

CSC 108H1 S February Midterm 2005

Duration — 50 minutes

Aids allowed: none

Student Number: _____

Lab day, time, room: _____

Last Name: _____

First Name: _____

CDF login: _____

Lecture: L0101, Jennifer Campbell

*Do **not** turn this page until you have received the signal to start.*

*(Please fill out the identification section above,
and read the instructions below.) Good Luck!*

This midterm consists of 3 questions on 6 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code.

For 1 bonus mark write your student number at the bottom of pages 2-6 of this test. Both sides.

If you use any space for rough work, indicate clearly what you want marked.

BONUS

MARK: _____/ 1

1: _____/14

2: _____/10

3: _____/13

TOTAL: _____/37

Short Java API descriptions (all methods are public):

```
class Integer:
```

```
    Integer(int i) // An Integer with value i  
    static int parseInt(String s) // = s's value, as an int.
```

```
class Double:
```

```
    Double(double d) // A Double with value d  
    static double parseDouble(String s) // = s's value, as a double.
```

```
class String:
```

```
    String substring(int i, int j) // = the letters between i (inclusive) and j (non-inclusive).  
    String substring(int i) // = the letters from i (inclusive) to the end.  
    int indexOf(String s) // = the index of s in this String; -1 if s is not a substring.  
    int indexOf(String s, int i) // = index of s in this String after index i; -1 if s not found.  
    int length() // = the number of characters in this String.
```

```
class Math
```

```
    static int min(int i, int j) // = the smaller of i and j  
    static int max(int i, int j) // = the larger of i and j
```

Question 1. [14 MARKS]

1. Given two `int` variables `redFish` and `blueFish`, write a statement that declares a `double` variable `newFish` and assigns to it: `redFish` subtracted from `blueFish`, then divided by 2 using floating point arithmetic. [2 marks]
2. Given two `int` variables `happiness` and `health`, write a statement that evaluates to `true` if their sum can be evenly divided by 7, and `false` otherwise. [1 mark]
3. Given three `int` variables: `minSpeed`, `currentSpeed`, and `speedReduction`. Without using `if`-statements or `?:`, write a statement that returns the sum of `currentSpeed` and `speedReduction` only if the sum is greater than or equal to `minSpeed`, otherwise it returns `minSpeed`. [2 marks]
4. An existing class `netWorth` has a `double` instance variable called `worth`. Write a `void` method `setWorth`, which has a `String` parameter . The method converts the given `String` to a type that is compatible with `worth` and assigns it to `worth`. [2 marks]
5. Write a class `Movie`, which has two instance variables: a `static int` variable `totalMovies`, and a `double` variable `admissionAmount`. It also has a `static int` method `getTotalMovies` that has no input parameters and returns the total number of movies. [3 marks]
6. Write two constructors for the class `Movie` (from the previous question). The first constructor has no input parameters and the admission amount is 12.00. The second constructor has a `double` input parameter, which is the admission amount. The total number of movies should be incremented by one for every movie created. [2.5 marks]
7. Write three statements, using the information in questions 5 and 6: The first statement declares a `Movie` variable `m1`, to which it assigns a newly created `Movie`. The second statement declares a `Movie` variable `m2`, to which it assigns a newly created `Movie` with an admission amount of 6.00. The third statement gets the total number of movies and assigns it to a previously declared `int` variable `totalMovies`. [1.5 marks]

Question 2. [10 MARKS]

In Assignment 1, you wrote a class `AVEquipment`. Write a customized `AVEquipment` class called `AVTracked` (`AVTracked` is a subclass of `AVEquipment`). `AVTracked` has two new variables:

- a `private int` variable called `roomNum`
- a `private static boolean` called `changedRoomsYet`

The class also has three new methods:

- a `public void` method called `setRoomNum`, which sets the new `int` variable to the given `int` parameter
- a `public static boolean` method called `hasChanged`, which returns the value of `changedRoomsYet`
- a `public String` method called `roomNumMessage`, which returns a `String`: “The room number is X” (where X is the room number).

The variable `changedRoomsYet` keeps track of whether **any** object of this customized class has had its `setRoomNum` method called.

Question 3. [13 MARKS]

Consider the classes `Instructor` and `Course`. There is a JUnit test class called `InstructorCourseTester` on the next page, which tests these classes. All code in this question compiles and runs without crashing. There are eight `test` methods and `assertEquals` calls. To the right of every `assertEquals` call, write **P** if the `assertEquals` passes, or **F** if the `assertEquals` fails. To the right of each failure write the value returned by the second argument to `assertEquals`.

```

public class Instructor {
    private String name;
    private String title;
    private int numInstructors;

    public Instructor(String fn, String ln) {
        name = fn + " " + ln;
        numInstructors++;
    }

    public String getName() {
        return name;
    }

    public int getNumInstructors() {
        return numInstructors;
    }

    public String toString() {
        return title + name;
    }
}

public class Course {
    private Instructor instructor;
    private String courseName;
    private static int instructorChanges;

    public Course(String n, Instructor i) {
        courseName = n;
        instructor = i;
    }

    public String getCourseName() {
        return courseName;
    }

    public void setInstructor(Instructor i) {
        instructor = i;
        instructorChanges++;
    }

    public Instructor getInstructor() {
        return instructor;
    }

    public int getInstructorChanges() {
        return instructorChanges;
    }

    public static void swapInstructors(Course c1,
                                       Course c2) {
        c1.instructor = c2.instructor;
        c2.instructor = c1.instructor;
    }

    public String toString() {
        return "Course: courseName" + " Instructor: "
            + instructor.getName();
    }
}

```

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()); }	----	-----
public void testInstToString() { Instructor i = new Instructor("Michael", "Szamosi"); assertEquals("Michael Szamosi", i.toString()); }	----	-----
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()); }	----	-----
public void testCourseConstructor() { Instructor i = new Instructor("Michael", "Szamosi"); Course c = new Course("CSC207", i); assertEquals("CSC207", c.getCourseName()); }	----	-----
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()); }	----	-----
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()); }	----	-----
public void testSwapInstructors() { Course c1 = new Course("CSC108", new Instructor("Faye", "Baron")); Course c2 = new Course("CSC340", new Instructor("Michael", "Szamozi")); Course.swapInstructors(c1, c2); assertEquals("Faye Baron", c2.getInstructor().getName()); }	----	-----
public void testCourseToString() { Course c = new Course("CSC108", new Instructor("Jen", "Campbell")); assertEquals("Course: CSC108 Instructor: Jen Campbell", c.toString()); }	----	-----
}		

Use this page for rough work.

Total Marks = 37

Student #: _____

Page 6 of 6

END OF EXAMINATION