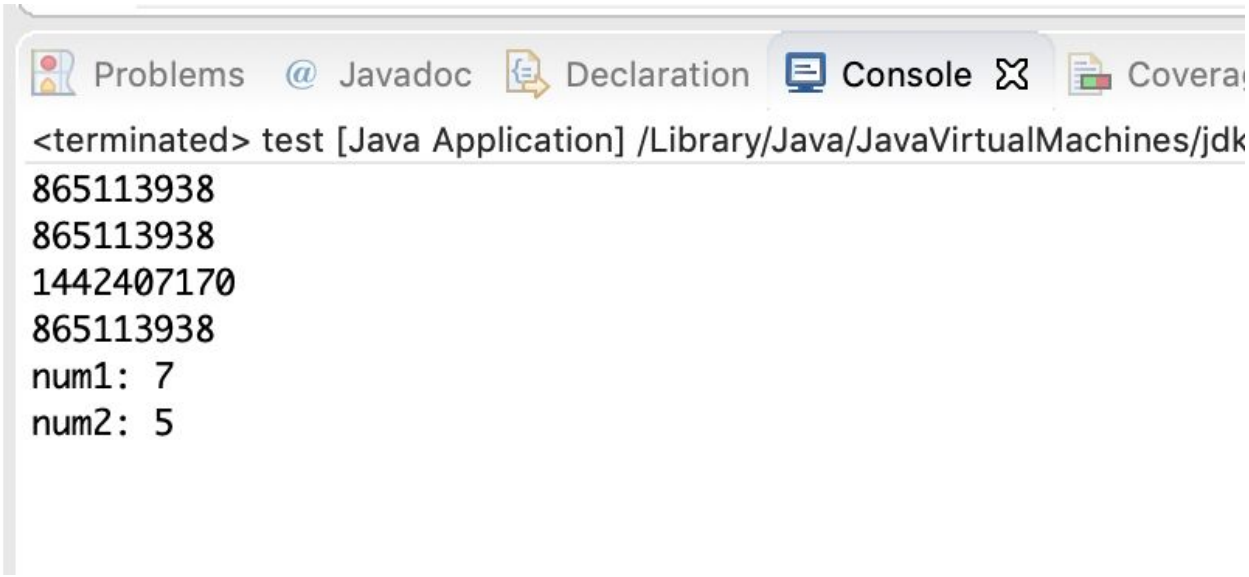
Group 7: Ling Xiang, Alex Kwon, Weiqi Hu

1. Results of test: When num2 = num1, num2 is pointing to the same reference of num1. Therefore, when printing out the identityHashCode, we get the same answer. Since integer is immutable, when num1 is assigned to a new integer, it references to a new modified copy with the value is 7, appends num1 to its reference. As a result, the identityHashCode changes and the value of the num1 changes, but the num2 still references the original memory, making no changes in its result.
2. Results of Mypoint class: .Mypoint class is immutable, but the instance variables (x and y) are int which is mutable. Therefore, the reference will not change, but the values inside the class will change.

<Screenshot of the output of test.java>



<Screenshot of the modified Mypoint.java>

```java
public class Mypoint {
 private final Integer x;
 private final Integer y;

 public Mypoint(Integer x, Integer y) {
 this.x = x;
 this.y = y;

 }
 public static void main(String []arguments){
 Mypoint pt1, pt2;
 pt1 = new Mypoint(1
 pt2 = pt1;

 System.out.println(
 System.out.println(System.identityHashCode(pt2));

 pt1 = new Mypoint(200,200);
//pt1.x = 200;
//pt1.y = 200;

 System.out.println(System.identityHashCode(pt1));
 System.out.println(System.identityHashCode(pt2));
 System.out.println("Point1: " + pt1.x + ", " + pt1.y);
 System.out.println("Point2: " + pt2.x + ", " + pt2.y);

 }
}
```

void Mypoint.main(String[] arguments)

Press 'F2' for focus

Problems @ Javadoc Declaration Console

<terminated> Mypoint [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_

```
865113938
865113938
1442407170
865113938
Point1: 200, 200
Point2: 100, 100
```