```
1
2  /**
3   * Classwork #6: Writing "while" loops within methods
4   * Objectives:
5   *     - Continue practicing with writing method headers
6   *     - Become familiar with Java Documentation for describing methods
7   *     - Declare local variables to assist with method definitions
8   *
9   *     - Identify "end" conditions for while loops
10  *     - Consider the order of statement execution within a "while" loop
11  *     - Learn how to break out of a "while" loop with a return statement
12  */
13
14 public class MethodsAndLoops
15 {
16     public static void main(String[] args)
17     {
18         // Test Methods #1-3 in this space
19         S.O.P. ( getLastIndex ("Kelley", "e")); // prints 4
20
21         S.O.P. ( getIndex ("Kelley", "e")); // prints 1
22         S.O.P. ( getIndexAfter ("Kelleyee", "e", 5)); // prints 6
23
24
25
26
27
28
29     }
30
31     /**
32      * Method #1: Define a method that will find and return the last existing index
33      * of a character within a given String.
34      *
35      * @param inputString This is the String that we are searching through
36      * @param character This is the character we are looking for in inputString
37      *
38      * @return The last existing index of character within inputString
39      */
40      public int getLastIndex(String inputString, String characters)
41      {
42          int index = -1;
43          int i = 0;
44          while (i < inputString.length())
45          {
46              String letter = inputString.substring(i, i+1);
47              if (letter.equals(character))
48                  index = i;
49              i++;
50          }
            return index;
        }
```

```
51      /**
52      * Method #2: Define a method that will find the first existing index
53      * of a character within a given String. (This similar to the indexOf method
54      * with one paramter)
55      *
56      * @param inputString This is the String that we are searching through
57      * @param character This is the character we are looking for in inputString
58      *
59      * @return The first existing index of character within inputString
60      */
61      public int getIndex(String inputString, String character)
62      {
63              int index = -1;
64              int i = 0
65              while (i < inputString.length())
66                  if (inputString.substring(i,i+1).equals(character))
67                  { return i;
68                    i++; }
69      }       return index;
70      /**
71      * Method #3: Define a method that will start at a given index within
72      * a String, and will search the remaining String data after that index,
73      * to find and return the next exisiting index of a character within
74      * that String
75      *
76      * @param inputString This is the String that we are searching through
77      * @param character This is the character we are looking for in inputString
78      * @param indexStart This is the index we will start at
79      *
80      * @return The first existing index of character within inputString,
81      *         located after indexStart
82      */
83      public int getIndexAfter(String inputString, String
84                               character, int indexStart)
85
86
87
88      {
89
90              int i = indexStart;
91              while (i < inputString.length())
92
93              { if (inputString.substring(i,i+1).equals(charater))
94                    return i;
95  }             i++; }
96              return -1;
```