```java
import components.queue.Queue;
import components.queue.Queue1L;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;

/**
 * Put a short phrase describing the program here.
 *
 * @author Sam Espanioly
 *
 */
public final class HW17 {

    /**
     * Private constructor so this utility class cannot be
  instantiated.
     */
    private HW17() {
    }

    /**
     * Reverses ("flips") {@code this}.
     *
     * @param <T>
     *
     * @updates this
     * @ensures this = rev(#this)
     */
    public <T> void flip() {
        Queue<T> temp = new Queue1L<>();
        //temp.copyFrom(this);
        int len = this.length();
```

```
34          for (int i = 0; i < len; i++) {
35              temp = this.dequeue();
36              this.enqueue(temp);
37          }
38      }
39
40      /**
41       * Reverses ("flips") {@code this}.
42       *
43       * @param <T>
44       *
45       * @updates this
46       * @ensures this = rev(#this)
47       *///static
48      public static void flip2(Queue<T> q) {
49          Queue<T> temp = new Queue1L<>();
50          int len = q.length();
51          for (int i = 0; i < len; i++) {
52              temp = q.dequeue();
53              q.enqueue(temp);
54          }
55      }
56
57      /**
58       * Main method.
59       *
60       * @param args
61       *            the command line arguments
62       */
63      public static void main(String[] args) {
64          SimpleReader in = new SimpleReader1L();
65          SimpleWriter out = new SimpleWriter1L();
66  //part 3. Genome Reassembly
67  //          TATACAT
```

```
68 //         AGCTGTTTTCGTT
69 //         CACTCCATTTTA
70 //         CATTTTAGCTGTT
71 //         TTTCGTTATACAT
72 //         CTGTTTTCGTTA
73         //first we do TTTCGTTATACAT && TATACAT
74 //      AGCTGTTTTCGTT
75 //      CACTCCATTTTA
76 //      CATTTTAGCTGTT
77 //      TTTCGTTATACAT
78 //      CTGTTTTCGTTA
79 //         second we do TTTCGTTATACAT && CATTTTAGCTGTT
80 //      AGCTGTTTTCGTT
81 //      CACTCCATTTTA
82 //      TTTCGTTATACATTTTAGCTGTT
83 //      CTGTTTTCGTTA
84         //third we do TTTCGTTATACATTTTAGCTGTT &&
   AGCTGTTTTCGTT
85 //      CACTCCATTTTA
86 //      TTTCGTTATACATTTTAGCTGTTTTCGTT
87 //      CTGTTTTCGTTA
88         //fourth we do TTTCGTTATACATTTTAGCTGTTTTCGTT &&
   CTGTTTTCGTTA
89 //      CACTCCATTTTA
90 //      TTTCGTTATACATTTTAGCTGTTTTCGTTTTCGTTA
91         //thats the result
92         //but if we flip CACTCCATTTTA becomes ATTTTACCTCAC
93         //then we can merge them
94         //TTTCGTTATACATTTTAGCTGTTTTCGTTTTCGTTA &&
   ATTTTACCTCAC
95         //resulting in
   TTTCGTTATACATTTTAGCTGTTTTCGTTTTCGTTATTTTACCTCAC
96
97         in.close();
```

```
 98         out.close();
 99     }
100 }
101
```