```java
import components.map.Map;
import components.naturalnumber.NaturalNumber;
import components.naturalnumber.NaturalNumber2;
import components.simplereader.SimpleReader;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;

/**
 * Simple HelloWorld program (clear of Checkstyle and FindBugs warnings).
 *
 * @author Sam Espanioly
 */
public final class HelloWorldMaps {

    /**
     * Default constructor--private to prevent instantiation.
     */
    private HelloWorldMaps() {
        // no code needed here
    }

    /**
     * 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) {
        int len = fileName.length();
        int i = 0;
        int w = 0;
        int t = -1;
        String temp = "";
        String key = "";
        int value = 0;
        while (i < len) {
            if (fileName.charAt(i) == ',' && t < 0) {
                // the key for the maP
                key = fileName.substring(w, i);
                //switching assuming there's a price or a value after every key
                t = t * t;
            }
            if (fileName.charAt(i) == ',' && t > 0) {
                // putting the value in a string
                temp = fileName.substring(w, i);
                // converting to int
                value = Integer.parseInt(temp);
                t = t * t;
                // adding both key and value. This works only if we assume
                // the value comes after the key every time
                priceMap.add(key, value);
            }

            i++;
        }
    }

    /**
     * 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,
            // what's the point of this Map?
            Map<String, Integer> sizePriceMap,
            Map<String, Integer> toppingPriceMap) {
        SimpleWriter out = new SimpleWriter1L();
        int total = 0;
        out.print("Enter the topping you would like to
add: ");
        String t = input.nextLine();
        //while loop in case user inputted wrong topping
        while (!toppingPriceMap.hasKey(t)) {
            out.print("Enter a correct topping you would
like to add: ");
            t = input.nextLine();
        }
        int temp = 0;
        // final price
        NaturalNumber price = new NaturalNumber2();
        // temporary value to find the value in the map
for the certain key
        NaturalNumber temp1 = new
NaturalNumber2(toppingPriceMap.value(t));
        // this is  why recursive can work in here
        price.add(temp1);
        // in case they want to add more
```

```java
        out.print("Would you like to add more? y/n");
        t = input.nextLine();
        // I was trying to do recursive method here
        if (t.equals("yes") || t.equals("y")) {
            getOneOrder(input, sizePriceMap, toppingPriceMap);
        }
        // converting to an int
        if (price.canConvertToInt()) {
            total = price.toInt();
            if (t.equals("no") || t.equals("n")) {
                out.println("Your total for topping is " + total
                        + " Thank you! Please come again.");
            }
        }
        return total;
    }

    /**
     * Main method.
     *
     * @param args
     *            the command line arguments; unused here
     */
    public static void main(String[] args) {
        SimpleWriter out = new SimpleWriter1L();
        out.println("Hello World!");
        out.close();
    }

}

```