```
/**
 * Generates the set of characters in the given {@code String}
into the
 * given {@code Set}.
 *
 * @param str
 *             the given {@code String}
 * @param charSet
 *             the {@code Set} to be replaced
 * @replaces charSet
 * @ensures charSet = entries(str)
 */
private static void generateElements(String str, Set<Character>
charSet) {
     for(int i = 0; i < charSet.size(); i++) {
          charSet.removeAny();
     }
     for(int i = 0; i < str.length(); i++) {
          if(!charSet.conatains(str.charAt(i)) {
               charSet.add(str.charAt(i);
          }
     }
}
```

```
/**
 * Returns the first "word" (maximal length string of characters
not in
 * {@code separators}) or "separator string" (maximal length
string of
 * characters in {@code separators}) in the given {@code text}
starting at
 * the given {@code position}.
 *
 * @param text
 *             the {@code String} from which to get the word or
separator
 *             string
 * @param position
```

```
 *             the starting index
 * @param separators
 *             the {@code Set} of separator characters
 * @return the first word or separator string found in {@code
text} starting
 *         at index {@code position}
 * @requires 0 <= position < |text|
 * @ensures <pre>
 * nextWordOrSeparator =
 *   text[position, position + |nextWordOrSeparator|)  and
 * if entries(text[position, position + 1)) intersection
separators = {}
 * then
 *   entries(nextWordOrSeparator) intersection separators = {}
and
 *   (position + |nextWordOrSeparator| = |text|  or
 *   entries(text[position, position + |nextWordOrSeparator| +
1))
 *   intersection separators /= {})
 * else
 *   entries(nextWordOrSeparator) is subset of separators  and
 *   (position + |nextWordOrSeparator| = |text|  or
 *   entries(text[position, position + |nextWordOrSeparator| +
1))
 *   is not subset of separators)
 * </pre>
 */
private static String nextWordOrSeparator(String text, int
position, Set<Character> separators) {
     String nextWordOrSeparator = new String();
     String subText = text.substring(position);
     char first = subText.charAt(0);
     if(separators.contains(first)) {
          for(int i = 0; i < subText.length() &&
separators.contains(subText.charAt(i)); i++) {
               nextWordOrSeparator += subtext.charAt(i);
          }
     } else {
          for(int i = 0; i < subText.length() &&
!separators.contains(subText.charAt(i)); i++) {
```

```
                nextWordOrSeparator += subtext.charAt(i);
            }
        }
        return nextWordOrSeparator;
}
```