PROGRAMACIÓN JAVA **Nivel:** **Sinxelo**
Puntos: **3**

Se pide unha implantación da interface java.util.Queue ***no package utils*** utilizando unha estrutura de datos dinámica que permita almacenar calquera tipo de obxectos facendo uso de tipos xenéricos.

Unha Queue é unha estrutura de tipo FIFO, na que o elemento que se lee do Queue é o que leva almacenado máis tempo (está nun índice menor).

Os obxectos desta clase deben ser Iterable sobre os elementos da Queue de xeito que cando se percorren os elementos mediante un ***iterator***, mediante un ***for*** ou mediante un ***forEach*** se van eliminando da cola.

Non se debe permitir insertar valores nulos. No caso de intentar insertar un elemento nulo debe lanzar unha ***NullPointerException*** coa mensaxe de "Esta Queue non acepta valores nulos"

PD: Consulta en internet o API que debes implantar.

```
import utils.Queue;

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

      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("Engadindo strings de 1 a 5");
      queue.offer("String 1");
      queue.offer("String 2");
      queue.add("String 3");
      queue.add("String 4");
      queue.add("String 5");
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nObtido: "+queue.peek());
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nExtraido: "+queue.poll());
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nExtraido: "+queue.poll());
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nResto de elementos: ");
      for(String s:queue) System.out.println("Extraido: "+s);
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nEngadindo Strings 1 e 2");
      queue.offer("String 1");
      queue.offer("String 2");
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nEliminando string: "+queue.peek());
      queue.remove();
      System.out.println("Obtido: "+queue.peek());
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());
      System.out.println("\nExtraido: "+queue.poll());
      System.out.println("ELEMENTOS EN QUEUE: "+queue.size());

      try {
         queue.add(null);
      } catch(Exception e) {
         System.out.println("ERROR: "+e.getMessage());
      }
   }
}
```

A execución deste programa debería retornar o seguinte:

```
ELEMENTOS EN QUEUE: 0
Engadindo strings de 1 a 5
ELEMENTOS EN QUEUE: 5

Obtido: String 1
ELEMENTOS EN QUEUE: 5

Extraido: String 1
ELEMENTOS EN QUEUE: 4

Extraido: String 2
ELEMENTOS EN QUEUE: 3

Resto de elementos:
Extraido: String 3
Extraido: String 4
Extraido: String 5
ELEMENTOS EN QUEUE: 0

Engadindo Strings 1 e 2
ELEMENTOS EN QUEUE: 2

Eliminando string: String 1
Obtido: String 2
ELEMENTOS EN QUEUE: 1

Extraido: String 2
ELEMENTOS EN QUEUE: 0
ERROR: Esta Queue non acepta valores nulos
```

Criterios de Avaliación

• A clase e os seus construtores están correctamente definidos	0.25
• A definición da clase permite reducir a cantidade de código empregado	0.15
• A estrutura elexida é axeitada	0.10
• Os métodos add e offer funcionan correctamente	0.25
• Os métodos remove e poll funcionan correctamente	0.50
• Os métodos element e peek funcionan correctamente	0.50
• A clase está completamente definida	0.25
• O método iterator() retorna un Iterator correcto.	0.50