```java

import components.map.Map;
import components.map.Map.Pair;
import components.queue.Queue;

/**
 * {@code Queue} represented as a {@code Sequence} of entries, with
 * implementations of primary methods.
 *
 * @param <T>
 *            type of {@code Queue} entries
 * @correspondence this = $this.entries
 */
public class HelloWorld {

    /**
     * Finds pair with first component {@code key} and, if such exists, moves it
     * to the front of {@code q}.
     *
     * @param <K>
     *            type of {@code Pair} key
     * @param <V>
     *            type of {@code Pair} value
     * @param q
     *            the {@code Queue} to be searched
     * @param key
     *            the key to be searched for
     * @updates q
     * @ensures <pre>
     * perms(q, #q)  and
     * if there exists value: V (<(key, value)> is substring of q)
     *  then there exists value: V (<(key, value)> is prefix of q)
     * </pre>
     */
    private static <K, V> void moveToFront(Queue<Pair<K, V>> q, K key) {
        boolean included = false;
        int i = 0;
        Queue<Pair<K, V>> temp = q.newInstance();
        while (i < q.length()) {
            Map.Pair<K, V> tempPair = q.dequeue();
            if (key != tempPair.key()) {
                q.enqueue(tempPair);
            } else {
                temp.enqueue(tempPair);
            }
            i++;
        }

        temp.append(q);
        q.transferFrom(temp);

    }
}
```