Lab 7 Collection , I/O

Q.1

Write a Java program that accepts two filenames. Based on the user's choice to copy or append, copy the first file into the second file or append the content of the first file to the second file.

Ans.

```
package lab7.que1;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Scanner;

public class InputOutput {
    public static void main(String[] args) {
        Scanner scanner=new Scanner(System.in);
        System.out.println("enter first file name");
        String st1=scanner.nextLine();
        System.out.println("enter second file name");
        String st2=scanner.nextLine();
        System.out.println("enter 0 for copying "+ st1 +" t
o" + st2 +"1 appending ");
        boolean flag=scanner.nextBoolean();
        StringBuffer st=new StringBuffer("/Users/lakhman/De
sktop/sem 4 coding/java/submission/src/lab7/que1/");
        String filePath = String.valueOf(st.append(st1));

        String  filepath2=String.valueOf(st.append(st2));
        try {
            FileInputStream in = new FileInputStream(filePa
th);
```

```
            FileOutputStream out=null;
            if(flag) out = new FileOutputStream(filepath2);
            else out = new FileOutputStream(filepath2,tru
e);

            int temp;


                while ((temp = in.read()) != -1) {


                    out.write(temp);

            }
        } catch (IOException e) {

            e.printStackTrace();
        }
    }
}
```

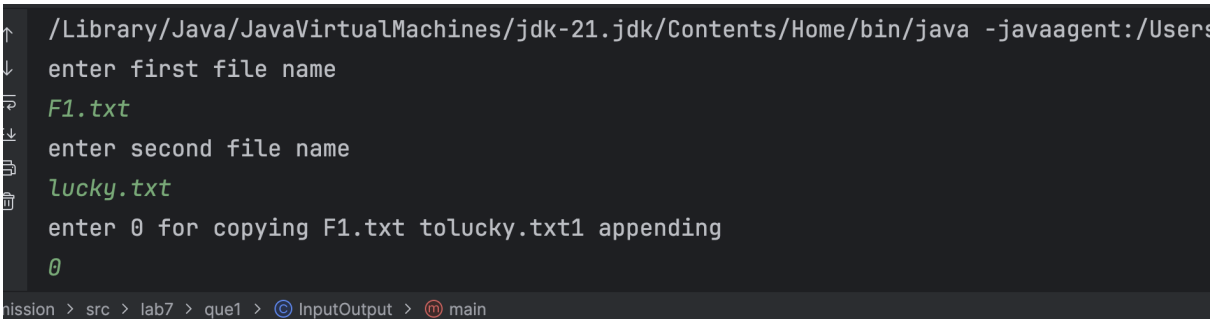
Output:

enter first file name
F1.txt
enter second file name
lucky.txt
enter 0 for copying F1.txt tolucky.txt1 appending
0

Process finished with exit code 0

Q.2

Write a Java program to generate a linked list of some five students (Student objects) and display the list of students in a sorted order as per name of students.



Ans.

```
package lab7.que1;
import java.util.*;
class Student {
    int rollno;
    String name;



}
class StudentComparator implements Comparator<Student> {
    @Override
    public int compare(Student student1, Student student2)
{
        // Compare students based on their names
        return student1.name.compareTo(student2.name);
    }
}
public class Linklistquestion {
    public static void main(String[] args) {
        Student st=new Student();
        LinkedList<Student> ll = new LinkedList<Student>();
        st.name="lakhman";
        st.rollno=30;
        Student st2=new Student();
        st2.rollno=33;
        st2.name="vaibha";
        Student st3=new Student();
        st3.rollno=3;
```

```
        st3.name="shivansh";
        Student st4=new Student();
        st4.rollno=13;
        st4.name="pradip";
        Student st5=new Student();
        st5.rollno=43;
        st5.name="raza";
        ll.add(st);
        ll.add(st2);
        ll.add(st3);
        ll.add(st4);
        ll.add(st5);
//        ll.sort(Comparator.comparingInt(student -> studen
t.rollno));
//        ll.sort(Comparator.comparing(studnet -> studnet.n
ame));
        ll.sort(new StudentComparator());
        for(Student studnet:ll) {
            System.out.println(studnet.rollno+"  "+studnet.
name);
        }
    }
}
```

Output:

```
lakhman  30
pradip  13
raza  43
shivansh  3
vaibha  33

Process finished with exit code 0
```

Q.3

1. Write a Java program to map Person objects to string hobby using TreeMap class. This mapping should store Person objects in ascending sorting by their name.

Persons	hobby
Name: Bhairavi Age: 22	Singing
Name: Dhara Age:23	Sketching
Name: Anmol Age: 23	Reading
Name: Megh Age 21	Singing
Name: Raag Age:22	Sketching

Find the unique list/set of all the hobbies that are mapped in this collection and display it.

Ans.

```
package lab7.que1;

import java.util.Arrays;
import java.util.HashMap;
import java.util.*;

class Person implements Comparable<Person>{
    int age;
    String name;

    public Person(int age, String name) {
        this.age = age;
        this.name = name;
    }
@Override public  int compareTo(Person p1) {
        return this.name.compareTo(p1.name);
    }
}
public class Treemaps {
    public static void main(String[] args) {
```

```
        String[] hobby={"Singing", "Sketching", "Readin
g","Singing","Sketching"};
        Person person1=new Person(22,"bhargav");
        Person person2=new Person(15,"nihar");
        Person person3=new Person(28,"duryodhan");
        Person person4=new Person(10,"karn");
        Person person5=new Person(50,"parth");
        Map<Person,String> newmap= new TreeMap<Person,Strin
g>();

        newmap.put(person1,hobby[0]);
        newmap.put(person2,hobby[1]);
        newmap.put(person3,hobby[2]);
        newmap.put(person4,hobby[3]);
        newmap.put(person5,hobby[4]);
        Set<String> uniuqe=new HashSet<>();
        uniuqe.addAll(List.of(hobby));

        System.out.println();
        for (Person p:newmap.keySet()){
            System.out.println(p.name +"  "+ p.age);
        }

        for(String lists:uniuqe) {
            System.out.println(lists);
        }









    }
}
```

Output:

Reading
Singing
Sketching
bhargav duryodhan karn
10
nihar
15
parth
50
22
28
Process finished with exit code 0

Q.4

Write a generic interface MinMax having two methods findMin() and findMax(). Create a class named MyClass which implements the above interface. Write an appropriate demo class to test your classes and interfaces. Create an object of MyClass which stores an array of Book and find books having minimum and maximum price.

Ans.

```
package lab7.que1;

class MyClass<T extends Comparable<T>> implements MinMax<T>
{

    @Override
    public T findMin(T[] arr) {
        if (arr == null || arr.length == 0) {
            return null;
        }
```

```
        T min = arr[0];
        for (T item : arr) {
            if (item.compareTo(min) < 0) {
                min = item;
            }
        }
        return min;
    }

    @Override
    public T findMax(T[] arr) {
        if (arr == null || arr.length == 0) {
            return null;
        }

        T max = arr[0];
        for (T item : arr) {
            if (item.compareTo(max) > 0) {
                max = item;
            }
        }
        return max;
    }
}
class Book implements Comparable<Book> {
     String title;
     double price;

    public Book(String book1, double price) {
        this.title=book1;
        this.price=price;
    }

    // Constructor and other methods for Book class

    @Override
    public int compareTo(Book other) {
        // Implement comparison based on book prices
```

```
        return Double.compare(this.price, other.price);
    }

    public String getTitle() {
        return title;
    }
}
public class Demo {
    public static void main(String[] args) {
        Book[] books = {
                new Book("Book1", 20.5),
                new Book("Book2", 15.8),
                new Book("Book3", 25.3),
                // Add more books as needed
        };

        // Create an object of MyClass
        MyClass<Book> myClass = new MyClass<>();

        // Find the book with the minimum price
        Book minPriceBook = myClass.findMin(books);
        System.out.println("Book with Minimum Price: " + mi
nPriceBook.getTitle());

        // Find the book with the maximum price
        Book maxPriceBook = myClass.findMax(books);
        System.out.println("Book with Maximum Price: " + ma
xPriceBook.getTitle());
    }
}
```

Output:

Book with Minimum Price: Book2
Book with Maximum Price: Book3

Process finished with exit code 0