Java Chapter 6: HashMaps

Contents

 No AI use.

Time required: 60 minutes

Do: Online Tutorial

Go through the following tutorials.

- Java HashMaps
- Java HashMap with Examples

Java HashMap

A Java HashMap is similar to a Python dictionary. It works with "**key:value**" pairs.

- A HashMap in Java is a collection that stores data in key-value pairs.
- It is a part of the "**java.util**" package and provides quick retrieval of data based on keys.
- Each key in a HashMap must be unique. Values can be duplicated. If a duplicate key is entered, it overwrites the existing element.

The keys and values can be any Java data type. The following example uses a String data type for the key and value.

Tutorial 1: FruitHashMap

Create a Java program called **FruitHashMap.java**

```
9   import java.util.HashMap;
```

Import the required HashMap package.

```
11  public class FruitHashMap {
        Run | Debug
12      public static void main(String[] args) {
13          // Create a HashMap with String key and Integer value
14          HashMap<String, Integer> myMap = new HashMap<String, Integer>();
```

Create a HashMap object specifying the types for key and value.

```
16          // Add key-value pairs to the 'myMap' HashMap
17          myMap.put("apple", 10);
18          myMap.put("orange", 20);
19          myMap.put("banana", 15);
```

Use the "**put()**" method to add elements to the HashMap. Here, "apple", "orange", and "banana" are keys, and 10, 20, and 15 are their corresponding values.

```
28          // Loop through each key in the HashMap 'myMap'
29          // 'myMap.keySet()' returns a set of all the keys in the HashMap
30          for (String entry : myMap.keySet()) {
31              // For each key (entry) in the set of keys, do the following:
32
33              // Print the key and its corresponding value
34              // 'myMap.get(entry)' retrieves the value associated with the key 'entry'
35              // 'System.out.println' prints the key and value to the console in the format
36              // "Key: <key>, Value: <value>"
37              System.out.println("Key: " + entry + ", Value: " + myMap.get(entry));
38          }
39
40      }
41  }
```

Use "**keySet()**" along with a for-each loop to iterate through the HashMap. This loop will print each key-value pair in the HashMap.

The **keySet()** method in the **HashMap** class returns a **Set** of all the keys contained in the HashMap. For each key we get and print the value.

Additional Operations and Methods:

- **Checking Existence of Keys or Values**: **containsKey(Object key)** and **containsValue(Object value)** methods determine whether a specific key or value exists in the HashMap.
- **Removing Elements:** Use **remove(Object key)** to delete a key-value pair based on the key provided.
- **Size of the HashMap**: **size()** method returns the number of key-value pairs present in the HashMap.

Example Input Loop

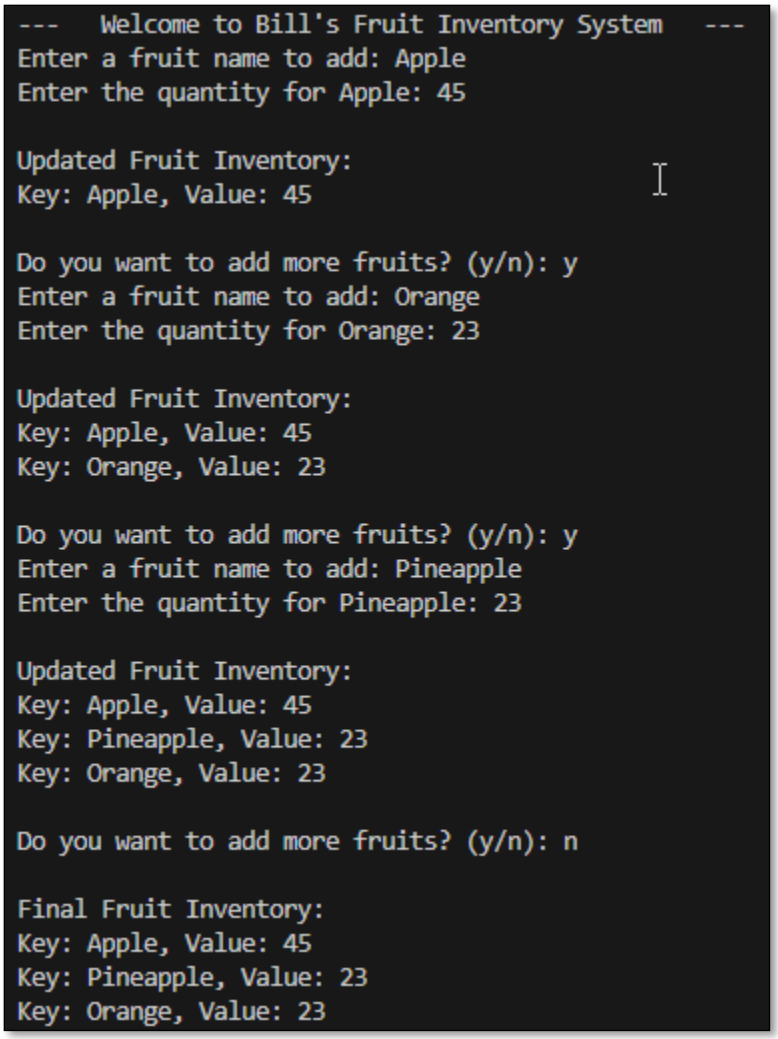
You will want to use an input loop to get the keys and values from the user. This is not an entire program, it is an example of an input loop to fill up your hashmap.

```
Scanner scanner = new Scanner(System.in);
// Characters always use a single quote '
char addMore = 'y';

     System.out.println("Welcome to Bill's Fruit Inventory System");
     while (addMore == 'y') {
        // Add new fruit and quantity based on user input
        System.out.print("Enter a fruit name to add: ");
        String newFruit = scanner.next();
        System.out.print("Enter the quantity for " + newFruit + ": ");
        int newQuantity = scanner.nextInt();
        myMap.put(newFruit, newQuantity);

        // TODO: Loop through each key in the HashMap to display to the user

        // Ask if the user wants to add more fruits
        System.out.print("Do you want to add more fruits? (y/n): ");
        // Only get the first character
        addMore = scanner.next().charAt(0);
        // Convert character to lowercase for easy comparison in the loop
        addMore = Character.toLowerCase(addMore);
    }
```

Assignment 1: Fruit Inventory

1. Make a copy of the FruitHashMap program. Name it **FruitInventory.java**
2. Change the method of element entry from hard coded to user entry.
3. Use a loop to continue adding elements.
4. Allow user to exit loop when they are done.
5. Print inventory for each loop, final inventory when they are done.

Example run:



Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.