Lab 9

Sets, Maps, Hashes

1. Sets (10 points, about 1 point per question)   
Thanks to <https://www.tutorialspoint.com/java/java_set_interface.htm> .

Please enter the following code into a Java file. Remember to type instead of copy-and-paste; Word loves to mangle source code.

// Student name today’s date

import java.util.\*;

public class SetDemo {

public static void main(String args[]) {

int count[] = {34, 22,10,60,30,22};

Set<Integer> unsortedSet = new HashSet<Integer>();

try {

for(int i = 0; i < 5; i++) {

unsortedSet.add(count[i]);

}

System.out.println("The unsorted set is: " + unsortedSet);

TreeSet sortedSet = new TreeSet<Integer>(unsortedSet);

System.out.println("The sorted set is: " + sortedSet);

System.out.println("The First element of the set is: "+ (Integer)sortedSet.first());

System.out.println("The last element of the set is: "+ (Integer)sortedSet.last());

}

catch(Exception e) {}

}

}

Run the program to confirm it works as expected. Then, expand the program to do the following steps in order. **For each test, display the result of the test to the console.**

1.1. Test to see if the unsorted set contains the value 10.

1.2. Test to see if the sorted set contains the value 5.

1.3. Remove the element 10 from the unsorted set.

1.4. Test to see if the unsorted set still contains the value 10.

1.5. Add the value **8** to the sorted set.

1.6. Test to see if the sorted test contains the value **8**.

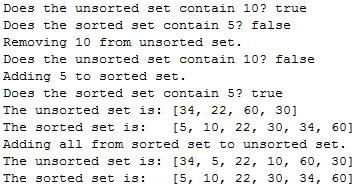
1.7. Display the unsorted set.

1.8. Display the sorted set. (The sorted set should now have two more values than the unsorted set.)

1.9. Add all values from the sorted set to the unsorted set by calling unsortedSet.addAll(sortedSet).

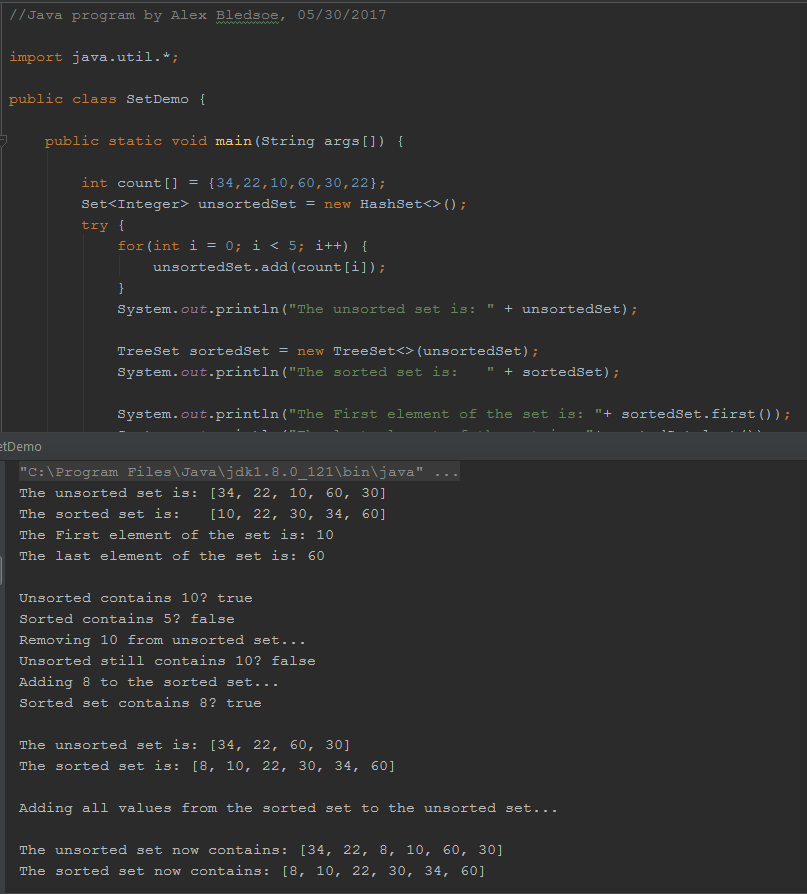
1.10. Display both sets and confirm they have the same contents, just in a different order.

Sample program output follows.  
**Your code must use the values listed above, even if they differ from the sample output.**



Rubric:  
Student name and today’s date is a comment in the first line of the programs: -5 points if fails  
Screenshot and program code: -5 points if fails  
Program uses values from the problem statement, **not** the sample output: -10 points if fails  
Ten test/steps: 1 point each

Please paste a screenshot of a successful program run, and copy-and-paste the source code from your .java file, here.



**SetDemo.java:**

//Java program by Alex Bledsoe, 05/30/2017  
  
import java.util.\*;  
  
public class SetDemo {  
  
 public static void main(String args[]) {  
  
 int count[] = {34,22,10,60,30,22};  
 Set<Integer> unsortedSet = new HashSet<>();  
 try {  
 for(int i = 0; i < 5; i++) {  
 unsortedSet.add(count[i]);  
 }  
 System.*out*.println("The unsorted set is: " + unsortedSet);  
  
 TreeSet sortedSet = new TreeSet<>(unsortedSet);  
 System.*out*.println("The sorted set is: " + sortedSet);  
  
 System.*out*.println("The First element of the set is: "+ sortedSet.first());  
 System.*out*.println("The last element of the set is: "+ sortedSet.last());  
  
 //1.1:  
 System.*out*.println("\nUnsorted contains 10? " + unsortedSet.contains(10));  
 //1.2:  
 System.*out*.println("Sorted contains 5? " + sortedSet.contains(5));  
 //1.3:  
 System.*out*.println("Removing 10 from unsorted set...");  
 unsortedSet.remove(10);  
 //1.4:  
 System.*out*.println("Unsorted still contains 10? " + unsortedSet.contains(10));  
 //1.5:  
 System.*out*.println("Adding 8 to the sorted set...");  
 sortedSet.add(8);  
 //1.6:  
 System.*out*.println("Sorted set contains 8? " + sortedSet.contains(8));  
 //1.7:  
 System.*out*.println("\nThe unsorted set is: " + unsortedSet);  
 //1.8:  
 System.*out*.println("The sorted set is: "+ sortedSet);  
 //1.9:  
 System.*out*.println("\nAdding all values from the sorted set to the unsorted set...");  
 unsortedSet.addAll(sortedSet);  
 //1.10:  
 System.*out*.println("\nThe unsorted set now contains: " + unsortedSet);  
 System.*out*.println("The sorted set now contains: " + sortedSet);  
  
  
 }  
 catch(Exception e) {}  
 }  
}

2. Maps (10 points)

Based on Reges self-check problem 11.19.

Please run the following code in Java, and then answer the following questions. Remember to type instead of copy-and-paste; Word loves to mangle source code.

// Student name today’s date

**import** java.util.\*;

**public** **class** MapTest {

**public** **static** **void** main(String args[]) {

Map<Integer, String> map = **new** HashMap<Integer, String>();

map.put(7, "Marty");

map.put(34, "Louann");

map.put(27, "Donald");

map.put(15, "Moshe");

map.put(84, "Larry");

map.put(7, "Ed");

map.put(2350, "Orlando");

map.remove(8);

map.put(5, "Moshe");

map.remove(84);

map.put(17, "Steve");

System.***out***.println(map);

}

}

2.1. What keys are in this map after the program runs?

**17, 34, 5, 7, 27, 2350, & 15**

2.2. What values are contained in the map after the program runs?

**Steve, Louann, Moshe (x2), Ed, Donald & Orlando**

2.3. There are two “put” commands with key 7. Why is there only one value for key 7 stored in the map?

**The second time “put” is used, the value store overwrites the original value with the new one.**

2.4. Are the values in a HashMap stored in sorted order?

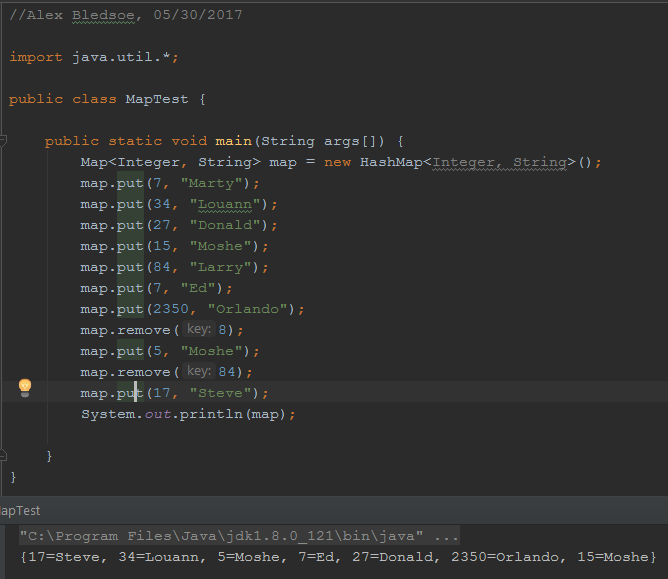
**No.**

2.5. When you issue a “put” command with a key that is already in the HashMap, Java returns the old value. Do you have to do anything with this returned value?  
(Hint: is the return value from map.put(7, "Ed"); assigned or output in the above program?)

**The original value is not output to the console when a new value is assigned to that key so no, you don’t have to do anything with it.**

Rubric:  
Student name and today’s date is a comment in the first line of the programs: -5 points if fails  
Screenshot and program code: -5 points if fails  
Five questions: 2 point each

Please paste a screenshot of a successful program run, and copy-and-paste the source code from your .java file, here.



**MapTest.java:**

//Alex Bledsoe, 05/30/2017  
  
import java.util.\*;  
  
public class MapTest {  
  
 public static void main(String args[]) {  
 Map<Integer, String> map = new HashMap<Integer, String>();  
 map.put(7, "Marty");  
 map.put(34, "Louann");  
 map.put(27, "Donald");  
 map.put(15, "Moshe");  
 map.put(84, "Larry");  
 map.put(7, "Ed");  
 map.put(2350, "Orlando");  
 map.remove(8);  
 map.put(5, "Moshe");  
 map.remove(84);  
 map.put(17, "Steve");  
 System.*out*.println(map);  
  
 }  
}

3. Iterators (5 points)

3.1. In your own words, describe what each iterator method does:

hasNext(): Returns true if . . .

**There is another element in the collection.**

next():

**Returns the next following element from the collection and throws an exception if there’s nothing left to return.**

remove ():

**Removes the element previously returned by next() and throws an exception if you haven’t called next() yet.**

3.2. If you have a Collection variable list, containing strings, how would you use a for loop to display the contents of list? Hint: for (String word: list) { // your code here }

for (String word: list){  
 System.*out*.println(list.get(list.indexOf(word)));}

3.3. Since an ArrayList, Linked List, HashSet, and TreeSet all implement the Collection interface, the methods in this section work for variables of any of those data types. Which of the following does that demonstrate?

Inheritance Subnetting Encapsulation

4. Hash basics. (10 points)

Hash a list of employee name/number combinations.

Hashing function h1(x) = the **last** three digits of the employee number

Compression function h2(x) = the value mod 10  
 (Hint: this is the last digit of the value coming in.)

Index into the array: h2(h1(x))

|  |  |  |
| --- | --- | --- |
| Key and value | Hash code  h1(x)  last three digits | Array index  h2(h1(x))  h1(x) % 10 |
| Haen324350 | 350 | 0 |
| Gurj313235 | 235 | 5 |
| Durd313553 | 553 | 3 |
| Puti125432 | 432 | 2 |
| Grog131354 | 354 | 4 |
| Rami967999 | 999 | 9 |

When complete, please put the values into the array according to the hash index

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Value | Haen324350 |  | Puti125432 | Durd313553 | Grog131354 | Gurj313235 |  |  |  | Rami967999 |

Rubric:

H1(x): 4 points

H2(h1(x)): 4 points

Array values: 2 points

5. Hash collisions (10 points)

Hash a list of employee name/number combinations.

Hashing function h1(x) = the **first** three digits of the employee number

Scaling function h2(x) = the value mod 10   
 The easiest way to calculate this: it’s the last digit of the value coming in

Index into the array: h2(h1(x))

|  |  |  |
| --- | --- | --- |
| Key and value | Hash code  h1(x)  first three digits | Array index  h2(h1(x))  h1(x) % 10 |
| Haen324350 | 324 | 4 |
| Gurj313235 | 313 | 3 |
| Durd313553 | 313 | ~~3~~ -> ~~4~~ -> 5 |
| Puti125432 | 125 | ~~5~~ -> 6 |
| Grog131354 | 131 | 1 |
| Rami967999 | 967 | 7 |

When complete, please put the values into the array according to the hash index.   
In case of collisions, use **linear probing**.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Value |  | Grog131354 |  | Gurj313235 | Haen324350 | Durd313553 | Puti125432 | Rami967999 |  |  |

Rubric:

H1(x): 3 points

H2(h1(x)): 3 points

Array values: 4 points