```java
import components.map.Map;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;

/**
 *
 * @author Vaishnavi Kasabwala
 *
 */
public final class HW20 {

    /**
     * Private constructor so this utility class cannot be instantiated.
     */
    private HW20() {
    }

    /**
     * Inputs a "menu" of words (items) and their prices from the given file and
     * stores them in the given {@code Map}.
     *
     * @param fileName
     *            the name of the input file
     * @param priceMap
     *            the word -> price map
     * @replaces priceMap
     * @requires <pre>
     * [file named fileName exists but is not open, and has the
     *  format of one "word" (unique in the file) and one price (in cents)
     *  per line, with word and price separated by ','; the "word" may
     *  contain whitespace but no ',']
     * </pre>
     * @ensures [priceMap contains word -> price mapping from file fileName]
     */
    private static void getPriceMap(String fileName,
            Map<String, Integer> priceMap) {

        SimpleReader file = new SimpleReader1L(fileName);

        while (!file.atEOS()) {
            String str = file.nextLine();

            String s1 = "";
            String s2 = "";

            for (int i = 0; i < str.length(); i++) {
                if (str.charAt(i) == ',') {
                    s1 = str.substring(0, i);
                    s2 = str.substring(i + 1, str.length());
                }
            }

            int price = Integer.parseInt(s2);
            priceMap.add(s1, price);
        }
    }

```

```
58      /**
59       * Input one pizza order and compute and return the total price.
60       *
61       * @param input
62       *            the input stream
63       * @param sizePriceMap
64       *            the size -> price map
65       * @param toppingPriceMap
66       *            the topping -> price map
67       * @return the total price (in cents)
68       * @updates input
69       * @requires <pre>
70       * input.is_open and
71       * [input.content begins with a pizza order consisting of a size
72       *  (something defined in sizePriceMap) on the first line, followed
73       *  by zero or more toppings (something defined in toppingPriceMap)
74       *  each on a separate line, followed by an empty line]
75       * </pre>
76       * @ensures <pre>
77       * input.is_open and
78       * #input.content = [one pizza order (as described
79       *              in the requires clause)] * input.content and
80       * getOneOrder = [total price (in cents) of that pizza order]
81       * </pre>
82       */
83      private static int getOneOrder(SimpleReader input,
84              Map<String, Integer> sizePriceMap,
85              Map<String, Integer> toppingPriceMap) {
86
87          int total = 0;
88
89          while (!input.atEOS()) {
90              String str = input.nextLine();
91
92              if (sizePriceMap.hasKey(str)) {
93                  int price = sizePriceMap.value(str);
94                  total += price;
95              }
96
97              if (toppingPriceMap.hasKey(str)) {
98                  int extra = toppingPriceMap.value(str);
99                  total += extra;
100             }
101
102         }
103
104         return total;
105     }
106
107 }
```