```java
import static org.junit.Assert.assertEquals;

public class GlossaryTest {
    /*
     * tests of generateElements
     */
    @Test
    public void testGenerateElements1() {
        String str = " . !   ";
        Set<Character> charSet = new Set1L<>();
        Glossary.generateElements(str, charSet);
        Set<Character> charSetExpected = charSet.newInstance();
        charSetExpected.add(' ');
        charSetExpected.add('.');
        charSetExpected.add('!');
        assertEquals(" . !   ", str);
        assertEquals(charSetExpected, charSet);
    }

    @Test
    public void testGenerateElements2() {
        String str = "heyo";
        Set<Character> charSet = new Set1L<>();
        Glossary.generateElements(str, charSet);
        Set<Character> charSetExpected = charSet.newInstance();
        charSetExpected.add('h');
        charSetExpected.add('e');
        charSetExpected.add('y');
        charSetExpected.add('o');
        assertEquals("heyo", str);
        assertEquals(charSetExpected, charSet);
    }

    //test about border
    @Test
    public void testGenerateElements3() {
        String str = "c";
        Set<Character> charSet = new Set1L<>();
        Glossary.generateElements(str, charSet);
        Set<Character> charSetExpected = charSet.newInstance();
        charSetExpected.add('c');
        assertEquals("c", str);
        assertEquals(charSetExpected, charSet);
    }

    /*
     * tests of nextWordOrSeparator
     */
    @Test
    public void testNextWordOrSeparator1() {
        int position = 0;
        String text = "u";
        final String separStr = " \t ";
        Set<Character> separators = new Set1L<>();
        Glossary.generateElements(separStr, separators);
        String res = Glossary.nextWordOrSeparator(text, position, separators);
        assertEquals("u", res);
    }

    @Test
    public void testNextWordOrSeparator2() {
        int position = 0;
```

```
74          String text = "halo.hi";
75          final String separStr = " . ";
76          Set<Character> separators = new Set1L<>();
77          Glossary.generateElements(separStr, separators);
78          String res = Glossary.nextWordOrSeparator(text, position, separators);
79          assertEquals("halo", res);
80      }
81
82      @Test
83      public void testNextWordOrSeparator3() {
84          String text = "...";
85          int position = 0;
86          final String separStr = " . ";
87          Set<Character> separators = new Set1L<>();
88          Glossary.generateElements(separStr, separators);
89          String res = Glossary.nextWordOrSeparator(text, position, separators);
90          assertEquals(".", res);
91      }
92
93      @Test
94      public void testNextWordOrSeparator4() {
95          String text = "hello..";
96          int position = 5;
97          final String separStr = " . ";
98          Set<Character> separators = new Set1L<>();
99          Glossary.generateElements(separStr, separators);
100         String res = Glossary.nextWordOrSeparator(text, position, separators);
101         assertEquals(".", res);
102     }
103
104     /*
105      * tests of generateIn
106      */
107
108     @Test
109     public void testGenereateIn1() {
110         SimpleReader in = new SimpleReader1L("test1.txt");
111         Queue<String> title = new Queue1L<>();
112         Map<String, String> word = new Map1L<>();
113         title = Glossary.generateIn(word, in);
114         Queue<String> titleExpected = title.newInstance();
115         titleExpected.enqueue("meaning");
116         Map<String, String> wordExpected = new Map1L<>();
117         wordExpected.add("meaning",
118                 "something that one wishes to convey, especially by language");
119
120         assertEquals(titleExpected, title);
121         assertEquals(wordExpected, word);
122     }
123
124     @Test
125     public void testGenereateIn2() {
126         SimpleReader in = new SimpleReader1L("test2.txt");
127         Queue<String> title = new Queue1L<>();
128         Map<String, String> word = new Map1L<>();
129         title = Glossary.generateIn(word, in);
130         Queue<String> titleExpected = title.newInstance();
131         titleExpected.enqueue("meaning");
132         titleExpected.enqueue("term");
133         Map<String, String> wordExpected = new Map1L<>();
134         wordExpected.add("meaning",
135                 "something that one wishes to convey, especially by language");
```

```
136         wordExpected.add("term", "a word whose definition is in a glossary");
137
138         assertEquals(titleExpected, title);
139         assertEquals(wordExpected, word);
140     }
141
142     @Test
143     public void testGenereateIn3() {
144         SimpleReader in = new SimpleReader1L("test3.txt");
145         Queue<String> title = new Queue1L<>();
146         Map<String, String> word = new Map1L<>();
147         title = Glossary.generateIn(word, in);
148         Queue<String> titleExpected = title.newInstance();
149         titleExpected.enqueue("meaning");
150         titleExpected.enqueue("glossary");
151         Map<String, String> wordExpected = new Map1L<>();
152         wordExpected.add("meaning",
153                 "something that one wishes to convey, especially by language");
154         wordExpected.add("glossary",
155                 "a list of difficult or specialized terms, with their definitions,usually
    near the end of a book");
156
157         assertEquals(titleExpected, title);
158         assertEquals(wordExpected, word);
159     }
160
161     @Test
162     public void changeTheTerms1() {
163         Map<String, String> word = new Map1L<>();
164         Queue<String> title = new Queue1L<>();
165         word.add("book", "a printed or written literary work");
166         title.enqueue("book");
167         Set<Character> strSet = new Set1L<>();
168         strSet.add(' ');
169         strSet.add(',');
170         strSet.add('.');
171         Glossary.changeTheTerms(word, title, strSet);
172         Map<String, String> wordExpected = new Map1L<>();
173         wordExpected.add("book", "a printed or written literary work");
174         assertEquals(wordExpected, word);
175     }
176
177 }
178
```