1)

package collections;

import java.util.Map;

import java.util.Map.Entry;

import java.util.TreeMap;

public class Main1Contact {

public static void main(String[] args) {

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

Contact c1=new Contact("Akash","akash@123");

Contact c2=new Contact("Monisha","monisha@123");

Contact c3=new Contact("Amir","amir@123");

//Adding Books to map

map.put(9916376453L,c1);

map.put(9916376343L,c2);

map.put(9916376253L,c3);

System.out.println("All keys");

for(Entry<Long, Contact> entry:map.entrySet()){

long key=entry.getKey();

System.out.println(key);

}

System.out.println("All values");

for(Entry<Long, Contact> entry:map.entrySet()){

Contact c=entry.getValue();

System.out.println(c.name+" "+c.email);

}

System.out.println("All keys and values");

for(Entry<Long, Contact> entry:map.entrySet()){

long key=entry.getKey();

Contact c=entry.getValue();

System.out.println(key+": "+c.name+" "+c.email);

}

}

}

**package** collections;

**public** **class** Contact {

String name,email;

**public** Contact(String name,String email) {

**this**.email=email;

**this**.name=name;

}

}

2)

package collections;

import java.util.HashSet;

import java.util.Set;

public class Hash {

public static void main(String[] args) {

Set<String>products = new HashSet<>();

products.add("product1");

products.add("product2");

products.add("product3");

products.add("product4");

products.add("product5");

products.add("product6");

products.add("product7");

products.add("product8");

products.add("product9");

products.add("product10");

System.out.println(products);

products.add("product1");

System.out.println(products);

}

}

3)

package collections;

import java.util.Comparator;

import java.util.Scanner;

import java.util.TreeSet;

public class Main3 {

public static void main(String a[]){

Scanner sc=new Scanner(System.in);

System.out.println("Enter the a:To sort by ID, b: To sort by name, c:TO stort by Dept, d: to sorrt by Salary");

String x=sc.next();

//By using name comparator (String comparison)

if(x.equals("b")) {

TreeSet<Employee> nameComp = new TreeSet<Employee>(new MyNameComp());

nameComp.add(new Employee(1,"Ram","HR",3000));

nameComp.add(new Employee(2,"John","Manager",6000));

nameComp.add(new Employee(3,"Crish","R&D",2000));

nameComp.add(new Employee(4,"Tom","MA",2400));

for(Employee e:nameComp){

System.out.println(e);

}

}

else if(x.equals("d")) {

System.out.println("===========================");

//By using salary comparator (int comparison)

TreeSet<Employee> salComp = new TreeSet<Employee>(new MySalaryComp());

salComp.add(new Employee(1,"Ram","HR",3000));

salComp.add(new Employee(2,"John","Manager",6000));

salComp.add(new Employee(3,"Crish","R&D",2000));

salComp.add(new Employee(4,"Tom","MA",2400));

for(Employee e:salComp){

System.out.println(e);

}

}

else if(x.equals("a")) {

System.out.println("===========================");

//By using id comparator (int comparison)

TreeSet<Employee> idComp = new TreeSet<Employee>(new MyIdComp());

idComp.add(new Employee(3,"Crish","R&D",2000));

idComp.add(new Employee(1,"Ram","HR",3000));

idComp.add(new Employee(2,"John","Manager",6000));

idComp.add(new Employee(4,"Tom","MA",2400));

for(Employee e:idComp){

System.out.println(e);

}

}

else if(x.equals("c")) {

System.out.println("===========================");

//By using dept comparator (String comparison)

TreeSet<Employee> deptComp = new TreeSet<Employee>(new MyDeptComp());

deptComp.add(new Employee(1,"Ram","HR",3000));

deptComp.add(new Employee(2,"John","Manager",6000));

deptComp.add(new Employee(3,"Crish","R&D",2000));

deptComp.add(new Employee(4,"Tom","MA",2400));

for(Employee e:deptComp){

System.out.println(e);

}

}

else {

System.out.println("You have to enter any value from a,b,c,d");

}

sc.close();

}

}

class MyDeptComp implements Comparator<Employee>{

@Override

public int compare(Employee e1, Employee e2) {

return e1.getDept().compareTo(e2.getDept());

}

}

class MyNameComp implements Comparator<Employee>{

@Override

public int compare(Employee e1, Employee e2) {

return e1.getName().compareTo(e2.getName());

}

}

class MySalaryComp implements Comparator<Employee>{

@Override

public int compare(Employee e1, Employee e2) {

if(e1.getSalary() > e2.getSalary()){

return 1;

} else {

return -1;

}

}

}

class MyIdComp implements Comparator<Employee>{

@Override

public int compare(Employee e1, Employee e2) {

if(e1.getId() > e2.getId()){

return 1;

} else {

return -1;

}

}

}