Homework 18

Shyam Sai Bethina

09 November 2021

```java
//1
/**
 * Removes and returns the minimum value from {@code q} according to the
 * ordering provided by the {@code compare} method from {@code order}.
 *
 * @param q
 *            the queue
 * @param order
 *            ordering by which to compare entries
 * @return the minimum value from {@code q}
 * @updates q
 * @requires <pre>
 * q /= empty_string  and
 *  [the relation computed by order.compare is a total preorder]
 * </pre>
 * @ensures <pre>
 * perms(q * <removeMin>, #q)  and
 *  for all x: string of character
 *      where (x is in entries (q))
 *    ([relation computed by order.compare method](removeMin, x))
 * </pre>
 */
private static String removeMin(Queue<String> q, Comparator<String> order) {
    String result = "";
    for (String s : q) {
        if (order.compare(s, result) < 0) {
            result = s;
        }
    }

    int i = 0;
    while (i < q.length()) {
        String s = q.dequeue();
        if (!s.equals(result)) {
            q.enqueue(s);
        }
    }
    return result;
}
```

```java
//2
/**
 * Sorts {@code q} according to the ordering provided by the {@code compare}
 * method from {@code order}.
 *
 * @param q
 *            the queue
 * @param order
 *            ordering by which to sort
 * @updates q
 * @requires [the relation computed by order.compare is a total preorder]
 * @ensures q = [#q ordered by the relation computed by order.compare]
 */
public static void sort(Queue<String> q, Comparator<String> order) {
    Queue<String> result = q.newInstance();
    while (q.length() != 0) {
        String min = removeMin(q, order);
        result.enqueue(min);
    }
    q.transferFrom(result);
}
```