

Question 1. [14 MARKS]

1. `double newFish = (double) (blueFish - redFish) / 2;`
2. `(health + happiness) % 7 == 0`
3. `return Math.max(currentSpeed + speedReduction, minSpeed);`
4.

```
public void setWorth(String a) {
    this.worth = Double.parseDouble(a);
}
```
5.

```
public class Movie {
    private static int totalMovies;
    private double admissionAmount;

    public static double getTotalMovies() {
        return totalMovies;
    }
}
```
6.

```
public Movie() {
    this.admissionAmount = 12.00;
    totalMovies++;
}

public Movie(Double amt) {
    this.admissionAmount = amt;
    totalMovies++;
}
```
7.

```
Movie m1 = new Movie();
Movie m2 = new Movie(6.00);
totalAdmitted = Movie.getTotalMovies();
```

Question 2. [10 MARKS]

```
public class AVTracked extends AVEquipment{

    private int roomNum;
    private static boolean changedRoomsYet;

    public void setRoomNum (int num) {
        roomNum = num;
        changedRoomsYet = true;
    }

    public static boolean hasChanged() {
        return changedRoomsYet;
    }

    public String roomNumMessage() {
        return "The room number is " + roomNum;
    }
}
```

Question 3. [13 MARKS]

import junit.framework.TestCase; public class InstructorCourseTester extends TestCase {	Result (P/F)	If failed, value returned by second argument?
public void testInstConstructorGetName() { Instructor i = new Instructor("Jen", "Campbell"); assertEquals("Jen Campbell", i.getName()); }	_P__	-----
public void testInstToString() { Instructor i = new Instructor("Michael", "Szamosi"); assertEquals("Michael Szamosi", i.toString()); }	_F__	nullMichaelSzamosi
public void testInstGetNumInstructors() { Instructor i1 = new Instructor("Faye", "Baron"); int numInstructors = i1.getNumInstructors(); Instructor i2 = new Instructor("Michael", "Szamosi"); Instructor i3 = new Instructor("Jen", "Campbell"); assertEquals(numInstructors+2, i3.getNumInstructors()); }	_F__	1
public void testCourseConstructor() { Instructor i = new Instructor("Michael", "Szamosi"); Course c = new Course("CSC207", i); assertEquals("CSC207", c.getCourseName()); }	_P__	-----
public void testCourseGetInstructor() { Course c1 = new Course("CSC108", new Instructor("Faye", "Baron")); Course c2 = new Course("CSC340", new Instructor("Faye", "Baron")); assertEquals(c1.getInstructor(), c2.getInstructor()); }	_F__	some memory address
public void testCourseInstructorChanges() { Course c = new Course("CSC108", new Instructor("Jen", "Campbell")); c.setInstructor(new Instructor("Faye", "Baron")); c.setInstructor(new Instructor("Michael", "Szamosi")); assertEquals(2, c.getInstructorChanges()); }	_P__	-----
public void testSwapInstructors() { Course c1 = new Course("CSC108", new Instructor("Faye", "Baron")); Course c2 = new Course("CSC340", new Instructor("Michael", "Szamosi")); Course.swapInstructors(c1, c2); assertEquals("Faye Baron", c2.getInstructor().getName()); }	_F__	Michael Szamosi

```
public void testCourseToString() {
    Course c = new Course("CSC108", new Instructor("Jen", "Campbell"));
    assertEquals("Course: CSC108 Instructor: Jen Campbell",
        c.toString());
}
}
```

F Course: courseName
Instructor: Jen Campbell

Total Marks = 37

Student #: _____

Page 4 of 4

END OF SOLUTIONS