CS61B, Fall 2009 **HW #5** **P. N. Hilfinger**

Due: Wed., 7 October 2009

Create a directory to hold your answers. There is a skeleton for your solutions in the repository under `staff/hw5`, and also in the directory `~cs61b/code/hw5`. Put non-program answers in a file hw5.txt. Use the usual command sequence to copy your final solution to a `hw5-`*N* entry in your tags repository directory.

1. Fill in the missing parts (only) of the following method *without* using the operators `+, -, /, *,` or `%`, `++`, `--`, `+=`, `-=`, `*=`, `/=`, `%=`, or any method calls. Just use the bitwise operators `&, |, ^` and `~`, `<<`, `>>`, `>>>`, and `&=`, etc. 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 */
      for (i = 0; i < 32; i += 1) {
          /* FILL IN */
      }
      /* FILL IN */
  }

}
```

2. 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 `Nybbles.java` in the `hw5` directory.

```
/** Represents an array of integers each in the range -8..7.
 *  Such integers may be represented in 4 bits (called nybbles). */
public class Nybbles {
    /** An array of size N. */
    public Nybbles (int N) {
        // DON'T CHANGE THIS.
        data = new int[ (N+7) / 8 ];
        this.N = N;
    }
    /** The size of THIS. */
    public int size () { return N; }

    /** Return the Kth integer in THIS array, numbering from 0.
     *  Assumes 0 <= K < N. */
    public int get (int k) {
        if (k < 0 || k >= N)
            throw new IndexOutOfBoundsException ();
        else
            return /* REPLACE WITH ANSWER */ 0;
    }

    /** Set the Kth integer in THIS array to VAL.  Assumes
     *  0 <= K < N and -8 <= VAL < 8. */
    public void set (int k, int val) {
        if (k < 0 || k >= N)
            throw new IndexOutOfBoundsException ();
        else if (val < -8 || val >= 8)
            throw new IllegalArgumentException ();
        else
            data[ /* REPLACE WITH ANSWER */ 0 ] =
                /* REPLACE WITH ANSWER */ 0;
    }

    // DON'T CHANGE OR ADD TO THESE.
    private private int N;
    private private int[] data;
}
```

3. Suppose that $f(n)$ is a positive, non-decreasing function. Show that $\lceil f(n) \rceil \in O(f(n))$.

4. Suppose that $p(x)$ is any polynomial in x with positive coefficients. Show that $\log p(x) \in O(\log x)$.

5. Show that $\log_b f(x) \in \Theta(\log f(x))$ for any constant $b > 1$.

6. What is the worst-case running time for the following program fragment?

```
int j;
j = 0;
for (int i = 0; i < N; i += 1) {
   for ( ; j < M; j += 1) {
      if (bump (i, j))
         break;
      }
    }
```

Assume that `M` and `N` are integer constants, and `bump` is a constant-time ($O(1)$) method that returns a boolean result. We're looking for a Θ result that uses `M` and `N`.