OBJECT ORIENTED PROGRAMMING WITH JAVA 8– LAB 11

Q1. Write a program to create three child threads, one to compute the first 25 even numbers, second to compute the first 50 fibonacci numbers and the third to print the first 20 numbers from the multiplication table of 15 . Implement multithreading using ExecutorService interface in such a way that only 2 threads execute at a time.

```
import java.util.concurrent.*;
class EvenFiboMulti
{
  public static void main(String[] args)
  {
   ExecutorService ec = Executors.newFixedThreadPool(2);
   ec.execute(()->{
                    System.out.println("First 25 even numbers :");
                    for(int i = 2; i <= 50; i += 2) {
                       System.out.println(i + " ");
                      }
                       System.out.println();
                   });
   ec.execute(()->{
                    System.out.println("First 50 Fibonacci numbers are :");
                     int a = 0, b = 1;
                    System.out.println(a + " " + b + " ");
                    for(int i = 3; i <= 50; i++) {
                        int c = a + b;
                    System.out.println(c + " ");
                        a = b;
                        b = c;
                    }
                    System.out.println();
                  });
   ec.execute(()->{
                    System.out.println("First 20 numbers from the multiplication table of
15 :");
                    for(int i = 1; i <= 20; i++) {
                      int n = 15 * i ;
                    System.out.print(n + " ");
                   }
                    System.out.println();
                 });
   ec.shutdown();
  }
}
```

```
First 25 even numbers :
First 50 Fibonacci numbers are :
2
0 1
4
1
6
2
3
8
5
10
8
12
13
14
16
21
18
34
20
55
22
89
24
144
26
233
377
610
987
28
1597
30
32
34
36
38
40
42
44
46
48
50
2584

4181
First 20 numbers from the multiplication table of 15 :
6765
10946
17711
28657
46368
75025
121393
15 196418
30 45 60 75 90 105 120 135 150 317811
514229
165 180 195 210 225 240 255 270 285 300
832040
1346269
2178309
3524578
5702887
9227465
14930352
24157817
39088169
63245986
102334155
165580141
267914296
```

Q2. Write a program to extract a portion of an ArrayList.

```
import java.util.*;
class ArrayListEx
{
 public static void main(String args[])
 {
   ArrayList<String> a = new ArrayList<String>();
      a.add("Dog");
      a.add("Cat");
      a.add("Elephant");
      a.add("Monkey");
      a.add("Lion");
   int startIndex = 1;
   int endIndex = 3;
   List<String> e = a.subList(startIndex,endIndex); // e-extracted list
   System.out.println("Extracted portion of Array :");
   for (String List : e) {
      System.out.println(List);
     }
   }
}
```

```
E:\java_notes>javac ArrayListEx.java

E:\java_notes>java ArrayListEx
Extracted portion of Array :
Cat
Elephant
```

Q3. Write a program to update a specific element in an ArrayList by a given element and remove the last element from the ArrayList and print it using for each loop.

```
import java.util.*;
class UpdateRemoveArray
{
  public static void main(String args[])
  {
    ArrayList<String>a = new ArrayList<String>();
    a.add("Dog");
    a.add("Cat");
    a.add("Tiger");
    a.add("Lion");
    System.out.println("Array elements are: " + a);
    a.set(1,"Elephant");
    System.out.println("Array Elements after update :" + a);
    int lastIndex = a.size()-1;
    String removedElement = a.remove(lastIndex);
    System.out.println("Removed elements :" + removedElement);
    System.out.println("Updated Array List :");
    for(String List : a) {
       System.out.println(List);
      }
    }
 }
```

```
E:\java_notes>javac UpdateRemoveArray.java

E:\java_notes>java UpdateRemoveArray
Array elements are: [Dog, Cat, Tiger, Lion]
Array Elements after update :[Dog, Elephant, Tiger, Lion]
Removed elements :Lion
Updated Array List :
Dog
Elephant
Tiger
```

Q4. Create a class named Student with following members

a. Data Members

i. regno

ii. name

iii. marks

b. Constructor to accept all values

c. Method to print the values of data members

Create another class named AddStudent which contains main method to create 5 Student objects and add it to an ArrayList and display the details of all the students using ListIterator.

```
import java.util.*;
class Student
{
    int regno;
    String name;
    double marks;
 Student(int r,String n,double m)
  {
    regno = r;
    name = n;
    marks = m;
  }
 public void Print()
 {
    System.out.println("Registration number  :" + regno);
    System.out.println("Name :" + name);
    System.out.println("Marks :" + marks);
    System.out.println();
 }
}
class AddStudent
{
  public static void main(String args[])
  {
     ArrayList<Student> s = new ArrayList<Student>();
     s.add(new Student(1,"Anna",85.50));
     s.add(new Student(2,"Mahi",90.65));
     s.add(new Student(3,"Ruhi",79.00));
     s.add(new Student(4,"Nidhi",65.90));
     s.add(new Student(5,"Sash",95));
     System.out.println("Students Details :" );
     ListIterator<Student> ir = s.listIterator();
     while(ir.hasNext())
     {
       Student st = ir.next();
       st.Print();
     }
   }
}
```

```
E:\java_notes>javac AddStudent.java

E:\java_notes>java AddStudent
Students Details :
Registration number  :1
Name :Anna
Marks :85.5

Registration number  :2
Name :Mahi
Marks :90.65

Registration number  :3
Name :Ruhi
Marks :79.0

Registration number  :4
Name :Nidhi
Marks :65.9

Registration number  :5
Name :Sash
Marks :95.0
```

Q5. Create a TreeSet and add some Integer objects to it. Create another TreeSet and copy only those elements from first TreeSet to the second which are greater than or equal to a specified value given as input. Display the second TreeSet values as output using Iterator.

```
import java.util.*;

class CopyArrayElements
{
 public static void main(String args[])
  {
    TreeSet<Integer> ts1 = new TreeSet<Integer>();

         ts1.add(1);
         ts1.add(2);
         ts1.add(3);
         ts1.add(4);
         ts1.add(5);

         System.out.println("First Array Elements :" +ts1);

         Scanner sc = new Scanner(System.in);
         System.out.println("Enter a Number :");
         int n = sc.nextInt();
     TreeSet<Integer> ts2 = new TreeSet<Integer>(ts1);
     ts2.removeAll(ts1.headSet(n));
     System.out.println("Elements in Second Tree greater than or equal to n :");
     Iterator<Integer> ir = ts2.iterator();
     while(ir.hasNext()) {
          System.out.println(ir.next());
         }
       }
  }
```

```
E:\java_notes>javac CopyArrayElements.java

E:\java_notes>java CopyArrayElements
First Array Elements :[1, 2, 3, 4, 5]
Enter a Number :
2
Elements in Second Tree greater than or equal to n :
2
3
4
5
```