TESTING DOCUMENT

GAME NAME: HALCYON

CPSC 233 T03-3

Caelum Ethen Sloan, Matt Sardhina, Stefan Gavriliuc, Benjamin MacKay, Jiexin (Jessica) Liu

PART I: TESTING MANUALLY THROUGH INTERFACE

Testing: Basic Movement & Collision

1. Hold "**W**". Hero should move **up** and **stop at the wall.**
2. Hold "**A**". Hero should move **left** and **stop at the wall.**
3. Hold "**S**". Hero should move **down** and **stop at the wall.**
4. Hold "**D**". Hero should move **right** and **stop at the wall.**

Movement Combos:
5. Hold **"A" and "W"** together. Hero should move **diagonally up-right** and **stop at the wall.**
6. Hold "**A**" and **THEN press "D**". Hero should **change direction**, from moving left to now moving right (however, **the opposite order should NOT be true** and hero should continue moving in the original direction).
7. Hold "**W**" and **THEN press "S".** Hero should **change direction**, from moving up to now moving down (however, **opposite order should NOT be true** and hero should continue moving in the original direction).

Movement speed:
8. Hold "**Shift**" and then try any combination of movements. Hero should **move faster** than before (running)!

Testing: Object ranges and player hitbox

1. Press "**H**". Should see a **small red rectangle** displayed on player (hitbox). Try moving into the walls. Hero **should stop when the corners of the hitbox contact the walls**.
2. Press "**R**". Should see **red circles** around each item, indicating the range at which items can be interacted with.
   a. Move hero **outside** the sword's range. Press "**Enter**". **Nothing should happen.**
   b. Move hero **inside** the sword's range. Press "**Enter**". Console should **print "That tickles! 0w0"**

Testing: Text-based version

1. Press "**P**". Console will **print the hero's current position** in tile units (row and column).
   a. **Hold "P"** and try **moving**. The hero's position should **continuously update and print** to the console to match current position.
2. Press "**T**". Console will **display the hero's current position** in the text-based tile grid.
   a. **Hold "T"** and try **moving**. The hero's char should **move accordingly** and print to console in the text grid.

Testing: Switching Rooms

1. Move hero to **interact with the lever.** A **black door should open**. Step inside the door and move up. Hero should **enter a new room** from the bottom.
   a. Explore the room! When ready, **re-enter the black door** from the second room and hero should **return to original room.**

Testing: Saving

1. **Run** the game. **Close** the game. Open **checkPoint.txt**. It should display the integer **"0"** in the first line, indicating that the last room the hero was in, had index 0 (spawn location).
2. **Run** the game. Move to a **new room**. **Close** the game. **Open checkPoint.txt**. It should now display a **non-zero integer** in the first line, depending on which room you entered. **Run the game again**. Hero should be in the **most recent room** he visited prior to exiting.

PART II: TESTING AUTOMATICALLY

USING JUNIT TESTS (Version should be JUnit 4, same as coding challenges)
Note: We were unable to figure out how to run the JUnits from the command line. However, running from any version of Eclipse that contains the JUnit 4 path should work.
Open the project in Eclipse.
Navigate to PositionTest (project > administrative > administrative.JUnits > PositionTest)

Click "Run PositionTest"
All 5 tests should run and Eclipse should output the following screen:

Package | JUnit ⊠

Finished after 2.144 seconds

Runs: 5/ Errors: | Failures:

▶ administrative.JUnits.PositionTest

Failure Trace

PositionTest.java ⊠

```java
1   package administrative.JUnits;
2
3   import static org.junit.Assert.*;
11
12  public class PositionTest {
13
14      @Test
15      public void test_constructors_never_negative() {
16          Position p = new Position(-45, -2, -1);
17          assertEquals(0, p.getRoomIndex());
18          assertEquals(0, p.getPosX());
19          assertEquals(0, p.getPosY());
20      }
21
22      @Test
23      public void testConstructor_all_zeros() {
24          Position p = new Position(0,0,0);
25          assertEquals(0,p.getRoomIndex());
26          assertEquals(0,p.getPosX());
27          assertEquals(0,p.getPosY());
28      }
29
30      @Test
31      public void test_copy_constructor() {
32          Position p = new Position(1,2,3);
33          Position p_copy = new Position(p);
34          assertEquals(1, p_copy.getRoomIndex());
35          assertEquals(2, p_copy.getPosX());
36          assertEquals(3, p_copy.getPosY());
37
38          p_copy.setPosX(10);
39          assertEquals(10, p_copy.getPosX());
40
41      }
42
43      @Test
44      public void test_setters_are_always_positive_or_0() {
45          Position p = new Position(1, 13, 2);
```

Problems @ Javadoc Declaration Console ⊠

<terminated> PositionTest [JUnit] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (Apr 13,