EXNO:2 Illustration of Collection framework by using Collection, Iterator and Comparator interfaces.

AIM:

To implement the collection framework by develop a java console application.

ALGORITHM:

Step 1 : Student class contains fields and age and a parameterized constructor.

Step 2: AgeComparator class defines comparison logic based on the age.

Step 3: NameComparator class provides comparison logic based on the name.

Step 4: FeesComparator class provides comparison logic based on the fees.

Step 5: Main class printing the values of the object by sorting on the basis of name, age and fees.

PROGRAM:

import java.io.\*;

import java.util.\*;

class Student {

int rollno;

String name;

float fees;

String branch;

int year;

int sem;

int age;

static String clg;

public Student(int rollno,String name,float fees,String branch,int year,int sem,int age) {

this.rollno = rollno;

this.name = name;

this.fees = fees;

this.branch = branch;

this.year = year;

this.sem = sem;

this.age = age;

clg="PU";

}

@Override

public String toString() {

return rollno + " "+ name + " " + fees + " " + branch + " " + year + sem + " " + age + " " + clg + "\n";

}

}

class AgeComparator implements Comparator {

public int compare(Object o1, Object o2) {

Student s1=(Student)o1;

Student s2=(Student)o2;

if(s1.age==s2.age)

return 0;

else if(s1.age>s2.age)

return 1;

else

return -1;

}

}

class NameComparator implements Comparator{

public int compare(Object o1, Object o2) {

Student s1=(Student)o1;

Student s2=(Student)o2;

return s1.name.compareTo(s2.name);

}

}

class FeesComparator implements Comparator {

public int compare(Object o1,Object o2) {

Student s1=(Student)o1;

Student s2=(Student)o2;

if(s1.fees==s2.fees)

return 0;

else if(s1.fees>s2.fees)

return 1;

else

return -1;

}

}

public class Temp1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

ArrayList sl=new ArrayList();

sl.add(new Student(1,"Shiva",10000.00f,"cse",1,1,18));

sl.add(new Student(2,"Venky",15000.00f,"ise",1,2,20));

sl.add(new Student(3,"Jesus",17000.00f,"ece",1,1,19));

sl.add(new Student(3,"Alla",12000.00f,"eee",1,1,19));

sl.add(new Student(3,"Budha",11000.00f,"mech",1,1,21));

System.out.println("Sorting by Name");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Collections.sort(sl,new NameComparator());

Iterator itr=sl.iterator();

while(itr.hasNext()){

Student st=(Student)itr.next();

System.out.println(st. );

}

System.out.println("Sorting by age");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Collections.sort(sl,new AgeComparator());

Iterator itr2=sl.iterator();

while(itr2.hasNext()) {

Student st=(Student)itr2.next();

System.out.println(st );

}

System.out.println("Sorting by fees");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Collections.sort(sl,new FeesComparator());

Iterator itr1=sl.iterator();

while(itr1.hasNext()){

Student st=(Student)itr1.next();

System.out.println(st);

}

}

}

RESULT:

Thus the collection framework was implemented by using its interfaces such as Collection, Iterator and Comparator that can be demonstrated with a java console application.