

CSC 108H1 F October Midterm 2003

Duration — 50 minutes

Aids allowed: none

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

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

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

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

Lecture Section: \_\_\_\_\_

Instructor: \_\_\_\_\_

*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 7 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: \_\_\_\_\_/18

# 3: \_\_\_\_\_/18

For 1 bonus mark write your student number at the bottom of pages 2-7 of this test.

TOTAL: \_\_\_\_\_/50

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

	Abbreviation	Stands for
To the right are abbreviations the you may use. You must write everything else in full.	S.o.p	System.out.print
	S.o.pln	System.out.println
	I.pI	Integer.parseInt
	JOP.sID	JOptionPane.showInputDialog

**Short Java API descriptions (all methods are public):**

```

class Integer:
    static int parseInt(String s) // = s's value, as an int.
class Double:
    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 letters in this String.
class JFrame:
    JFrame() // An empty window with no title, not visible on the screen.
    JFrame(String s) // An empty window title s, not visible on the screen.
    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 setLocation(int x, int y) // set this JFrame's location to (x, y).
    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 show() // make this JFrame visible.
class JOptionPane:
    static String showInputDialog(String m) // get input from the user, prompting with m.

```

**Question 1.** [14 MARKS]

1. Given two integer variables **distance** and **speed**, write an expression that divides **distance** by **speed** using floating point arithmetic. (A fractional result should be produced.)
2. Write an expression that evaluates to **true** if the value of the integer variable **numberOfPrizes** is divisible (with no remainder) by the integer variable **numberOfParticipants**. Assume that **numberOfParticipants** is not zero.
3. Write a statement that declares a **Date** reference named **rightNow**, and creates and assigns it an instance of **Date** that stores the current date and time.
4. Suppose a class call **Nominee** has been defined. Given two **Nominee** reference variables called **first** and **second**, write some code that swaps their references. Declare any additional variables as necessary.
5. Given that **int** variables **x** and **y** have already been declared and assigned values, write an expression that evaluates to **true** if **x** is positive (including 0) and **y** is negative.
6. You are given a class named **Student** that has one **int** instance called **courseMark**. Write a constructor for the class **Student** that takes one parameter, an **int**, and gives this value to **courseMark**.
7. Suppose a **String** variable named **sentence** is declared and assigned a sequence of words separated by single blanks. Write an assignment statement that removes the first word and stores the result in **sentence**. For example, if the **String** was initially "This midterm is easy." the result would be "midterm is easy." You can assume that the **String** has at least one space.

**Question 2.** [18 MARKS]

Write a customized `JFrame` class that includes a method called `stretch`. The `stretch` method reduces the height of the `JFrame` by half and doubles the width of the `JFrame`. The `JFrame` does not change location. Do not worry about the stretched `JFrame` going off the edge of the display. Don't forget to import `JFrame` from package `javax.swing`.

**Question 3.** [18 MARKS]

Consider these two classes.

```

public class Easier {
    private String easyString;

    public Easier(String s) {
        easyString = s;
    }

    public void setEasier(String s) {
        easyString = s + easyString;
    }

    public String getEasyString() {
        return easyString;
    }
}

public class Harder {
    private int hardValue = 10;
    private String hardString = "term";
    private Easier easy;
    private static int hardNum = 0;

    public Harder(Easier e) {
        easy = e;
        easy.setEasier(hardValue + "mid");
        hardNum = hardNum + 1;
    }

    public Easier getEasier() {
        return easy;
    }

    public static int getHardNum() {
        return hardNum;
    }

    public int getHardValue() {
        return hardValue;
    }

    public String toString() {
        return hardValue
            + easy.getEasyString();
    }
}

```

On the next page is a `TestCase` subclass that tests these classes. There are four `test` methods and 8 `assertEquals` calls. To the left of every `assertEquals` call, write:

**P** if the `assertEquals` passes,

**F** if the `assertEquals` fails, and

**N** if the `assertEquals` call is not reached because a previous `assertEquals` failed.

To the right of each failure write the correct expected value.

	Result (P/F/N)	If failed, expected value?
import junit.framework.TestCase;		
public class MT extends TestCase {		
public void testEasierConstructorAndGetters() {		
Easier b1 = new Easier("midterm");		
assertEquals("midterm", b1.getEasyString());	----	-----
}		
public void testEasierSetterAndGetters() {		
Easier b1 = new Easier("midterm");		
b1.setEasier("test");		
assertEquals("test", b1.getEasyString());	----	-----
}		
public void testHarderConstructorAndGetters() {		
Easier b1 = new Easier("midterm");		
Harder c1 = new Harder(b1);		
assertEquals	----	-----
(Integer.parseInt		
(c1.getEasyString().substring(0, 2)),		
c1.getHardValue());		
assertEquals(1, Harder.getHardNum());	----	-----
assertEquals("20midtermmid", c1.toString());	----	-----
}		
public void testHarderMore() {		
Easier b1 = new Easier("ack");		
Harder c1 = new Harder(b1);		
assertEquals(10, c1.getHardValue());	----	-----
assertEquals(1, Harder.getHardNum());	----	-----
assertEquals("1010midack", c1.toString());	----	-----
}		
}		

Use this page for rough work.

Use this page for rough work.

**You can tear this page off, but we will collect it. You must fill in your student number if you tear it off, and you will lose 20% if you keep this page. Why? Because in the past students have passed detailed notes about the midterm to friends in later sections. Bleah.**

Total Marks = 50

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

Page 7 of 7

END OF EXAMINATION