

CSC 108H1 F October Midterm 2004

Duration — 50 minutes

Aids allowed: none

Student Number: \_\_\_\_\_

Lab day, time, room: \_\_\_\_\_

Last Name: \_\_\_\_\_

First Name: \_\_\_\_\_

CDF login: \_\_\_\_\_

Lecture: L0201, 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.*

# 1: \_\_\_\_\_/14

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.

# 2: \_\_\_\_\_/14

# 3: \_\_\_\_\_/10

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

TOTAL: \_\_\_\_\_/38

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

---

**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 JFrame:

```
JFrame() // An empty window with no title, not visible on the screen
JFrame(String s) // An empty window with title s, not visible on the screen
Container getContentPane() // this JFrame's content pane
int getWidth() // this JFrame's width
int getHeight() // this JFrame's height
int getX() // this JFrame's horizontal coordinate
int getY() // this JFrame's vertical coordinate
void setSize(int w, int h) // set this JFrame's size to w wide and h high
void setTitle(String s) // set this JFrame's title to s
void setLocation(int x, int y) // set this JFrame's location to (x, y)
void show() // make this JFrame visible
void hide() // make this JFrame invisible
```

class Date:

```
int getDate() // return the day of the month (1-31) of this Date
int getMonth() // return the month as a number (0-11) of this Date; 0 is January
```

class JOptionPane:

```
static String showInputDialog(String m) // get input from the user, prompting with m.
```

**Question 1.** [14 MARKS]

1. Given two `int` variables `yesterday` and `today`, write an expression that multiplies `yesterday` by `today` and has a floating point result.
2. Write an expression that evaluates to `true` if the variables `here` and `there` are referencing the same object, and evaluates to `false` otherwise.
3. Without using `if`-statements or `?:`, write a single statement that is equivalent to:

```
if (a && !b) {  
    return false;  
} else {  
    return true;  
}
```

4. Write a method `calculateCost`, which has two `double` parameters `price` and `tax` and returns their sum.
5. Write a class `CompactDisc`, which has a `boolean` instance variable `originalCopy` and a `static int` variable `numDiscs`. There are no methods.
6. Write a class `Novel`, which has an instance variable that is a reference to an `Author` object. It also has a constructor, which has one `Author` parameter. Assume class `Author` has already been defined.
7. Write a method called `getCorner`, which has one parameter, a `JFrame`, and returns a `String` containing the `JFrame`'s top left (X,Y) coordinate. Assume all necessary packages have been imported. For example, for a `JFrame` located at the top left corner of the screen, the result would be "(0,0)".

**Question 2.** [14 MARKS]

Write a customized `JFrame` class that contains a `public` method called `shift`, a `private static int` variable called `numShifts`, and a `public static` method called `getNumShifts` that returns the value of the variable `numShifts`.

The variable `numShifts` keeps track of the number of times that objects of this customized class have had their `shift` method called. The `shift` method moves the entire `JFrame` left by one quarter of the width of the `JFrame`, but should never move the left edge of the `JFrame` off the left of the screen; if that were to happen, the `JFrame` should instead be placed at the top left of the screen at location (0,0). (You might need to use an `if` statement.) Don't forget to import `JFrame` from package `javax.swing`.

**Question 3.** [10 MARKS]

There is a JUnit test class called `NewFmTester` on the next page. All code in this question compiles and runs without crashing. The `NewFmTester` class tests class `NewFM`, which is below. 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 NewFM {

    private String firstName = null;
    private String lastName = null;
    private NewFM spouse = null;

    public NewFM(String firstN, String lastN) {
        firstName = firstN;
        lastName = lastN;
    }

    public static void marry(NewFM fm1, NewFM fm2) {
        if (!fm1.lastName.equals(fm2.lastName)) {
            fm1.spouse = fm2;
            fm2.spouse = fm1;
        }
    }

    public void unmarry() { spouse = null; }

    /**
     * If the last names of this NewFM and the fiancée are not the same, create a new last name
     * for both of them by joining the last names together with a hyphen.
     */
    public void setLastNames(NewFM fiancée) {
        if (this.lastName != fiancée.lastName) {
            this.lastName += "-" + fiancée.lastName;
            fiancée.lastName = this.lastName;
        }
    }

    public void revertName() {
        int divider = lastName.indexOf("-");
        lastName = lastName.substring(0, divider);
    }

    public void swapNames() {
        this.lastName = this.firstName;
        this.firstName = this.lastName;
    }

    public String toString() { return firstName + " " + lastName; }
    public String getFirstName() { return firstName; }
    public String getLastName() { return lastName; }
    public NewFM getSpouse() { return spouse; }
}
```

	Result (P/F)	If failed, value returned by second argument?
import junit.framework.TestCase;		
public class NewFMTester extends TestCase {		
public void testConstructor() {		
NewFM papa = new NewFM("Papa", "Smurf");		
assertEquals("Papa Smurf", papa.toString());	----	-----
}		
public void testConstructorGetName() {		
NewFM smurfette = new NewFM("Smurfette", "Smurf");		
smurfette.swapNames();		
assertEquals("Smurfette", smurfette.getFirstName());	----	-----
}		
public void testMarry1() {		
String smurfy = "Smurf";		
NewFM grouchy = new NewFM("Grouchy", smurfy);		
NewFM brainy = new NewFM("Brainy", smurfy);		
NewFM.marry(grouchy, brainy);		
assertEquals("Smurf-Smurf", brainy.getLastName());	----	-----
}		
public void testMarry2() {		
NewFM vanity = new NewFM("Vanity", "Smurf");		
NewFM.marry(vanity, vanity);		
assertEquals(vanity, vanity.getSpouse());	----	-----
}		
public void testMarryUnmarry() {		
NewFM clumsy = new NewFM("Clumsy", "Smurf");		
NewFM sassy = new NewFM("Sassy", "Smurfling");		
NewFM.marry(clumsy, sassy);		
sassy.unmarry();		
assertEquals(sassy, clumsy.getSpouse());	----	-----
}		
public void testSetNames() {		
NewFM motherN = new NewFM("Mother", "Nature");		
NewFM fatherT = new NewFM("Father", "Time");		
motherN.setLastNames(fatherT);		
assertEquals("Nature-Time", motherN.getLastName());	----	-----
}		
public void testSetNamesAndMarry() {		
String smurfy = "Smurfingly Smurftastic!";		
NewFM painter = new NewFM("Painter", smurfy.substring(0,5));		
NewFM harmony = new NewFM("Harmony", smurfy.substring(11,16));		
harmony.setLastNames(painter);		
NewFM.marry(harmony, painter);		
assertEquals(harmony, painter.getSpouse());	----	-----
}		
public void testSetNamesAndRevert() {		
NewFM grandpa = new NewFM("Grandpa", "Smurf");		
NewFM slouchy = new NewFM("Slouchy", "Smurfling");		
grandpa.setLastNames(slouchy);		
grandpa.revertName();		
slouchy.revertName();		
assertEquals("Smurfling", slouchy.getLastName());	----	-----
}		
}		

Use this page for rough work.

Total Marks = 38

Student #: \_\_\_\_\_

Page 6 of 6

END OF EXAMINATION