package Collections.Queue;  
import java.util.\*;  
  
public class PriorityQueueTest {  
 public static void main(String[] args) {  
  
 PriorityQueue<Integer> queue = new PriorityQueue<Integer>();  
 // Adding elements to the queue  
 boolean addValue = queue.add(20);  
 boolean offerValue = queue.offer(14);  
 System.*out*.println("addValue "+ addValue +" offerValue "+offerValue);  
 queue.add(30);  
 queue.offer(25);  
 queue.add(45);  
 queue.offer(50);  
 System.*out*.println("Priority queue values are: " + queue);  
 int removeValue = queue.remove();  
 System.*out*.println("Priority queue values are: " + queue+" removeValue "+removeValue);  
 int pollvalue = queue.poll();  
 System.*out*.println("Priority queue values are: " + queue+" pollvalue "+pollvalue);  
 int element = queue.element();  
 System.*out*.println("Priority queue values are: " + queue+" element "+element);  
 int peek = queue.peek();  
 System.*out*.println("Priority queue values are: " + queue+" peek "+peek);  
  
 // Equals test  
  
 PriorityQueue<Integer> queueClone = queue;  
 queueClone.add(25);  
 System.*out*.println("queue after queueClone " + queue);  
 Boolean hasAnimalEquals = queue.equals(queueClone);  
 System.*out*.println("hasAnimalEquals " + hasAnimalEquals);  
 PriorityQueue<Integer> queueClone1 = queue;  
 queueClone1.add(25);  
 queueClone1.add(25);  
 queueClone1.add(30);  
 queueClone1.add(50);  
 queueClone1.add(45);  
 Boolean hasAnimalEquals1 = queue.equals(queueClone);  
 System.*out*.println("hasAnimalEquals " + hasAnimalEquals1);  
 System.*out*.println("hashCode() " + queue.hashCode());  
  
 //unimplemented Collections methods  
  
 PriorityQueue<Integer> queue1 = new PriorityQueue<>();  
 queue1.add(1);  
 queue1.add(4);  
 queue1.add(3);  
 System.*out*.println("queue1 " + queue1);  
 Boolean hasQueueContains = queue.contains(45);  
 System.*out*.println("hasQueueContains " + hasQueueContains);  
 Boolean hasQueueContainsAll = queue.containsAll(queue1);  
 System.*out*.println("hasQueueContains " + hasQueueContainsAll);  
 queue.addAll(queue1);  
 System.*out*.println("queue after AddAll " + queue);  
 queue.retainAll(queue1);  
 System.*out*.println("queue after retainAll " + queue);  
 queue.removeAll(queue1);  
 System.*out*.println("queue after removeAll " + queue);  
 System.*out*.println("queue after isEmpty " + queue.isEmpty());  
 queue.add(12);  
 queue.add(45);  
 queue.add(60);  
 System.*out*.println("queue after New Elemets added " + queue);  
 System.*out*.println("Animals after AscendingIterator " + queue);  
 Integer[] finalQueueArray = queue.toArray(new Integer[0]);  
 for (Integer finalQueue : finalQueueArray) {  
 System.*out*.println(" " + finalQueue);  
 }  
 Object[] listAarray = queue.toArray();  
 int count = 0;  
 System.*out*.println("ArrayValues");  
 for (Object arrayedValues : listAarray) {  
 System.*out*.println("index " + count + "Arrayvalues " + arrayedValues);  
 count++;  
 }  
 System.*out*.println("queue after AscendingIterator " + queue);  
 queue.clear();  
 System.*out*.println("queue after clear " + queue);  
  
  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=61024:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace Collections.Queue.PriorityQueueTest

addValue true offerValue true

Priority queue values are: [14, 20, 30, 25, 45, 50]

Priority queue values are: [20, 25, 30, 50, 45] removeValue 14

Priority queue values are: [25, 45, 30, 50] pollvalue 20

Priority queue values are: [25, 45, 30, 50] element 25

Priority queue values are: [25, 45, 30, 50] peek 25

queue after queueClone [25, 25, 30, 50, 45]

hasAnimalEquals true

hasAnimalEquals true

hashCode() 1323165413

queue1 [1, 4, 3]

hasQueueContains true

hasQueueContains false

queue after AddAll [1, 25, 3, 30, 25, 4, 25, 50, 50, 45, 45, 30, 25]

queue after retainAll [1, 3, 4]

queue after removeAll []

queue after isEmpty true

queue after New Elemets added [12, 45, 60]

Animals after AscendingIterator [12, 45, 60]

12

45

60

ArrayValues

index 0 Arrayvalues 12

index 1 Arrayvalues 45

index 2 Arrayvalues 60

queue after AscendingIterator [12, 45, 60]

queue after clear []

Process finished with exit code 0