**Assignment 6**

**1]ANS:-**

import java.util.EnumSet;

public class Contact {

long phoneNumber;

String name, email;

public enum gender {F,M}

gender g;

public Contact(long phoneNumber, String name, String email, gender g) {

super();

this.phoneNumber = phoneNumber;

this.name = name;

this.email = email;

this.g = g;

}

public long getPhoneNumber() {

return phoneNumber;

}

public void setPhoneNumber(long phoneNumber) {

this.phoneNumber = phoneNumber;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public gen getG() {

return g;

}

public void setG(gender g) {

this.g = g;

}

}

import java.util.Set;

import java.util.Collections;

import java.util.Map;

import java.util.TreeMap;

import org.collection.app.Contact.gen;

public class TreeMapDemo

{

public static void main(String[] args) {

Map<Long,Contact> map = new TreeMap<Long,Contact>();

Contact c1 = new Contact((long)889702956, "Jack", "Jack@gmail.com", gen.M);

Contact c2 = new Contact((long)986689189, "Chinnu", "Chinnu@gmail.com",gen.F);

Contact c3 = new Contact((long)694212857, "Manu", "manu@gmail.com",gen.F);

map.put((long)326452139, c1);

map.put((long)986312475, c2);

map.put((long)694213857, c3);

Map<Long,Contact> sortedMapDesc = new TreeMap<>(

Collections.reverseOrder());

sortedMapDesc.putAll(map);

for(Map.Entry<Long, Contact> entry1: sortedMapDesc.entrySet())

{

Long key = entry1.getKey();

Contact c = entry1.getValue();

System.out.println(key + " -->Phone Number in descending order");

System.out.println(c.name+" "+c.email+" "+c.g + " -->Other Details");

System.out.println(c.phn + " "+ c.name+ " "+ c.email+ " "+ c.g + " -->Full Details");

}

}

}

**Output:-**

986312475 -->Phone Number in descending order

Chinnu Chinnu@gmail.com F -->Other Details

986689189 Chinnu Chinnu@gmail.com F -->Full Details

694213857 -->Phone Number in descending order

Manu manu@gmail.com F -->Other Details

694212857 Manu manu@gmail.com F -->Full Details

326452139 -->Phone Number in descending order

Jack Jack@gmail.com M -->Other Details

889702956 Jack Jack@gmail.com M -->Full Details

**2]ANS:-**

import java.util.\*;

public class Duplicate

{

public static void main(String [] args)

{

int arr[] = {6,3,0,6,7,1,7,3};

ArrayList<Integer> a = new ArrayList<>();

HashSet<Integer> hs = new HashSet<>();

for(int i : arr)

{

if(!hs.contains(i))

{

a.add(i);

hs.add(i);

}

}

for (int i : a)

{

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

}

}

}

**OUTPUT:-**

6 3 0 7 1

**3]ANS:-**

public class Emp1

{

private int id;

private String name;

private String dept;

private double sal;

public Emp1(int id, String name, String dept, double sal)

{

super();

this.id = id;

this.name = name;

this.dept = dept;

this.sal = sal;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDept() {

return dept;

}

public void setDept(String dept) {

this.dept = dept;

}

public double getSal() {

return sal;

}

public void setSal(double sal) {

this.sal = sal;

}

}

import java.util.Comparator;

public class Dcompare implements Comparator<Emp1>{

public int compare(Emp1 o1, Emp1 o2)

{

return o1.getDept().compareTo(o2.getDept());

}

}

import java.util.Comparator;

public class Idcompare implements Comparator<Emp1>{

public int compare(Emp1 o1, Emp1 o2)

{

return o1.getId() - o2.getId();

}

}

import java.util.Comparator;

public class Ncompare implements Comparator<Emp1>

{

public int compare(Emp1 o1, Emp1 o2)

{

return o1.getName().compareTo(o2.getName());

}

}

import java.util.Comparator;

public class SCompare implements Comparator<Emp1>{

public int compare(Emp1 o1, Emp1 o2)

{

return o1.getDept().compareTo(o2.getDept());

}

}

import java.util.\*;

import java.util.TreeSet;

public class Compare

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.println("You want to sort in order of \n\n1.ID\n2.Department\n3.Name\n4.Salary\n\nEnter your option: ");

int option = sc.nextInt();

switch(option)

{

case 1:

TreeSet<Emp1> tset = new TreeSet<Emp1>(new Idcompare());

tset.add(new Emp1(1,"Trainee","Sam",18000.0));

tset.add(new Emp1(2,"Manager","Rob",32000.0));

tset.add(new Emp1(3,"Analyst","Tom",21000.0));

System.out.println(" Increasing Order with the Id : ");

for(Emp1 o : tset)

{

System.out.print(o.getId()+","+o.getDept()+","+o.getName()+","+o.getSal());

System.out.println();

}

break;

case 2:

TreeSet<Emp1> tset1 = new TreeSet<Emp1>(new Ncompare());

tset1.add(new Emp1(1,"Trainee","Sam",18000.0));

tset1.add(new Emp1(2,"Manager","Rob",32000.0));

tset1.add(new Emp1(3,"Analyst","Tom",21000.0));

System.out.println(" Increasing Order with the Name : ");

for(Emp1 o : tset1)

{

System.out.print(o.getId()+","+o.getDept()+","+o.getName()+","+o.getSal());

System.out.println();

}

break;

case 3:

TreeSet<Emp1> tset2 = new TreeSet<Emp1>(new Dcompare());

tset2.add(new Emp1(1,"Trainee","Sam",18000.0));

tset2.add(new Emp1(2,"Manager","Rob",32000.0));

tset2.add(new Emp1(3,"Analyst","Tom",21000.0));

System.out.println(" Increasing Order with the Department : ");

for(Emp1 o : tset2)

{

System.out.print(o.getId()+","+o.getDept()+","+o.getName()+","+o.getSal());

System.out.println();

}

break;

case 4:

TreeSet<Emp1> tset3 = new TreeSet<Emp1>(new SCompare());

tset3.add(new Emp1(1,"Trainee","Sam",18000.0));

tset3.add(new Emp1(2,"Manager","Rob",32000.0));

tset3.add(new Emp1(3,"Analyst","Tom",21000.0));

System.out.println(" Increasing Order with the Salary : ");

for(Emp1 o : tset3)

{

System.out.print(o.getId()+","+o.getDept()+","+o.getName()+","+o.getSal());

System.out.println();

}

break;

}

}

}

OUTPUT :

You want to sort in order of

1.ID

2.Department

3.Name

4.Salary

Enter your option:

2

Increasing Order with the Name :

3,Tom,Analyst,21000.0

2,Rob,Manager,32000.0

1,Sam,Trainee,18000.0

**4]ANS:-**

**A]:-**

import java.time.LocalDate;

import java.time.LocalDateTime;

import java.time.format.DateTimeFormatter;

import java.time.format.FormatStyle;

import java.util.\*;

public class Leap

{

public static void main(String[] args)

{

LocalDate cal = LocalDate.of(2000, 12, 23);

List<LocalDate> calendarList = new LinkedList<>();

calendarList.add(cal);

for (LocalDate c : calendarList)

{

String formattedDate = c.format(DateTimeFormatter.ofPattern("dd-MM-yyyy"));

if (c.isLeapYear())

{

System.out.println("Your Date is "+formattedDate + " is a leap year");

} else

{

System.out.println(("Your Date is "+formattedDate + " is not a leap year");

}

}

}

}

**OUTPUT:-**

Your Date is 23-12-2000 is a leap year

**B]:-**

import java.time.LocalDate;

import java.time.LocalDateTime;

import java.time.format.DateTimeFormatter;

import java.time.format.FormatStyle;

import java.util.\*;

public class Leap

{

public static void main(String[] args)

{

LocalDate cal = LocalDate.of(2001, 12, 23);

List<LocalDate> calendarList = new LinkedList<>();

calendarList.add(cal);

for (LocalDate c : calendarList)

{

String formattedDate = c.format(DateTimeFormatter.ofPattern("dd-MM-yyyy"));

if (c.isLeapYear())

{

System.out.println(("Your Date is "+formattedDate + " is a leap year");

} else

{

System.out.println(("Your Date is "+formattedDate + " is not a leap year");

}

}

}

}

**OUTPUT :-**

Your Date is 23-12-2001 is not a leap year