Brendan Whitaker

CSE 2221 Homework 18

Professor Bucci

6 April 2017

1. We implement the map generation method:

```
/**
 * 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 input = new SimpleReader1L(fileName);
    Set<String> lines = new Set1L<String>();
    String str = "";
    //if the input is not at the end of stream
    while (!input.atEOS()) {
        //set str reference to next line
        str = input.nextLine();
```

```
            //add in str if not subset
            lines.add(str);
        }
        for (String elem : lines) {
            priceMap.add(elem.substring(0, elem.indexOf(',') - 1),
                    Integer.parseInt(elem.substring
                    (elem.indexOf(',') + 1,
                            elem.length())));
        }
        input.close();
    }
```

2. We price computation method:

```
    /**
     * Input one pizza order and compute and return the total price.
     *
     * @param input
     *            the input stream
     * @param sizePriceMap
     *            the size -> price map
     * @param toppingPriceMap
     *            the topping -> price map
     * @return the total price (in cents)
     * @updates input
     * @requires <pre>
     * input.is_open and
     * [input.content begins with a pizza order consisting of a size
     *  (something defined in sizePriceMap) on the first line, followed
     *  by zero or more toppings (something defined in toppingPriceMap)
     *  each on a separate line, followed by an empty line]
     * </pre>
     * @ensures <pre>
     * input.is_open and
     * #input.content = [one pizza order (as described
     *              in the requires clause)] * input.content and
     * getOneOrder = [total price (in cents) of that pizza order]
     * </pre>
     */
    private static int getOneOrder(SimpleReader input,
            Map<String, Integer> sizePriceMap,
            Map<String, Integer> toppingPriceMap) {
        int price = 0;
        Queue<String> lines = new Queue1L<String>();
        String str = "";
        //if the input is not at the end of stream
```

```
    while (!input.atEOS()) {
        //set str reference to next line
        str = input.nextLine();
        //add in str if not subset
        lines.enqueue(str);
    }
    price = sizePriceMap.value(lines.dequeue());
    for (String elem : lines) {
        price += toppingPriceMap.value(elem);
    }
    return price;
}
```