```java
// Lab 12: PriorityQueue
// Had to turn this in late, sorry :(
// Andrea Smith
// CSCI 1913

class PriorityQueue<Base>
{
  private class Node
  {
    private Base object;
    private int rank;
    private Node left;
    private Node right;

    private Node(Base object, int rank)
    {
      this.object = object;
      this.rank = rank;
      this.left = left;
      this.right = right;
    }
  }
  private Node root; // Root of BST
  public PriorityQueue()
  {
    root = new Node(null,-1);
  }


  public Base dequeue()
  {
    if (isEmpty())
    {
      throw new IllegalStateException("The priority queue is empty.");
    }
    else
    {
      Node temp = root;
      Node tempR = root.right;

      while(true)
      {
        if (tempR.left == null)
        {

          if (temp.left != tempR)
          {
            temp.right = tempR.right;
            return tempR.object;
          }
          else
          {
            temp.left = tempR.right;
            return tempR.object;
          }
        }

        else
        {
          temp = tempR;
          tempR = tempR.left;
        }
      }
    }
}

  public void enqueue(Base object, int rank)
  {
    if (rank < 0)
    {
      throw new IllegalArgumentException("Rank is negative.");
    }

    else
    {
      Node temp = root;
      while(true)
      {
        if (rank >= temp.rank)
        {
          if (temp.right != null)
          {
            temp = temp.right;
          }

          else
          {
            temp.right = new Node(object, rank);
            break;
          }
        }

        else
        {
          if(temp.left != null)
          {
            temp = temp.left;
          }

          else
          {
            temp.left = new Node(object, rank);
            break;
          }

        }
      }
    }
  }

  public boolean isEmpty()
  {
    return ((root.right == null) && (root.left == null));
  }

}


//  SNOBBERY. How the aristocracy behaves in a queue. 20 points.

class Snobbery
{

//  MAIN. Queue them up.

  public static void main(String[] args)
  {
    PriorityQueue<String> queue = new PriorityQueue<String>();

    System.out.println(queue.isEmpty());  //  true        2 points

    try
    {
      System.out.println(queue.dequeue());
    }
    catch (IllegalStateException ignore)
    {
      System.out.println("Blimey!");      //  Blimey!     2 points
    }
```

```java

    queue.enqueue("Lancelot",  5);
    queue.enqueue("Fawlty",    7);
    queue.enqueue("Elizabeth", 0);
    queue.enqueue("Charles",   1);
    queue.enqueue("Turing",    7);

    try
    {
      queue.enqueue("Zeus", -100);
    }
    catch (IllegalArgumentException ignore)
    {
      System.out.println("No gods!");     //  No gods!    2 points
    }

    System.out.println(queue.isEmpty());  //  false       2 points

    System.out.println(queue.dequeue());  //  Elizabeth   2 points
    System.out.println(queue.dequeue());  //  Charles     2 points
    System.out.println(queue.dequeue());  //  Lancelot    2 points
    System.out.println(queue.dequeue());  //  Turing      2 points
    System.out.println(queue.dequeue());  //  Fawlty      2 points

//  It's OK if Fawlty comes out before Turing, but both must come
out last.

    System.out.println(queue.isEmpty());  //  true        2 points.
  }

}

```