**CYCLE- 4**

**1. 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.**

**CODE:**

**main\_graphics.java**

import package\_graphics.\*;

import java.util.\*;

public class main\_graphics

{

public static void main(String []args)

{

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:16/06/2023\n\n");

package\_graphics testObj = new package\_graphics();

int l,h,r,a,c,d;

Scanner s=new Scanner(System.in);

System.out.println("Enter the length for rectangle");

l=s.nextInt();

System.out.println("Enter the breadth for rectangle");

h=s.nextInt();

System.out.println("Enter the radius of circle");

r=s.nextInt();

System.out.println("Enter the side for Square");

a=s.nextInt();

System.out.println("Enter the breadth for triangle");

c=s.nextInt();

System.out.println("Enter the height for triangle");

d=s.nextInt();

System.out.println(" Area of rectangle= "+testObj.recArea(l,h));

System.out.println(" Area of circle= "+testObj.cirArea(r));

System.out.println(" Area of square= "+testObj.squArea(a));

System.out.println(" Area of triangle= "+testObj.triArea(c,d));

}

}

**package\_graphics.java**

package package\_graphics;

interface interface\_graphics{

public float recArea(int l, int h);

public float cirArea(int r);

public float squArea(int a);

public float triArea(int l, int h);

}

public class package\_graphics implements interface\_graphics {

public float recArea(int l, int h){

return l\*h;

}

public float cirArea(int r){

return r\*r\*(float)3.14;

}

public float squArea(int a){

return a\*a;

}

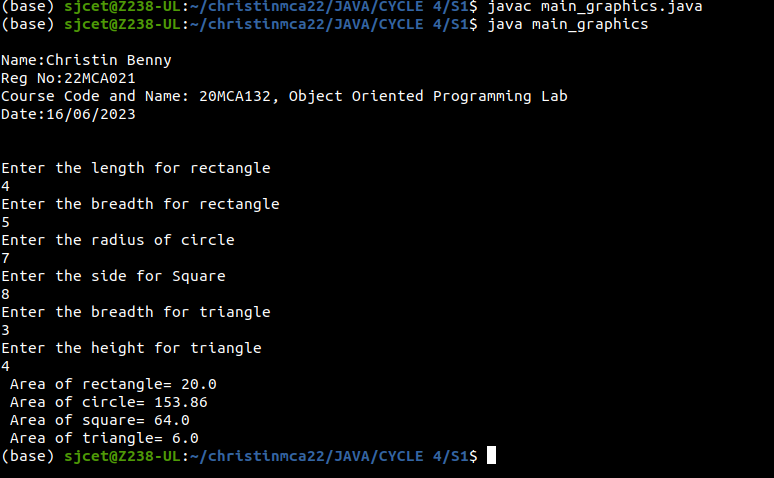
public float triArea(int l, int h){

return l\*h\*(float)(.5);

}

}

**OUTPUT:**

****

**2. 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**

**CODE:**

**ArithmeticMain.java**

import arithmetic.ArithmeticOperations;

import java.util.Scanner;

public class ArithmeticMain {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

ArithmeticOperations operations = new ArithmeticOperations();

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the first number: ");

double num1 = scanner.nextDouble();

System.out.print("Enter the second number: ");

double num2 = scanner.nextDouble();

System.out.println("Addition: " + operations.add(num1, num2));

System.out.println("Subtraction: " + operations.subtract(num1, num2));

System.out.println("Multiplication: " + operations.multiply(num1, num2));

System.out.println("Division: " + operations.divide(num1, num2));

}

}

**ArithmeticOperations.java**

package arithmetic;

public class ArithmeticOperations implements Addition, Subtraction, Multiplication, Division {

@Override

public double add(double num1, double num2) {

return num1 + num2;

}

@Override

public double subtract(double num1, double num2) {

return num1 - num2;

}

@Override

public double multiply(double num1, double num2) {

return num1 \* num2;

}

@Override

public double divide(double num1, double num2) {

if (num2 == 0) {

throw new ArithmeticException("Division by zero error!");

}

return num1 / num2;

}

}

**Addition.java**

package arithmetic;

public interface Addition {

public double add(double num1, double num2);

}

**Division.java**

package arithmetic;

public interface Division {

public double divide(double num1, double num2);

}

**Multiplication.java**

package arithmetic;

public interface Multiplication {

public double multiply(double num1, double num2);

}

**Subtraction.java**

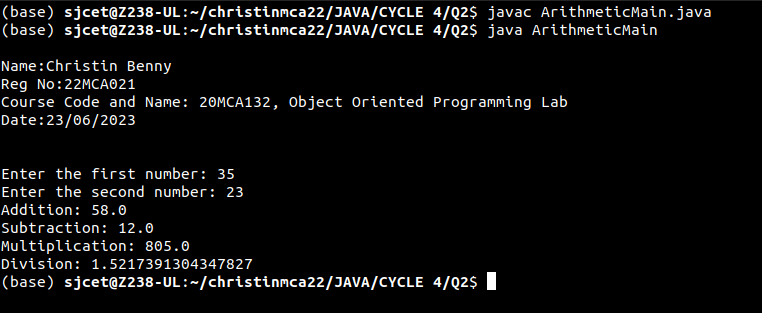
package arithmetic;

public interface Subtraction {

public double subtract(double num1, double num2);

}

**OUTPUT:**

****

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

**CODE:**

import java.util.Scanner;

class authException extends Exception{

public authException(String s){

super(s);

}

}

public class Userauthentication{

public static void main(String[] args){

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

String username = "mca";

String passcode = "mca123";

String user\_name,password;

Scanner sc = new Scanner(System.in);

try

{

System.out.println("Enter username: ");

user\_name = sc.nextLine();

System.out.println("Enter password: ");

password = sc.nextLine();

if(username.equals(user\_name) && passcode.equals(password)){

System.out.println("Authentication successful...");

}

else

throw new authException("Invalid user credentials");

}

catch(authException e)

{

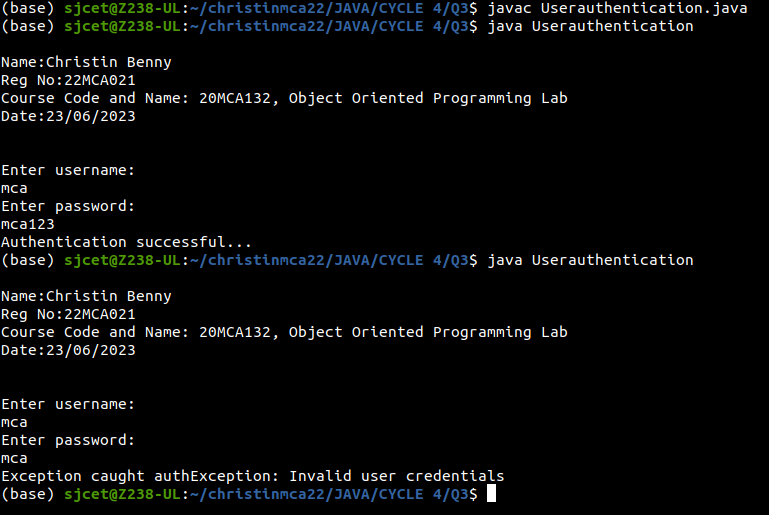
System.out.println("Exception caught "+e);

}

}

}

**OUTPUT:**

****

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

**CODE:**

import java.util.Scanner;

class NegException extends Exception{

public NegException(String s){

super(s);

}

}

public class average {

public static void main(String[] args){

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

int i;

double sum=0,avg=0;

Scanner sc=new Scanner(System.in);

System.out.println("Enter n numbers:");

int n=sc.nextInt();

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

try{

System.out.println("Enter number"+i);

int a=sc.nextInt();

if(a<0){

i--;

throw new NegException("Negative numbers not allowed,Try again");

}

else{

sum=sum+a;

}

}

catch(NegException e){

System.out.println("NEGETIVE EXCEPTION OCCURED:"+e);

}

}

avg=sum/n;

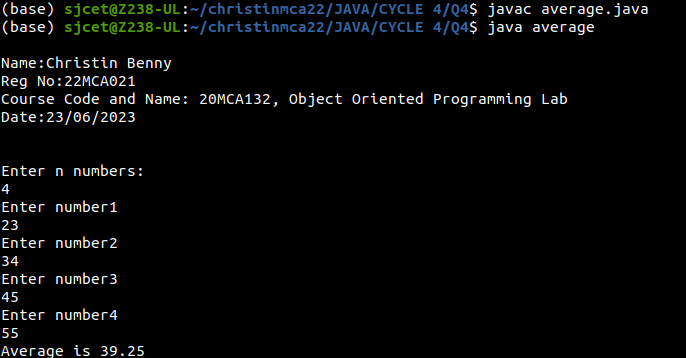
System.out.println("Average is "+avg);

sc.close();

}

}

**OUTPUT:**

****

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

**CODE:**

class MultiplicationTable implements Runnable {

@Override

public void run() {

System.out.println("Multiplication Table of 5:");

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

System.out.println("5 \* " + i + " = " + (5 \* i));

}

}

}

class PrimeNumbers implements Runnable {

@Override

public void run() {

System.out.println("First 10 Prime Numbers:");

int count = 0;

int num = 2;

while (count < 10) {

if (isPrime(num)) {

System.out.println(num + " ");

count++;

}

num++;

}

System.out.println();

}

private boolean isPrime(int num) {

if (num < 2) {

return false;

}

for (int i = 2; i <= Math.sqrt(num); i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

}

public class MainThread {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

MultiplicationTable multiplicationTable = new MultiplicationTable();

PrimeNumbers primeNumbers = new PrimeNumbers();

Thread thread1 = new Thread(multiplicationTable);

Thread thread2 = new Thread(primeNumbers);

thread1.start();

try {

thread1.join();

} catch (InterruptedException e) {

e.printStackTrace();

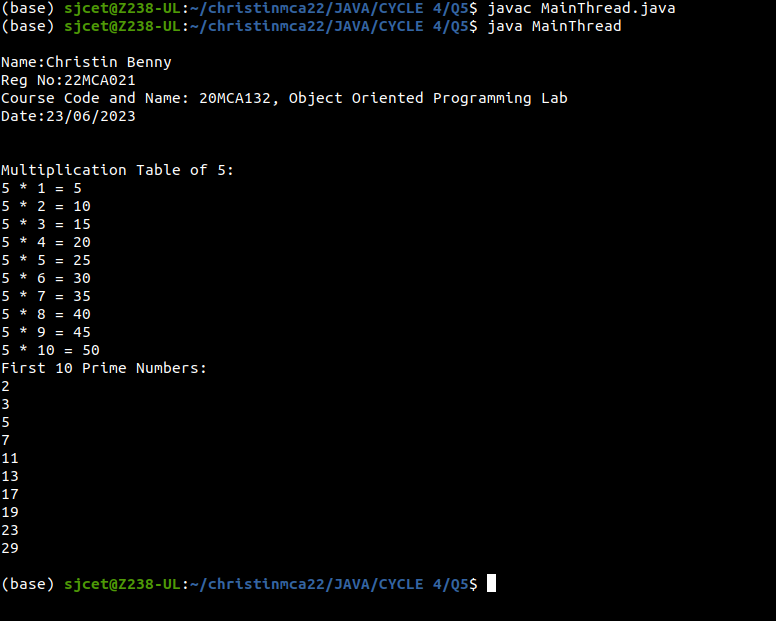
}

thread2.start();

}

}

**OUTPUT:**

****

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

**CODE:**

import java.util.Scanner;

class Fibonacci implements Runnable {

private int count;

public Fibonacci(int count) {

this.count = count;

}

@Override

public void run() {

System.out.println("Fibonacci numbers:");

int num1 = 0;

int num2 = 1;

System.out.println(num1);

System.out.println(num2);

for (int i = 2; i < count; i++) {

int fib = num1 + num2;

System.out.println(fib);

num1 = num2;

num2 = fib;

}

}

}

class EvenNumber implements Runnable {

private int start;

private int end;

public EvenNumber(int start, int end) {

this.start = start;

this.end = end;

}

@Override

public void run() {

System.out.println("Even numbers from " + start + " to " + end + ":");

for (int i = start; i <= end; i++) {

if (i % 2 == 0) {

System.out.println(i);

}

}

}

}

public class multiThread {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the count of Fibonacci numbers: ");

int fibonacciCount = scanner.nextInt();

System.out.print("Enter the start of the range for even numbers: ");

int start = scanner.nextInt();

System.out.print("Enter the end of the range for even numbers: ");

int end = scanner.nextInt();

scanner.close();

Thread fibonacciThread = new Thread(new Fibonacci(fibonacciCount));

Thread evenNumberThread = new Thread(new EvenNumber(start, end));

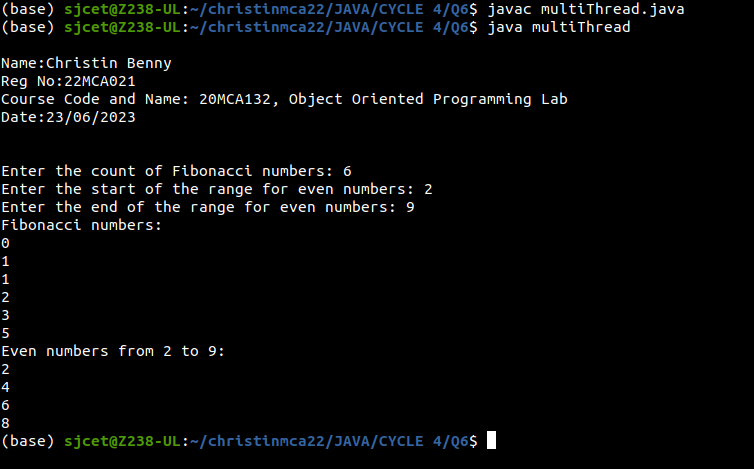
fibonacciThread.start();

evenNumberThread.start();

}

}

**OUTPUT:**

****

**7. Producer/Consumer using ITC.**

**CODE:**

import java.util.LinkedList;

class Buffer {

private LinkedList<Integer> buffer;

private int capacity;

public Buffer(int capacity) {

this.buffer = new LinkedList<>();

this.capacity = capacity;

}

public void produce(int value) throws InterruptedException {

synchronized (this) {

while (buffer.size() == capacity) {

wait();

}

buffer.add(value);

System.out.println("Produced: " + value);

notifyAll();

}

}

public void consume() throws InterruptedException {

synchronized (this) {

while (buffer.isEmpty()) {

wait();

}

int value = buffer.removeFirst();

System.out.println("Consumed: " + value);

notifyAll();

}

}

}

class Producer implements Runnable {

private Buffer buffer;

private int numProductions;

public Producer(Buffer buffer, int numProductions) {

this.buffer = buffer;

this.numProductions = numProductions;

}

@Override

public void run() {

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

try {

buffer.produce(i);

Thread.sleep(1000); // Simulate production time

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

class Consumer implements Runnable {

private Buffer buffer;

private int numConsumptions;

public Consumer(Buffer buffer, int numConsumptions) {

this.buffer = buffer;

this.numConsumptions = numConsumptions;

}

@Override

public void run() {

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

try {

buffer.consume();

Thread.sleep(2000); // Simulate consumption time

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

public class ProducerConsumerITC {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

Buffer buffer = new Buffer(5);

int numProductions = 10;

int numConsumptions = 10;

Producer producer = new Producer(buffer, numProductions);

Consumer consumer = new Consumer(buffer, numConsumptions);

Thread producerThread = new Thread(producer);

Thread consumerThread = new Thread(consumer);

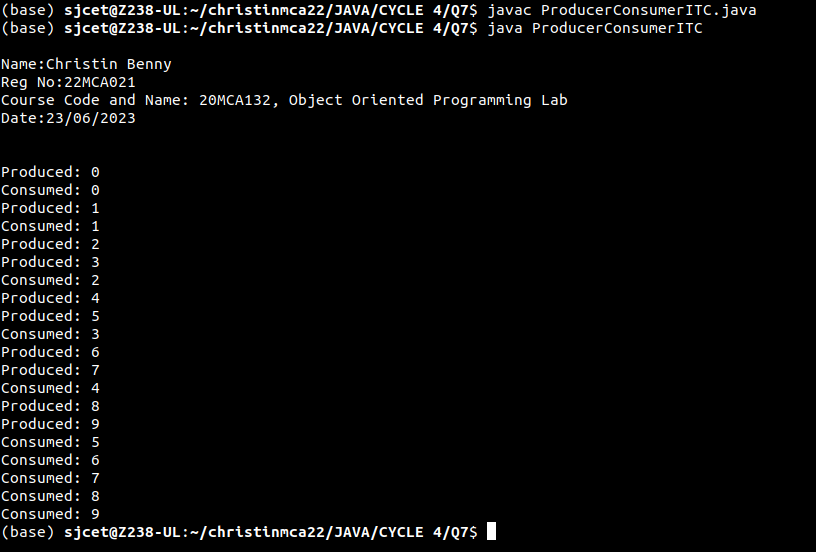
producerThread.start();

consumerThread.start();

}

}

**OUTPUT:**

****

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

**CODE:**

class Stack {

private int arr[];

private int top;

private int capacity;

Stack(int size) {

arr = new int[size];

capacity = size;

top = -1;

}

public void push(int x) {

if (isFull()) {

System.out.println("Stack OverFlow");

System.exit(1);

}

System.out.println("Inserting " + x);

arr[++top] = x;

}

public int pop() {

if (isEmpty()) {

System.out.println("STACK EMPTY");

System.exit(1);

}

return arr[top--];

}

public int getSize() {

return top + 1;

}

public Boolean isEmpty() {

return top == -1;

}

public Boolean isFull() {

return top == capacity - 1;

}

public void printStack() {

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

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

}

}

public static void main(String[] args) {

System.out.println("\nName : Christin Benny\nReg No: 22MCA021\nCourse Code and Name : 20MCA132,Object Oriented Programming Lab\nDate : 27/06/2023\n\n");

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

Stack stack = new Stack(5);

stack.push(1);

stack.push(2);

stack.push(3);

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

stack.printStack();

stack.pop();

System.out.println("\nAfter popping out\n");

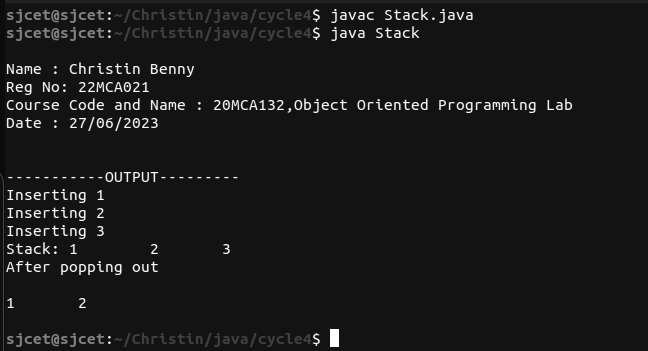
stack.printStack();

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

}

}

**OUTPUT:**

****

**9. Using generic method perform Bubble sort.**

**CODE:**

import java.util.Arrays;

public class BubbleSort {

public static <T extends Comparable<T>> void bubbleSort(T[] arr) {

int n = arr.length;

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

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

if (arr[j].compareTo(arr[j + 1]) > 0) {

// Swap arr[j] and arr[j + 1]

T temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

Integer[] numbers = { 4, 2, 9, 6, 23, 12, 34, 0, 1 };

String[] names = { "John", "Alice", "Bob", "Diana", "Carol" };

System.out.println("Before sorting: " + Arrays.toString(numbers));

bubbleSort(numbers);

System.out.println("After sorting: " + Arrays.toString(numbers));

System.out.println("Before sorting: " + Arrays.toString(names));

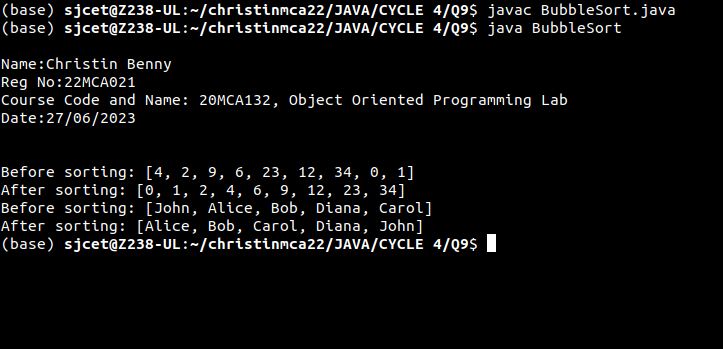
bubbleSort(names);

System.out.println("After sorting: " + Arrays.toString(names));

}

}

**OUTPUT:**

****

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

**CODE:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class ArrayListExample {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

List<String> stringList = new ArrayList<>();

stringList.add("Apple");

stringList.add("Banana");

stringList.add("Orange");

stringList.add("Mango");

System.out.println("Original List: " + stringList);

String firstElement = stringList.get(0);

System.out.println("First Element: " + firstElement);

stringList.set(1, "Grapes");

System.out.println("Modified List: " + stringList);

boolean containsMango = stringList.contains("Mango");

System.out.println("Contains Mango? " + containsMango);

stringList.remove("Orange");

System.out.println("List after removing Orange: " + stringList);

Collections.sort(stringList);

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

Collections.reverse(stringList);

System.out.println("Reversed List: " + stringList);

int size = stringList.size();

System.out.println("Size of the list: " + size);

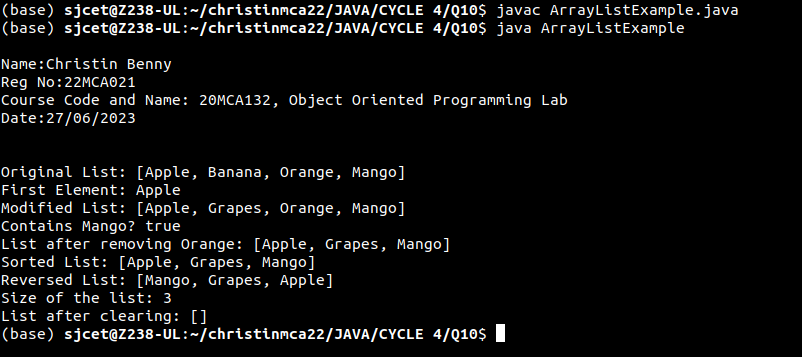
stringList.clear();

System.out.println("List after clearing: " + stringList);

}

}

**OUTPUT:**

****

**11. Program to remove all the elements from a linked list.**

**CODE:**

import java.util.Scanner;

public class LinkedListDemo {

private Node head;

private class Node {

String data;

Node next;

Node(String data) {

this.data = data;

this.next = null;

}

}

public void add(String data) {

Node newNode = new Node(data);

if (head == null) {

head = newNode;

} else {

Node currentNode = head;

while (currentNode.next != null) {

currentNode = currentNode.next;

}

currentNode.next = newNode;

}

}

public void removeAll() {

head = null;

}

public void display() {

Node currentNode = head;

while (currentNode != null) {

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

currentNode = currentNode.next;

}

System.out.println();

}

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:23/06/2023\n\n");

LinkedListDemo linkedList = new LinkedListDemo();

Scanner scanner = new Scanner(System.in);

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

int numElements = scanner.nextInt();

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

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

String element = scanner.next();

linkedList.add(element);

}

System.out.println("Linked List before removal:");

linkedList.display();

linkedList.removeAll();

System.out.println("Linked List after removal:");

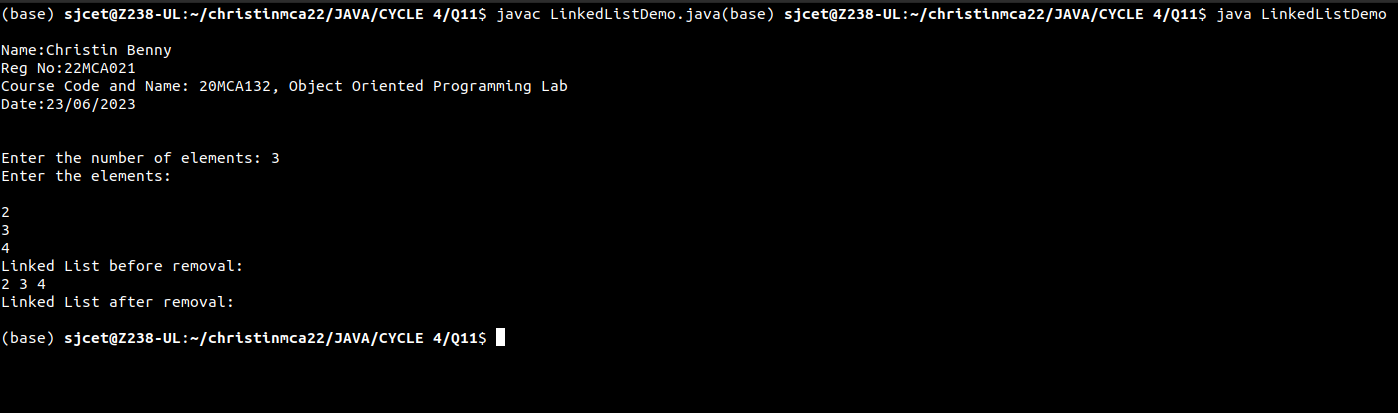
linkedList.display();

scanner.close();

}

}

**OUTPUT:**

****

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

**CODE:**

import java.util.Stack;

public class StackRemoveElementExample {

public static void main(String[] args) {

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

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

stack.push("Apple");

stack.push("Banana");

stack.push("Orange");

stack.push("Mango");

System.out.println("Stack elements: " + stack);

int positionToRemove = 2;

removeElement(stack, positionToRemove);

System.out.println("Stack after removal: " + stack);

}

public static void removeElement(Stack<String> stack, int position) {

if (stack.isEmpty() || position <= 0 || position > stack.size()) {

System.out.println("Invalid position or stack is empty.");

return;

}

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

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

tempStack.push(stack.pop());

}

stack.pop();

while (!tempStack.isEmpty()) {

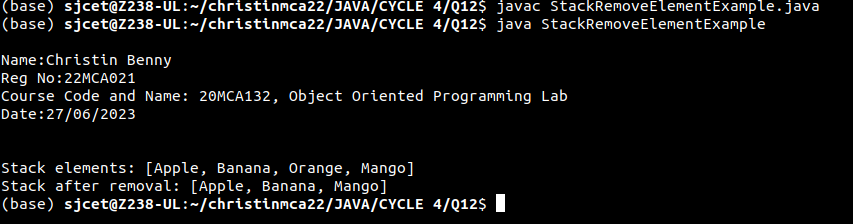
stack.push(tempStack.pop());

}

}

}

**OUTPUT:**

****

**13. Program to demonstrate the creation of queue object using the PriorityQueue class.**

**CODE:**

import java.util.PriorityQueue;

import java.util.Queue;

public class PriorityQueueExample {

public static void main(String[] args) {

Queue<Integer> queue = new PriorityQueue<>();

queue.offer(5);

queue.offer(2);

queue.offer(8);

queue.offer(1);

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

System.out.println("Queue elements: " + queue);

while (!queue.isEmpty()) {

int element = queue.poll();

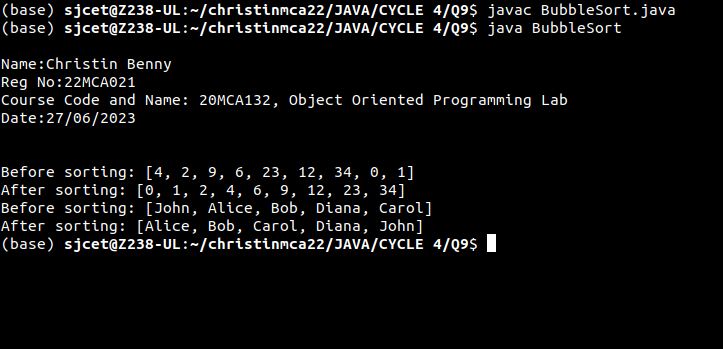
System.out.println("Removed element: " + element);

}

}

}

**OUTPUT:**

****

**14. Program to demonstrate the addition and deletion of elements in deque.**

**CODE:**

import java.util.\*;

class deque

{

public static void main(String[] args)

{

Deque<String> deque = new LinkedList<String>();

deque.add("Java");

deque.addFirst("Python");

deque.addLast("Datastructure");

deque.push("Web-programming");

deque.offer("Networking");

deque.offerFirst("DBMS");

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

System.out.println(deque + "\n");

deque.removeFirst();

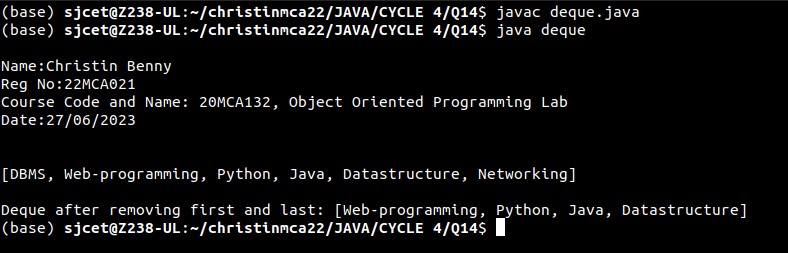
deque.removeLast();

System.out.println("Deque after removing " + "first and last: " + deque);

}

}

**OUTPUT:**

****

**15. Program to demonstrate the creation of Set object using the LinkedHashset class.**

**CODE:**

import java.util.LinkedHashSet;

import java.util.Set;

public class LinkedHashSetExample {

public static void main(String[] args) {

// Create a LinkedHashSet

Set<String> set = new LinkedHashSet<>();

// Add elements to the set

set.add("Apple");

set.add("Banana");

set.add("Orange");

set.add("Apple"); // Adding a duplicate element

// Print the set

System.out.println("\nName:Christin Benny\nReg No:22MCA021\nCourse Code and Name: 20MCA132, Object Oriented Programming Lab\nDate:27/06/2023\n\n");

System.out.println("Set elements: " + set);

// Check if an element exists in the set

boolean containsBanana = set.contains("Banana");

System.out.println("Contains 'Banana'? " + containsBanana);

// Remove an element from the set

boolean removedOrange = set.remove("Orange");

System.out.println("Removed 'Orange'? " + removedOrange);

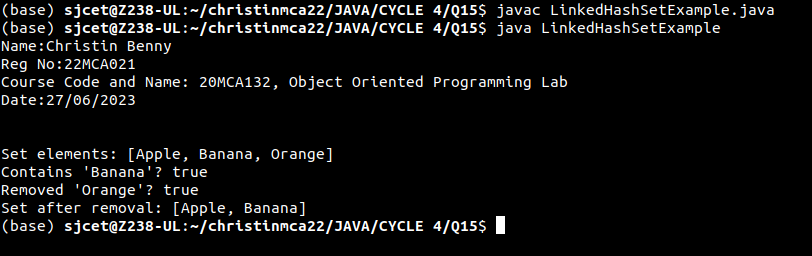
// Print the set after removal

System.out.println("Set after removal: " + set);

}

}

**OUTPUT:**

****

**16. Write a Java program to compare two hash set.**

**CODE:**

import java.util.\*;

public class CompareHash {

public static void main(String[] args) {

System.out.println("\nName : Christin Benny\nReg No: 22MCA021\nCourse Code and Name : 20MCA132,Object Oriented Programming Lab\nDate : 27/06/2023\n\n");

HashSet<String> h\_set = new HashSet<String>();

h\_set.add("Red");

h\_set.add("Green");

h\_set.add("Black");

h\_set.add("White");

HashSet<String>h\_set2 = new HashSet<String>();

h\_set2.add("Red");

h\_set2.add("Pink");

h\_set2.add("Black");

h\_set2.add("Orange");

HashSet<String>result\_set = new HashSet<String>();

for (String element : h\_set){

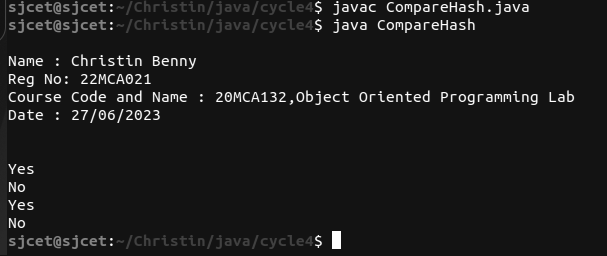
System.out.println(h\_set2.contains(element) ? "Yes" : "No");

}

}

}

**OUTPUT:**

****

**17. Program to demonstrate the working of Map interface by adding, changing and removing elements.**

**CODE:**

import java.util.HashMap;

import java.util.Map;

public class MapExample {

public static void main(String[] args) {

// Create a new HashMap

Map<String, Integer> map = new HashMap<>();

// Add elements to the map

map.put("John", 25);

map.put("Alice", 30);

map.put("Bob", 35);

// Print the initial map

System.out.println("\nName : Christin Benny\nReg No: 22MCA021\nCourse Code and Name : 20MCA132,Object Oriented Programming Lab\nDate : 27/06/2023\n\n");

System.out.println("Initial Map: " + map);

// Changing an element

map.put("Alice", 32);

// Print the map after changing an element

System.out.println("Map after changing an element: " + map);

// Removing an element

map.remove("Bob");

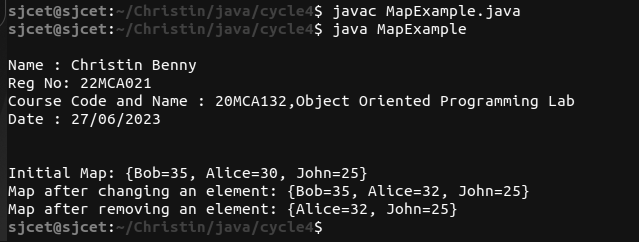
// Print the map after removing an element

System.out.println("Map after removing an element: " + map);

}

}

**OUTPUT:**

****

**18. Program to Convert HashMap to TreeMap.**

**CODE:**

import java.util.HashMap;

import java.util.Map;

import java.util.TreeMap;

public class HashMapToTreeMapExample {

public static void main(String[] args) {

// Create a HashMap

Map<String, Integer> hashMap = new HashMap<>();

hashMap.put("John", 25);

hashMap.put("Alice", 30);

hashMap.put("Bob", 35);

// Convert HashMap to TreeMap

Map<String, Integer> treeMap = new TreeMap<>(hashMap);

// Print the HashMap

System.out.println("\nName : Christin Benny\nReg No: 22MCA021\nCourse Code and Name : 20MCA132,Object Oriented Programming Lab\nDate : 27/06/2023\n\n");

System.out.println("HashMap: " + hashMap);

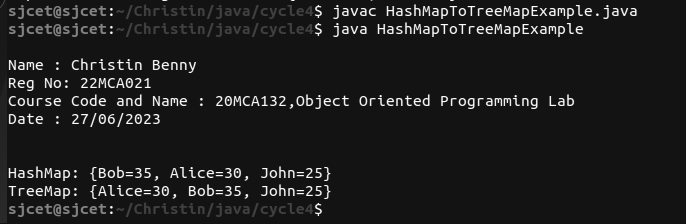
// Print the TreeMap

System.out.println("TreeMap: " + treeMap);

}

}

**OUTPUT:**

****