

/\* Alex Porter / Havin Lim

Lab: 11 - Polymorphism and Interfaces

September 19th 2022

Sources : Hash Codes

<https://www.baeldung.com/java-hashcode>

Help obtained : Peng, Yuxin (Kevin)

We confirm that the above list of sources is complete AND that we have not talked to anyone else (e.g., CSC207 students) about the solution to this problem

\*/

<Counter.java>

```
package tallying;
```

```
class Counter {
```

```
    private String name = "";
```

```
    private int count = 0;
```

```
    public Counter(String name, int count) {
```

```
        this.name = name;
```

```
        this.count = count;
```

```
    }
```

```
    public Counter(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    public int getCount() {
```

```
        return this.count;
```

```
    }
```

```
    public void setCount(int n) {
```

```
        this.count = n;
```

```
    }
```

```
    public String getName () {
```

```
        return this.name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```

    }

    public void increment() {
        this.count++;
    }

    public void reset() {
        this.count = 0;
    }

    public String toString() {
        return (name + ": " + count);
    }
}

```

#### **<CounterWithMemory.java>**

```

package tallying;
import java.nio.BufferUnderflowException;

public abstract class CounterWithMemory extends Counter {
    private int memory = 0;
    public CounterWithMemory(String name, int count) {
        super(name, count);
    }
    public CounterWithMemory(String name) {
        super(name);
    }

    abstract String show();

    public void storeTally () {
        if (this.memory != 0) {
            System.out.println("Overwritten value: " + this.memory);
        }
        this.memory = this.getCount();
    }
    public void recoverTally () {
        try {
            this.setCount(this.memory);
        }
    }
}

```

```

        catch (BufferUnderflowException e) {
            System.err.println(e);
        }
    }
    protected int getMemory () {
        return this.memory;
    }
    public void resetMemory () {
        this.memory = 0;
    }
}

```

#### <BasicCounterWithMemory.java>

```

package tallying;

public class BasicCounterWithMemory extends CounterWithMemory {
    public BasicCounterWithMemory (String name) {
        super(name);
    }
    String show () {
        int count = this.getCount();
        Integer last = (Integer) count;
        return last.toString();
    }
}

```

#### <LoggingCounterWithMemory>

```

package tallying;
import java.util.Date;

public class LoggingCounterWithMemory extends CounterWithMemory {
    public LoggingCounterWithMemory (String name) {
        super(name);
    }
    String show () {
        int count = this.getCount();
        Integer last = (Integer) count;
    }
}

```

```

        Date time = new Date();
        return this.getName() + " " + last.toString() + " " +
time.toString();
    }
}

```

#### <LoggingCounterWithMemoryTest.java>

```

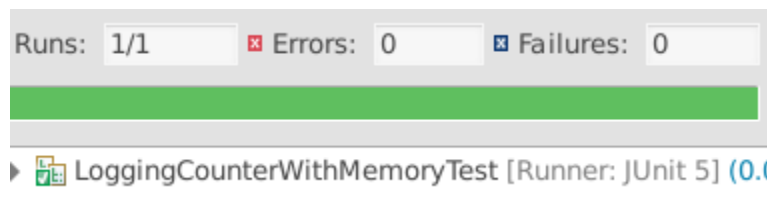
package tallying;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

class LoggingCounterWithMemoryTest {
    @Test
    void testLoggingCounterWithMemory() {
        LoggingCounterWithMemory test = new
LoggingCounterWithMemory("test");
        System.out.println(test.show());
    }
}

```



#### <BasicCounterWithMemoryTest.java>

```

package tallying;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

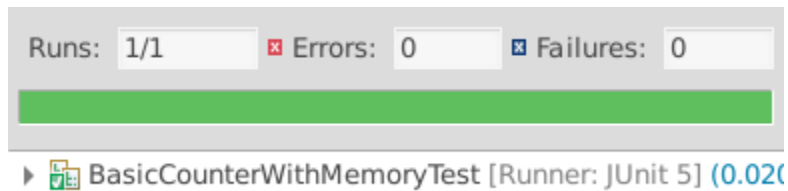
class BasicCounterWithMemoryTest {

```

```

    @Test
    void testBasicCounterWithMemory() {
        BasicCounterWithMemory counter = new
BasicCounterWithMemory("test");
        assertEquals(counter.show(), "0");
    }
}

```



#### <Responder.java>

```

package conversation;

/*
 * The Responder interface can hold up one end of a conversation. The String
 returned by the sole method is a reply to the String given.
 */
public interface Responder {
    public abstract String respond(String str);
}

```

#### <Bore.java>

```

package conversation;

public class Bore implements Responder {
    private String response;
    public String respond(String str) {
        return this.response;
    }
}

```

```

    }
    public Bore (String reply) {
        this.response = reply;
    }
}

```

#### <BoreTest.java>

```

package conversation;

import static org.junit.jupiter.api.Assertions.*;

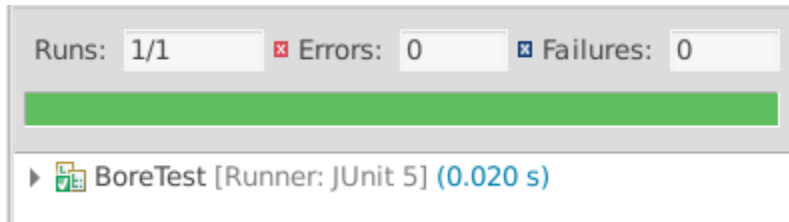
import org.junit.jupiter.api.Test;

class BoreTest {

    @Test
    void test() {
        Bore b = new Bore("bore");
        assertEquals(b.respond("hello"), "bore");
    }

}

```



#### <Numberer.java>

```

package conversation;

public class Numberer implements Responder {
    private int count;
    private Responder numerand;

    public String respond(String str) {
        String response = numerand.respond(str);
        this.count += 1;
        return String.format("%d: %s", this.count, response);
    }
}

```

```

    }

    public Numberer(Responder style) {
        this.numerand = style;
        this.count = 0;
    }
}

```

#### <NumbererTest.java>

```

package conversation;

import static org.junit.jupiter.api.Assertions.*;

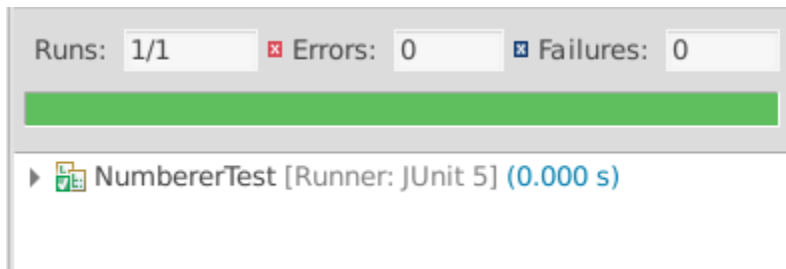
import org.junit.jupiter.api.Test;

class NumbererTest {

    @Test
    void test() {
        Bore b = new Bore("bore");
        Numberer test = new Numberer(b);
        assertEquals(test.respond("hello"), "1: bore");
        assertEquals(test.respond("still there?"), "2: bore");
    }

}

```



```
/* Alex Porter / Havin Lim
Lab: 12 - Generics
September 21th 2022
Sources : None
Help obtained : Peng, Yuxin (Kevin)
We confirm that the above list of sources is complete AND that we have not
talked to anyone else (e.g., CSC207 students) about the solution to this
problem
*/
```

### <KeyValuePair.java>

```
package kvpairs;

public class KeyValuePair <K, V> {

    private K key;
    private V value;

    public KeyValuePair (K key, V value) {
        this.key = key;
        this.value = value;
    }
    public K getKey() {
        return this.key;
    }
    public V getValue() {
        return this.value;
    }

    @Override
    public int hashCode() {
        return value.hashCode() + key.hashCode();
    }

    @Override
    public boolean equals(Object e) {
        if (e instanceof KeyValuePair) {
            if (this.hashCode() == e.hashCode()) {
                return true;
            }
        }
    }
}
```



```

        }
        else {
            return false;
        }
    }
    else {
        return false;
    }
}

@Override
public String toString() {
    return "Key: " + this.key + " Value: " + this.value;
}
}

```

#### <ClientProgram.java>

```

package kvpairs;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;

public class ClientProgram {

    public static <K,V> void printVals (List l) {
        Iterator<KeyValuePair<K,V>> traverser = l.iterator();
        while (traverser.hasNext()) {
            KeyValuePair<K,V> element = traverser.next();
            System.out.println("Key: " + element.getKey() + " Value: " +
element.getValue());
        }
    }

    public static <T> void printVals2 (List<T> l) {
        for(T element : l) {
            System.out.println(element);
        }
    }
}

```

```

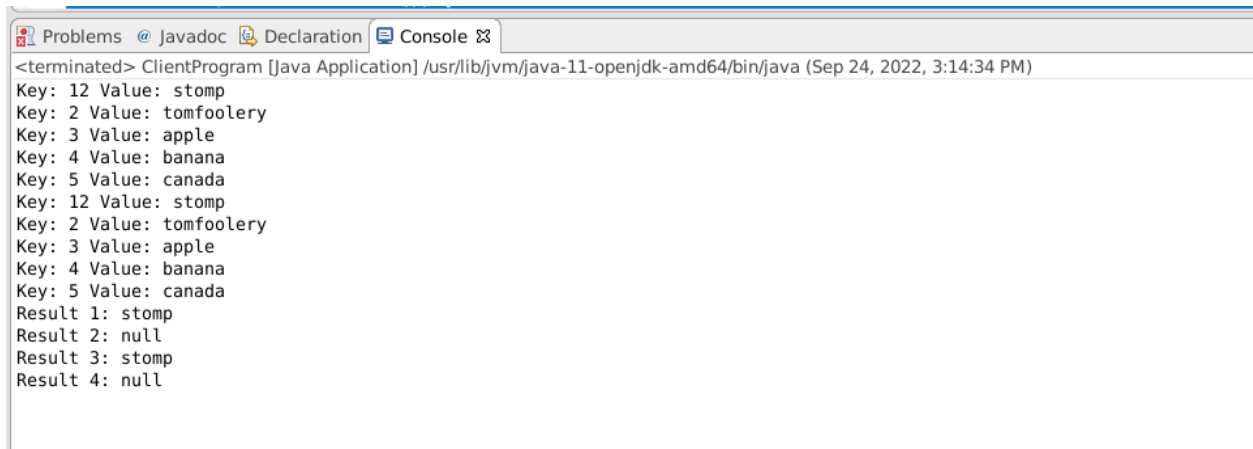
    }
}

public static <K,V> V findElement (List l, K key) {
    Iterator<KeyValuePair<K,V>> traverser = l.iterator();
    while (traverser.hasNext()) {
        KeyValuePair<K,V> element = traverser.next();
        if (element.getKey() == key) {
            return element.getValue();
        }
    }
    return null;
}

public static void main(String[] args) {
    KeyValuePair<Integer, String> k = new KeyValuePair<>(12, "stomp");
    KeyValuePair<Integer, String> v = new KeyValuePair<>(2,
"tomfoolery");
    KeyValuePair<Integer, String> y = new KeyValuePair<>(3, "apple");
    KeyValuePair<Integer, String> w = new KeyValuePair<>(4, "banana");
    KeyValuePair<Integer, String> x = new KeyValuePair<>(5, "canada");
    //System.out.println(k.toString());
    //System.out.println(k.hashCode());
    //System.out.println(v.equals(y));
    ArrayList<KeyValuePair<Integer, String>> jank = new ArrayList<>();
    jank.add(k);
    jank.add(v);
    jank.add(y);
    jank.add(w);
    jank.add(x);
    LinkedList<KeyValuePair<Integer, String>> tester = new
LinkedList<>();
    tester.add(k);
    tester.add(v);
    tester.add(y);
    tester.add(w);
    tester.add(x);
    //printVals(jank);
    //printVals(tester);
    printVals2(jank);
    printVals2(tester);
    System.out.println("Result 1: " + findElement(jank, 12));
}

```

```
        System.out.println("Result 2: " + findElement(jank, 9));  
        System.out.println("Result 3: " + findElement(tester, 12));  
        System.out.println("Result 4: " + findElement(tester, 9));  
    }  
}
```



The screenshot shows an IDE console window with the following output:

```
<terminated> ClientProgram [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Sep 24, 2022, 3:14:34 PM)  
Key: 12 Value: stomp  
Key: 2 Value: tomfoolery  
Key: 3 Value: apple  
Key: 4 Value: banana  
Key: 5 Value: canada  
Key: 12 Value: stomp  
Key: 2 Value: tomfoolery  
Key: 3 Value: apple  
Key: 4 Value: banana  
Key: 5 Value: canada  
Result 1: stomp  
Result 2: null  
Result 3: stomp  
Result 4: null
```

```
/* Alex Porter / Havin Lim
Lab: 13 - The Git Version Control System.
September 23th 2022
Sources : None
Help obtained : Peng, Yuxin (Kevin)
We confirm that the above list of sources is complete AND that we have not
talked to anyone else (e.g., CSC207 students) about the solution to this
problem
*/
```

Merging:

```
porteral@mauchly:lab13$ git log
```

```
commit 0b0b1edb667b9aaffcb0569ea507da7d5fb4032a (HEAD -> master,
experimental)
```

```
Author: Porter <porteral@wijnngaarden.cs.grinnell.edu>
```

```
Date:   Sat Sep 24 14:51:21 2022 -0500
```

Added peekAtContents

```
commit 4bfc4bbe9e2f155c832c76377fca126ca4362dc
```

```
Author: Porter <porteral@wijnngaarden.cs.grinnell.edu>
```

```
Date:   Sat Sep 24 14:06:49 2022 -0500
```

Added replaceContents

```
commit c8d02de3c7c8f4dca67fa4967c5ec6697b543567
```

Author: Porter <porteral@strachey.cs.grinnell.edu>

Date: Fri Sep 23 15:07:04 2022 -0500

No changes yet. First commit.

-----

Deleting a branch:

porteral@mauchly:lab13\$ git log

commit 0b0b1edb667b9aaffcb0569ea507da7d5fb4032a (HEAD -> master)

Author: Porter <porteral@wijngaarden.cs.grinnell.edu>

Date: Sat Sep 24 14:51:21 2022 -0500

Added peekAtContents

commit 4bfcb4bbe9e2f155c832c76377fca126ca4362dc

Author: Porter <porteral@wijngaarden.cs.grinnell.edu>

Date: Sat Sep 24 14:06:49 2022 -0500

Added replaceContents

commit c8d02de3c7c8f4dca67fa4967c5ec6697b543567

Author: Porter <porteral@strachey.cs.grinnell.edu>

Date: Fri Sep 23 15:07:04 2022 -0500

No changes yet. First commit.