CS61B, Fall 2004 **HW #4** **P. N. Hilfinger**

Due: Wed., 29 September 2004

Create a directory to hold your answers to this homework set. Copy the files from `$master/hw/hw4` into this directory. Use the command `submit hw4` to submit your solutions to the problems below.

1. Implement the following *without* using the operators `+, -, /, *,` or `%`, `++`, `--`, `+=`, `-=`, `*=`, `/=`, `%=`, or any other classes from the Java class library or elsewhere. Just use the bitwise operators `&, |, ^` and `~`, `<<`, `>>`, `>>>`, and `&=`, etc. You may include one **for** loop with constant bounds (such as '**for** `(i = 0; i < 32; i += 1)...`'). The usual admonition applies: *Don't fight the problem!*

```
public class Adder {
  /** The value x+y. */
  public static int add(int x, int y) {  /* FILL IN */ }
}
```

2. Assuming `x` and `y` are both `int`s, what does the following do and why?

```
x ^= y;  // NOTE: Same as x = x ^ y
y ^= x;
x ^= y;
```

Demonstrate it empirically by filling in the template `xor.java` in the `hw4` directory.

3. I want to create a class called `Unique` such that there is exactly one object of type `Unique`—one is always created (with **new**), and no more can be created after that. Furthermore, I want any attempt at creating additional `Unique`s to be caught by the compiler (i.e., before the program is ever run). Devise a way to do this in Java and give an example. This is a thought problem, so just put your answer in an ordinary text file called `hw4.txt`.

4. A singly linked list structure (like `IntList`) can be circular. That is, some element in the list can have a tail (next) field that points to an item *earlier* in the list (not necessarily to the first element in the list). Come up with a way to detect whether there is such a circularity somewhere in a list. Do *not,* however, perform any **new** operations, or otherwise create new objects, and don't modify any existing objects in the process. Use just simple list pointers without changing any fields of any list. See `List.java` in the `hw4` directory.

5. We want a data type that provides an array of integers that are limited in range to -8 to 7. Such integers are representable in 4 bits. Let's suppose we have an application that strains our computer's primary memory capacity and need to fit large arrays of these integers into as little space as possible. Specifically, I'd like to be able to store N integers in an $N/8$-**int** array (packing 8 4-bit integers into each **int**). Fill in the template below to provide a suitable small-int array type. Do *not* perform any **new** operations in the implementation (you may include as many as

you want for testing, if you put them in a different file). Use the template `SmallIntArray.java` in the `hw4` directory.

```
public class SmallIntArray {
   /** Create an array of N 4-bit signed integers (range -8 to 7)
    *  that are stored in the array REP, which must have at
    *  least N/8 elements. */
   public SmallIntArray (int N, int[] rep) { /* FILL IN */ }

   /** The size of this array (number of 4-bit components) */
   public int size () { /* FILL IN */ }

   /** The current value of item #K, 0 <= K < size().  Throws
    *  IndexOutOfBoundsException if K not in range. */
   public int get (int k) { /* FILL IN */ }

   /** Set the current value of item #K, 0 <= K < size() to X,
    *  -8 <= X < 8.  Throws IndexOutOfBoundsException if K not in
    *  range, and IllegalArgumentException if X not in range. */
   public int set (int k, int x) { /* FILL IN */ }

   /* FILL IN FIELDS */
}
```