14. Notes - PriorityBlockingQueue

PriorityBlockingQueue -

PriorityBlockingQueue orders the elements through natural order if they are Comparable or we should use the Comparator.

Example -

```
import java.util.Comparator;
import java.util.concurrent.PriorityBlockingQueue;

class Student {
   String name;
   int rank;

   public Student(String name, int rank) {
       this.name = name;
       this.rank = rank;
   }

   public int getRank() {
       return this.rank;
   }

   public String toString() {
       return String.format("name : %s, rank : %d", name, rank);
   }
}

// Compares the Student objects based on the rank field value.
class StudentComparator implements Comparator<Student> {
   @Override
   public int compare(Student o1, Student o2) {
       return o1.getRank() - o2.getRank();
   }
}

public class Main {

   public static void main(String[] args) {
       PriorityBlockingQueue<Integer> queue = new PriorityBlockingQueue<Integer>();
       queue.add(10);
       queue.add(2);
       queue.add(5);

       System.out.println(queue.poll());
       System.out.println(queue.poll());
       System.out.println(queue.poll());
```

```
            PriorityBlockingQueue<Student> queue1 =
                   new PriorityBlockingQueue<Student>(5, new StudentComparator());

            queue1.add(new Student("a", 12));
            queue1.add(new Student("b", 1));
            queue1.add(new Student("c", 4));

            System.out.println(queue1.poll());
            System.out.println(queue1.poll());
            System.out.println(queue1.poll());
        }
    }
```

Output -

```
2
5
10
name : b, rank : 1
name : c, rank : 4
name : a, rank : 12
```