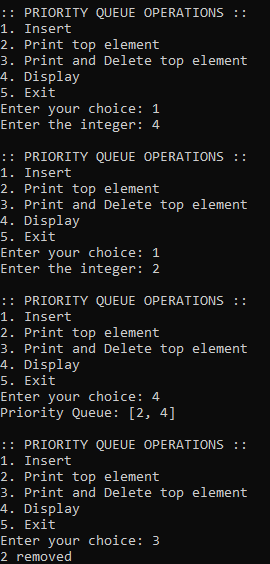
}

}

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 47**

**Aim:**

Program to demonstrate the addition and deletion of elements in deque

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.ArrayDeque;

import java.util.Deque;

public class deque{

public static void insertBeg(Deque<Integer> dq){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int num = Snr.nextInt();

dq.addFirst(num);

}

public static void insertEnd(Deque<Integer> dq){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int num = Snr.nextInt();

dq.addLast(num);

}

public static void delBeg(Deque<Integer> dq){

System.out.println(dq.removeFirst() + " removed");

}

public static void delEnd(Deque<Integer> dq){

System.out.println(dq.removeLast() + " removed");

}

public static void main(String[] args){

Scanner Snr =new Scanner(System.in);

Deque<Integer> dq = new ArrayDeque<Integer>();

int choice = 0;

while(choice != 6){

System.out.println(" ");

System.out.println(":: DEQUE OPERATIONS ::");

System.out.println("1. Insert at Beginning");

System.out.println("2. Insert at End");

System.out.println("3. Delete from Beginning");

System.out.println("4. Delete from End");

System.out.println("5. Display");

System.out.println("6. Exit");

System.out.print("Enter your choice: ");

choice = Snr.nextInt();

switch(choice){

case 1:

insertBeg(dq);

break;

case 2:

insertEnd(dq);

break;

case 3:

delBeg(dq);

break;

case 4:

delEnd(dq);

break;

case 5:

System.out.println(dq);

break;

case 6:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice!");

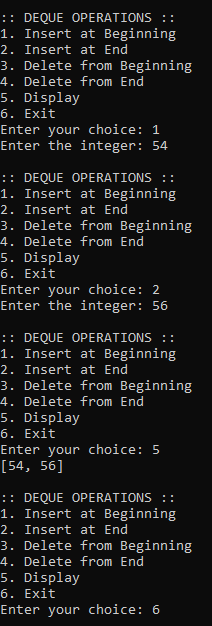
}

}

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 48**

**Aim:**

Program to demonstrate the creation of Set object using the LinkedHashset class

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.LinkedHashSet;

import java.util.Set;

public class set{

public static void insert(Set<Integer> st){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int line = Snr.nextInt();

st.add(line);

}

public static void del(Set<Integer> st){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the position: ");

int pos = Snr.nextInt();

pos = pos-1;

System.out.println(st.remove(pos) + " removed");;

}

public static void display(Set<Integer> st){

System.out.println("Stack: " + st);

}

public static void delAll(Set<Integer> st){

st.clear();

System.out.println("Set successfully cleared");

}

public static void main(String[] args){

Set<Integer> st = new LinkedHashSet<Integer>();

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

Scanner Snr = new Scanner(System.in);

int choice1 = 0, choice2 = 0, choice3 = 0;

while(choice1 != 5){

System.out.println(":: SET OPERATIONS ::");

System.out.println("1. Insert");

System.out.println("2. Delete");

System.out.println("3. Display");

System.out.println("4. Clear All");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice1 = Snr.nextInt();

switch(choice1){

case 1:

insert(st);

break;

case 2:

del(st);

break;

case 3:

display(st);

break;

case 4:

delAll(st);

break;

case 5:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice");

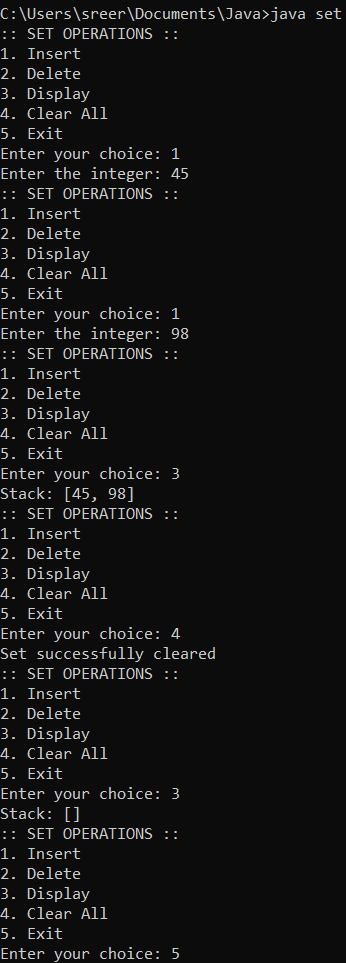
}

}

}

}

**Output:**

****

**Result:**

Output displayed successfully and CO4 was obtained

**Experiment 49**

**Aim:**

Write a Java program to compare two hash set

**Course Outcome(CO4):**

Implement packages, exception handling, multithreading and generic programming. Use java.util package and Collection framework

**Procedure:**

import java.util.\*;

import java.util.LinkedHashSet;

import java.util.Set;

public class set{

public static void insert1(Set<Integer> set1){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int line = Snr.nextInt();

set1.add(line);

}

public static void del1(Set<Integer> set1){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the position: ");

int pos = Snr.nextInt();

pos = pos-1;

System.out.println(set1.remove(pos) + " removed");;

}

public static void display1(Set<Integer> set1){

System.out.println("Stack: " + set1);

}

public static void delAll1(Set<Integer> set1){

set1.clear();

System.out.println("Set successfully cleared");

}

public static void insert2(Set<Integer> set2){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the integer: ");

int line = Snr.nextInt();

set2.add(line);

}

public static void del2(Set<Integer> set2){

Scanner Snr = new Scanner(System.in);

System.out.print("Enter the position: ");

int pos = Snr.nextInt();

pos = pos-1;

System.out.println(set2.remove(pos) + " removed");;

}

public static void display2(Set<Integer> set2){

System.out.println("Stack: " + set2);

}

public static void delAll2(Set<Integer> set2){

set2.clear();

System.out.println("Set successfully cleared");

}

public static void union(Set<Integer> set1, Set<Integer> set2){

Set<Integer> un = new LinkedHashSet<Integer>(set1);

un.addAll(set2);

System.out.println(un);

}

public static void intersection(Set<Integer> set1, Set<Integer> set2){

Set<Integer> inter = new LinkedHashSet<Integer>(set1);

inter.retainAll(set2);

System.out.println(inter);

}

public static void difference(Set<Integer> set1, Set<Integer> set2){

Set<Integer> diff = new LinkedHashSet<Integer>(set1);

diff.removeAll(set2);

System.out.println(diff);

}

public static void main(String[] args){

Set<Integer> set1 = new LinkedHashSet<Integer>();

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

Scanner Snr = new Scanner(System.in);

int choice1 = 0, choice2 = 0, choice3 = 0;

while(choice1 != 5){

System.out.println(":: FIRST SET ::");

System.out.println("1. Insert");

System.out.println("2. Delete");

System.out.println("3. Display");

System.out.println("4. Clear All");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice1 = Snr.nextInt();

switch(choice1){

case 1:

insert1(set1);

break;

case 2:

del1(set1);

break;

case 3:

display1(set1);

break;

case 4:

delAll1(set1);

break;

case 5:

break;

default:

System.out.println("Enter a valid choice");

}

}

while(choice2 != 5){

System.out.println(":: SECOND SET ::");

System.out.println("1. Insert");

System.out.println("2. Delete");

System.out.println("3. Display");

System.out.println("4. Clear All");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice2 = Snr.nextInt();

switch(choice2){

case 1:

insert2(set2);

break;

case 2:

del2(set2);

break;

case 3:

display2(set2);

break;

case 4:

delAll2(set2);

break;

case 5:

break;

default:

System.out.println("Enter a valid choice");

}

}

while(choice3 != 4){

System.out.println(":: SET OPERATIONS ::");

System.out.println("1. Union");

System.out.println("2. Intersection");

System.out.println("3. Difference");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

choice3 = Snr.nextInt();

switch(choice3){

case 1:

union(set1, set2);

break;

case 2:

intersection(set1, set2);

break;

case 3:

difference(set1, set2);

break;

case 4:

System.exit(0);

break;

default:

System.out.println("Enter a valid choice");

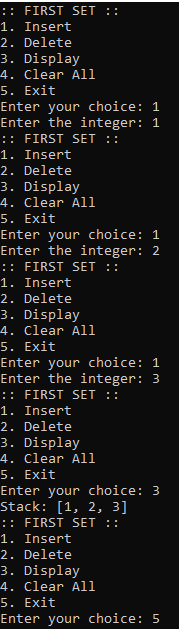
}

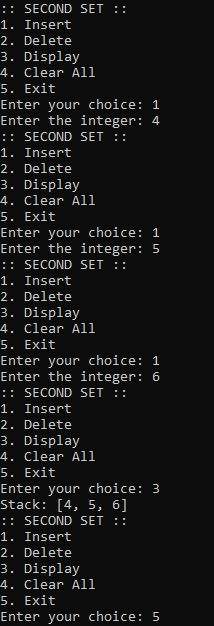
}

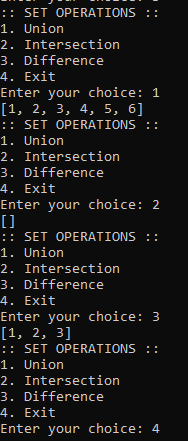
}

}

**Output:**

****

****

****

**Result:**

Output displayed successfully and CO4 was obtained