Homework 20

Farhan Sadeek

November 14, 2023

1.
```
/**
* 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 {@code priceMap}
* @requires <pre>
* {@code [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 not ',']}
* </pre>
* @ensures <pre>
* {@code [priceMap contains word -> price mapping from file fileName]}
* </pre>
*/

public static void getPriceMap(String fileName, Map<String, Integer> priceMap){
    SimpleReader input = new SimpleReader1L(fileName);
    while (!input.atEOS()) {
        String thisLine = input.nextLine();
        String key = thisLine.substring(0, thisLine.indexOf(','));
        String thisValue = thisLine.substring(thisLine.indexOf(',') + 1);
        int value = Integer.parseInt(thisValue);
        priceMap.add(key, value);
    }
    input.close();
}
```

2.
```
/**
* 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 {@code input}
* @requires <pre>
* {@code 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>
* {@code 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<Strin
    int size = sizePriceMap.value(input.nextLine());
    int price = size;
    String toppingString = input.nextLine();
    while(!toppingString.isEmpty()){
        int topping = toppingPriceMap.get(toppingString);
        price += topping;
        toppingString = input.nextLine();
    }
    return price;
}
```