```java
import HelloWorld.WaitingLineKernel;

public class WaitingLineSecondary {
    public interface WaitingLine<T> extends WaitingLineKernel<T> {

        @Override
        public final boolean equals(Object obj) {
            if (obj == this) {
                return true;
            }
            if (obj == null) {
                return false;
            }
            if (!(obj instanceof Queue<?>)) {
                return false;
            }
            Queue<?> q = (Queue<?>) obj;
            if (this.lengthOfLine() != q.length()) {
                return false;
            }
            Iterator<T> it1 = this.iterator();
            Iterator<?> it2 = q.iterator();
            while (it1.hasNext()) {
                T x1 = it1.next();
                Object x2 = it2.next();
                if (!x1.equals(x2)) {
                    return false;
                }
            }
            return true;
        }

        @Override
        public int hashCode() {
            final int samples = 3;
            final int a = 20;
            final int b = 10;
            int result = 0;
            int n = 0;
            Iterator<T> it = this.iterator();
            while (n < samples && it.hasNext()) {
                n++;
                T x = it.next();
                result = a * result + b * x.hashCode();
            }
            return result;
        }

        @Override
        public String toString() {
            StringBuilder result = new StringBuilder("<");
            Iterator<T> it = this.iterator();
            while (it.hasNext()) {
                result.append(it.next());
                if (it.hasNext()) {
                    result.append(",");
                }
            }
            result.append(">");
            return result.toString();
        }

        /**
         * Replaces the entry in {@code this} at position
 {@code pos} with {@code x}
         * , and returns the old entry.
         *
         * @param pos
         *           the position to replace
         * @param x
         *           the new entry at position {@code pos}
         * @return the old entry at position {@code pos}
         * @aliases reference {@code x}
         * @updates this
         * @clear x
         * @requires
         *
         *          <pre>
         * {@code  this /= <>, 0 <= pos and pos < |this|}
         *          </pre>
         *
         * @ensures
         *
         *         <pre>
         * {@code this = #this[0, pos) * <x> * #this[pos+1, |
 #this|) and
         * <replaceEntry> = #this[pos, pos+1)}
```

```java
         *          </pre>
         */

        @Override
        public T replaceEntry(int pos, T x) {
            T removed = null;
            int length = this.lengthOfLine();
            for (int i = 0; i < length; i++) {
                if (i == pos) {
                    removed = this.removeFront();
                    this.addLine(x);
                } else {
                    this.addLine(this.removeFront());
                }

            }
            return removed;

        }

    }
}

```