CSE 413 Winter 2001 Midterm Exam

Name ______________________________ ID # ___________ Score _________

_____ _____ _____ _____ _____ _____ _____ _____
1 2 3 4 5 6 7 8

There are **8** questions worth a total of **75** points. Please budget your time so you get to all of the questions. ***Keep your answers brief and to the point.***

You may refer to the textbook (Budd's *Understanding Object-Oriented Programming using Java*) the Scheme report, and clean copies of the course handouts *only*. No other books or notes are allowed.

Please wait to turn the page until you are told to begin.

CSE 413 Winter 2001 Midterm Exam

Question 1. (18 points, 2 points each part) Suppose we enter the following top-level definitions into a Scheme interpreter.

```
(define a 17)
(define b 42)
(define c 100)
(define n (list 1 (+ 1 1) (- c 1)))
(define colors '(red green blue))
(define p '((1 2) (3 4)))
(define (f b) (+ a b))
(define g (lambda (x y) (* x y)))
```

What is the value of each of the following expressions, given that the above definitions are in effect? If evaluating the expression produces an error, explain what is wrong.

```
a) (car p)
```

```
b) (cdar p)
```

```
c) (cddr p)
```

```
d) (cddr g)
```

```
e) n
```

```
f) (cons (reverse (cdr colors)) p)
```

```
g) (append (reverse (cdr colors)) p)
```

```
h) (let ((a 10)
         (c (+ a 1)))
       (f c))
```

```
i) (let* ((a 10)
          (c (+ a 1)))
       (f c))
```

CSE 413 Winter 2001 Midterm Exam

Question 2. (10 points) Write a Scheme function `divisors-of` that returns a list of numbers that exactly divide its positive integer argument. Examples:

```
(divisors-of 12) => (1 2 3 4 6)
(divisors-of 28) => (1 2 4 7 14)
(divisors-of 17) => (1)
(divisors-of 1)  => ()
```

Feel free to use appropriate Scheme library functions in your solution. You can also write additional auxiliary functions if that is helpful.

Question 3. (10 points) A perfect number is an integer that is the sum of its positive integer divisors. The first three perfect numbers are

- 6 = 1 + 2 + 3
- 28 = 1 + 2 + 4 + 7 + 14
- 496 = 1 + 2 + 4 + 8 + 16 + 31 + 62 + 124 + 248

Write a Scheme function `perfect-numbers` that returns a list of the first *n* perfect numbers. Examples:

```
(perfect-numbers 0) => ()
(perfect-numbers 1) => (6)
(perfect-numbers 3)  => (6 28 496)
```

You should use function `divisors-of` from question 2 when you need to calculate the divisors of a number. (Assume that it works as specified, even if you're not sure that your solution to question 1 is entirely correct.) You can write additional helper functions if needed, and you should use functions from the standard Scheme library as needed.

Question 4. (6 points) Suppose we have defined a list of numbers at top level:

```
(define nums '(2 -6 4 0 9))
```

Write a single scheme expression involving `nums` that evaluates to a list whose elements are each three times the corresponding values in `nums`. If `nums` is defined as above, your expression should evaluate to

```
(6 -18 12 0 27)
```

Here's the catch: the expression should produce the right answer regardless of the current length of `nums` or the integer values in it. In other words, the literal expression `(6 -18 12 0 27)` is not the right answer.

You ***may not*** define any new functions (i.e., no `(define …)` at top-level).

Hint: Lambda and higher-order functions.

Question 5. (10 points) In one of the homework problems, we defined constructor and accessor functions to manipulate binary trees in Scheme. The definitions were

```
(make-node value left right) => (value left right)
(value-of    '(value left right)) => value
(left-child  '(value left right)) => left
(right-child '(value left right)) => right
```

A tree node is an ***internal node*** if it is not a leaf of the tree. In other words, interior nodes have at least one child.

Write a function

```
(n-internal aTree)
```

that returns the number of internal nodes in `aTree`. If `aTree` is empty, `n-internal` should return 0. Use the accessor and constructor functions given above to manipulate the nodes in `aTree`.

Question 6. (15 points, 3 each) A few short-answer questions about Java.

a) Methods in a class may be declared `protected` or `private` (among other things). What's the difference between these two?

b) A well-written Java class should contain a `toString` method. What is the purpose of this method? When is it called?

Question 6 (cont).

c) In Java, there is a fairly elaborate hierarchy of classes whose instances represent exceptions. Part of the hierarchy looks like this.

```
class Exception …
class RuntimeException extends Exception …
class IndexOutOfBoundsException extends RuntimeException …
class NullPointerException extends RuntimeException …
class IOException extends Exception …
class FileNotFoundException extends IOException …
…
```

The question is, why does this hierarchy of exception classes exist? Why is it useful to have many different, related classes instead of a single class `Exception`?

d) Several recent Java compilers have included an analysis stage that detects objects that are truly local variables inside methods. That is, they detect objects that are allocated during the execution of a method, are only used inside the method, and cannot be referenced after the method returns. Why bother to accumulate this information? How is this information used? Why is this effective?

Question 6 (cont).

e) Suppose we have a class that represents the balance in a bank account.

```
class BankAccount {
     private double balance;    // account balance

     public BankAccount(double initialDeposit) {
          balance = initialDeposit;
     }

     public double getBalance() { return balance; }

     public void deposit(double amount) {
          balance = balance + amount;
     }
}
```

Recently, a program that uses these objects has been enhanced to use multiple threads so it can process several transactions at the same time. Unfortunately, the auditors have discovered that account balances are occasionally incorrect. What’s wrong, and how could you fix it? (Getting rid of the multithreading is not an option for fixing it.)

Question 7. (3 points) What output does the following Java program produce when method `babble` is called?

```
class That {
     // return class name
     public String name() {
        return "That";
    }
}

class Thing extends That {
    public String name() {
        return "Thing";
    }

    public babble() {
        That ref = (That)this;

        System.out.println("this.name()  = " + this.name());
        System.out.println("ref.name()   = " + ref.name());
        System.out.println("super.name() = " + super.name());
    }
}
```

Question 8. (3 points) The following program attempts to track the coordinates of the mouse and print them to System.out. But it doesn't compile. What's wrong? (If there's more than one error, identify all of them.)

```
import java.awt.event.*;

public class Cat implements MouseMotionListener {

      showMouseCoordinates(int x, int y) {
            System.out.println("mouse x = " + x + ", y = " + y);
      }

      public void MouseMoved(MouseEvent e) {
            showMouseCoordinates(e.getX(), e.gety());
      }

      public void MouseDragged(MouseEvent e) {
            showMouseCoordinates(e.getX(), e.gety());
      }

      public static void main(String [] args) {
            addMouseMotionListener(this);
      }
}
```