1.

A> Printing only key from TreeMap….(K)

Copy below code and just remove getValue() method from for loop…..

**for**(Map.Entry m:map.entrySet()){

System.***out***.println(m.getKey());

}

B> Printing only value from TreeMap….(V)

Copy below code and just remove getKey() method from for loop…..

**for**(Map.Entry m:map.entrySet()){

System.***out***.println(m.getValue());

}

C> Printing all values in key value in a TreeMap…….(K,V)

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

**class** contact **implements** Comparable<contact>

{

**long** Phone;

String name;

String email;

//constructor for contact

**public** contact(String name,String email)

{

**this**.name = name;

**this**.email = email;

}

//this will return the string

**public** String toString()

{

**return** name +" "+email ;

}

//WE HAVE TO OVERIDE IT WHEN WE ARE IMPLEMENTING COMPARABLE INTERFACE

@Override

**public** **int** compareTo(contact o) {

**return** 0;

}

}

**public** **class** treemap {

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

// contact c= new contact();

//TREEMAP OBJECT....

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

//PUTTING KEY VALUE IN TREEMAP

map.put(77545646l,**new** contact("simon","simon@"));

map.put(756747l,**new** contact("giant","giant@"));

map.put(8979399511l,**new** contact("sumit","sumit@"));

map.put(67876867l,**new** contact("jack","jack@"));

map.put(56757657l,**new** contact("gotlib","gotlib@"));

//PRINTING VALUES......

**for**(Map.Entry m:map.entrySet()){

System.***out***.println(m.getKey()+" "+m.getValue());

}

}

} //above one is for ascending order..here below one is for descending order..

Map<Long, contact> reverseSortedMap = **new** TreeMap<>(Collections.*reverseOrder*());

reverseSortedMap.putAll(map);

System.***out***.println("=========descending order ============");

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

2. add 10 product key in a hashSet and confirm that it only take unique key……

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

**class** product

{

**int** id;

**public** product(**int** id)

{

**this**.id= id;

}

//override hashcode method

**public** **int** hashCode()

{

**return** id;

}

//logic for not getting same input again....

//override equals method

**public** **boolean** equals(Object o)

{

**if**(o **instanceof** product)

{

//downcasting the object

product i =(product)o;

**if**(i.id==**this**.id)

{

**return** **true**;

}**return** **false**;

}

**return** **false**;

}

//integer to String

**public** String toString()

{

**return** id +" ";

}

}

**public** **class** Hashsetproduct {

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

HashSet<product> set = **new** HashSet<>();

product p1=**new** product(10001);

product p2=**new** product(10002);

product p3=**new** product(10003);

product p4=**new** product(10003);

product p5=**new** product(10003);

product p6=**new** product(10004);

product p7=**new** product(10005);

product p8=**new** product(10006);

product p9=**new** product(10007);

product p10=**new** product(10007);

set.add(p1);

set.add(p2);

set.add(p3);

set.add(p4);

set.add(p5);

set.add(p6);

set.add(p7);

set.add(p8);

set.add(p9);

set.add(p10);

Iterator<product> i=set.iterator();

**while**(i.hasNext())

{

System.***out***.println(i.next());

}

}

}

3. Use TreeSet and store object of employee. make a interface for user to press 1 for printing id. 2 for printing name, 3 for printing department, 4 for printing salary…..

**package** newjava;

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

**class** empl **implements** Comparable<empl>

{

**int** id;

String name;

String department;

**double** salary;

**public** empl(**int** id,String name, String department,**double** salary)

{

**this**.id=id;

**this**.name=name;

**this**.department=department;

**this**.salary=salary;

}

**public** **void** getid()

{

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

}

**public** **void** getname()

{

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

}

**public** **void** getdepartment()

{

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

}

**public** **void** getsalary()

{

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

}

**public** **void** display1()

{

System.***out***.println(id + " "+" " +name +" "+" "+department+" " +" "+salary);

}

@Override

**public** **int** compareTo(empl o) {

**if**(id>o.id)

{

**return** 1;

}

**else** **if**(id<o.id)

{**return** -1;}

**else**

{

**return** 0;

}

}

}

**public** **class** useraskinput {

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

TreeSet<empl> emp= **new** TreeSet<>();

empl s = **new** empl(1,"thor","IT",1231);

empl s1 = **new** empl(2,"giant","HR",21231);

empl s2 = **new** empl(3,"kackie","food",51231);

empl s4= **new** empl(4,"loki","IT",1123231);

empl s5 = **new** empl(5,"giant","HR",241231);

empl s6 = **new** empl(6,"jasminee","food",234234);

emp.add(s);

emp.add(s1);

emp.add(s2);

emp.add(s4);

emp.add(s5);

emp.add(s6);

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

**int** user;

**while** (**true**){

System.***out***.println("Here is the operation which you can perform\n 1. getting id of employee. \n 2. getting name for employee\n 3. getting department for employee\n 4. getting salary for employee\n 5. printing all details");

user= sc.nextInt();

**switch** (user)

{

**case** 1:

**for** ( empl em:emp)

{

em.getid();

}

**break**;

**case** 2:

**for** ( empl em:emp)

{

em.getname();

}

**break**;

**case** 3:

**for** ( empl em:emp)

{

em.getdepartment();

}

**break**;

**case** 4:

**for** ( empl em:emp)

{

em.getsalary();

}

**break**;

**case** 5:

**for** ( empl em:emp)

{

em.display1();

}

**break**;

}

}}

}

}

4.