1.Given a TreeMap<Long, Contact> which has phone numbers for keys and contact objects for values.

Write solutions to

* 1. Fetch all the keys and print them,
  2. Fetch all the values and print them
  3. Print all key-value pairs

**import** java.util.TreeMap;

**public** **class** Collection1 {

**public** **static** **void** main(String[] args)

{

TreeMap<Long,String> contact=**new** TreeMap<>();

contact.put((**long**) 992262470, "Mani");

contact.put((**long**) 943388789, "juhi");

contact.put((**long**) 987456983, "Anu");

contact.put((**long**) 969874152, "Dany");

System.***out***.println(contact.keySet());

System.***out***.println(contact.values());

System.***out***.println(contact);

}

}

2.Write an application to store 10 unique product objects. In case there is an attempt to add a duplicate product, it should be silently rejected. Hint: **Use** HasSet or TreeSet

**import** java.util.HashSet;

**public** **class** Collection2 {

**public** **static** **void** main(String[] args)

{

HashSet<String> hs=**new** HashSet<>();

hs.add("Titan");

hs.add("Fastrack");

hs.add("Casio");

hs.add("Fossil");

hs.add("Rolex");

System.***out***.println("Original");

System.***out***.println(hs);

System.***out***.println(hs.size());

hs.add("Rolex");

hs.add("Fossil");

System.***out***.println("After adding duplicate elements");

System.***out***.println(hs);

System.***out***.println(hs.size());

}

}

3.Store at least 10 Employee Objects in an TreeSet<Employee>. When the application runs the user should be asked to select one of the options upon which you will print the employee details in a sorted manner.

**For E.g.**

Run Application:

1. ID
2. Name
3. Department
4. Salary

**import** java.util.\*;

**class** Employee

{

**int** id,salary;

String name,dept;

**public** Employee(**int** id, String name, String dept, **int** salary) {

**this**.id=id;

**this**.name=name;

**this**.dept=dept;

**this**.salary=salary;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **int** getSalary() {

**return** salary;

}

**public** **void** setSalary(**int** salary) {

**this**.salary = salary;

}

**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;

}

@Override

**public** String toString() {

**return** "Employee [id=" + id + ", salary=" + salary + ", name=" + name + ", dept=" + dept + "]";

}

}

**class** IdComparator **implements** Comparator<Employee>{

@Override

**public** **int** compare(Employee o1, Employee o2) {

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

}

}

**class** NameComparator **implements** Comparator<Employee>{

@Override

**public** **int** compare(Employee o1, Employee o2) {

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

}

}

**class** DeptComparator **implements** Comparator<Employee>{

**public** **int** compare(Employee o1, Employee o2){

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

}

}

**class** SalaryComparator **implements** Comparator<Employee>{

**public** **int** compare(Employee o1, Employee o2){

**return** o1.getSalary()-o2.getSalary();

}

}

**public** **class** Collection3 {

**public** **static** **void** main(String[] args)

{

System.***out***.println("Please Select an Option");

System.***out***.println("a) Id\nb) Name\nc) Department\nd) Salary");

Scanner sc=**new** Scanner(System.***in***);

String select=sc.nextLine();

**if**(select.equals("a"))

{

TreeSet<Employee> emp=**new** TreeSet<>(**new** IdComparator());

emp.add(**new** Employee(101,"Aman","CSE",100000));

emp.add(**new** Employee(102,"Binod","ECE",75000));

emp.add(**new** Employee(103,"Daman","CSE",40000));

emp.add(**new** Employee(104,"Faruk","AERO",80000));

emp.add(**new** Employee(105,"John","EEE",60000));

emp.add(**new** Employee(106,"Brock","ECE",85000));

emp.add(**new** Employee(107,"Rock","AERO",45000));

emp.add(**new** Employee(108,"Alexa","CSE",95000));

emp.add(**new** Employee(109,"Nikki","ECE",110000));

emp.add(**new** Employee(110,"Roman","EEE",55000));

**for** (Employee employee : emp) {

System.***out***.println(employee.toString());

}

}

**else** **if**(select.equals("b"))

{

TreeSet<Employee> emp=**new** TreeSet<>(**new** NameComparator());

emp.add(**new** Employee(101,"Aman","CSE",100000));

emp.add(**new** Employee(102,"Binod","ECE",75000));

emp.add(**new** Employee(103,"Daman","CSE",40000));

emp.add(**new** Employee(104,"Faruk","AERO",80000));

emp.add(**new** Employee(105,"John","EEE",60000));

emp.add(**new** Employee(106,"Brock","ECE",85000));

emp.add(**new** Employee(107,"Rock","AERO",45000));

emp.add(**new** Employee(108,"Alexa","CSE",95000));

emp.add(**new** Employee(109,"Nikki","ECE",110000));

emp.add(**new** Employee(110,"Roman","EEE",55000));

**for** (Employee employee : emp) {

System.***out***.println(employee.toString());

}

}

**else** **if**(select.equals("c"))

{

TreeSet<Employee> emp=**new** TreeSet<>(**new** DeptComparator());

emp.add(**new** Employee(101,"Aman","CSE",100000));

emp.add(**new** Employee(102,"Binod","ECE",75000));

emp.add(**new** Employee(103,"Daman","CSE",40000));

emp.add(**new** Employee(104,"Faruk","AERO",80000));

emp.add(**new** Employee(105,"John","EEE",60000));

emp.add(**new** Employee(106,"Brock","ECE",85000));

emp.add(**new** Employee(107,"Rock","AERO",45000));

emp.add(**new** Employee(108,"Alexa","CSE",95000));

emp.add(**new** Employee(109,"Nikki","ECE",110000));

emp.add(**new** Employee(110,"Roman","EEE",55000));

**for** (Employee employee : emp) {

System.***out***.println(employee.toString());

}

}

**else** **if**(select.equals("d"))

{

TreeSet<Employee> emp=**new** TreeSet<>(**new** SalaryComparator());

emp.add(**new** Employee(101,"Aman","CSE",100000));

emp.add(**new** Employee(102,"Binod","ECE",75000));

emp.add(**new** Employee(103,"Daman","CSE",40000));

emp.add(**new** Employee(104,"Faruk","AERO",80000));

emp.add(**new** Employee(105,"John","EEE",60000));

emp.add(**new** Employee(106,"Brock","ECE",85000));

emp.add(**new** Employee(107,"Rock","AERO",45000));

emp.add(**new** Employee(108,"Alexa","CSE",95000));

emp.add(**new** Employee(109,"Nikki","ECE",110000));

emp.add(**new** Employee(110,"Roman","EEE",55000));

**for** (Employee employee : emp) {

System.***out***.println(employee.toString());

}

}

}

}

4.Given a LinkedList of Objects representing date of birth’s (use any inbuild java class to represent date), print the date’s along with the message: Your date of Birth is DD-MM-YYYY, and it (was or was not) a leap year.

E.g.

a) For the date 23-12-2000

Your date of birth is 23-12-2000 and it **was** a leap year

1. For the date 23-12-2001

Your date of birth is 23-12-2000 and it **was not** a leap year

**import** java.util.\*;

**public** **class** Collection4 {

**int** date,month,year;

**public** Collection4(**int** date,**int** month,**int** year){

**this**.date = date;

**this**.month = month;

**this**.year = year;

}

**public** String toString(){

**return** date+"-"+month+"-"+year;

}

**public** **int** getYear(){

**return** year;

}

**public** **static** **void** main(String[] args) {

LinkedList<Collection4> dob = **new** LinkedList<>();

dob.add(**new** Collection4(16,10,1999));

dob.add(**new** Collection4(13,04,1997));

dob.add(**new** Collection4(22,01,2000));

dob.add(**new** Collection4(06,07,2008));

dob.add(**new** Collection4(12,10,2011));

dob.add(**new** Collection4(19,05,2012));

dob.add(**new** Collection4(21,01,1992));

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

**int** year = dob.get(i).getYear();

**if** (year%400==0 || year%100==0 || year%4==0){

System.***out***.println(dob.get(i)+" : Born in Leap Year.");

}**else**{

System.***out***.println(dob.get(i)+" : Not Born in a Leap Year.");

}

}

}

}