Homework 15

Shyam Sai Bethina

01 November 2021

```
//1
/**
 * Reports the smallest integer in the given {@code Queue<Integer>}.
 *
 * @param q
 *            the queue of integer
 * @return the smallest integer in the given queue
 * @requires q /= empty_string
 * @ensures <pre>
 * min is in entries(q) and
 * for all x: integer
 *     where (x is in entries(q))
 *   (min <= x)
 * </pre>
 */
private static int min(Queue<Integer> q) {
    int min = q.dequeue();
    while (q.length() != 0) {
        int temp = q.dequeue();
        if (temp > min) {
            min = temp;
        }
    }

    return min;

}
```

1)i) Because the deque method is being used, and it requires clause states that the Queue length cannot be zero, so it must be included in the min() requires clause too

ii) This is to make sure q stays the same through the method. If this were not there, then q itself could've been

```java
//2
/**
 * Reports an array of two {@code int}s with the smallest and the largest
 * integer in the given {@code Queue<Integer>}.
 *
 * @param q
 *            the queue of integer
 * @return an array of two {@code int}s with the smallest and the largest
 *         integer in the given queue
 * @requires q /= empty_string
 * @ensures <pre>
 * { minAndMax[0], minAndMax[1] } is subset of entries(q) and
 * for all x: integer
 *     where (x in in entries(q))
 *   (minAndMax[0] <= x <= minAndMax[1])
 * </pre>
 */
private static int[] minAndMax(Queue<Integer> q) {
    int min = q.dequeue();
    int max = min;
    while (q.length() != 0) {
        int temp = q.dequeue();
        if (temp > min) {
            min = temp;
        }
        if (temp < max) {
            max = temp;
        }

    }
    int[] result = { min, max };
    return result;
}
```

```java
//3
private static int[] minAndMax2(Queue<Integer> q) {
    int min = q.dequeue();
    int max = q.dequeue();
    if (min > max) {
        int temp = max;
        max = min;
        min = temp;
    }
    while (q.length() != 0) {
        int comp1 = q.dequeue();
        int comp2 = q.dequeue();
        if (comp1 > comp2) {
            int temp2 = comp1;
            comp1 = comp2;
            comp2 = temp2;
        }
        if (comp1 < min) {
            min = comp1;
        }
        if (comp2 > max) {
            max = comp2;
        }
    }
    int[] result = { min, max };
    return result;

}
```