

Lab Exercises. (Due: Tues., 25 September 2001 in lab) Copy the directory `$master/hw/lab3` to a directory of your own, using commands such as

```
cp -r $master/hw/lab3 .
```

(Sometimes we are tardy getting these directories set up. If you are working on this before lab and get messages such as “Permission denied” or “cannot access,” it means we’re not quite ready yet, or you misspelled something.) In this directory, you should find a file called `README`, with directions about what to do during the scheduled lab. We strongly suggest that you read over this file *before* going to your lab.

We intend that you finish the lab exercises *in lab* and have your TA check them off. You can have any TA in any lab section check off your lab. As you can see, you have 11 days to complete the lab.

Homework Exercises. (Due: Fri., 21 September 2001 at midnight) Create a directory to hold your answers to this homework set. Copy the files from `$master/hw/hw3` into this directory. Use the command `submit hw3` to submit your solutions to the problems below.

1. Read the documentation for the class `java.io.Reader` (see page 176 of *Programming Into Java* or the on-line documentation). Create a class that extends `Reader`, and provides a kind of reader that translates the characters from another `Reader`:

```
public class TrReader extends Reader {
    /** A new TrReader that produces the stream of characters produced
     * by STR, converting all characters that occur in FROM to the
     * corresponding characters in TO. That is, change occurrences of
     * FROM.charAt(0) to TO.charAt(0), etc., leaving other characters
     * unchanged. */
    public TrReader (Reader str, String from, String to) {
        // FILL IN
    }

    // FILL IN
}
```

See the template file `~cs61b/hw/hw3/TrReader.java`.

2. Using the `TrReader` class from problem #1, fill in the following function. You may use any number of ‘new’ operations, *one* other (non-recursive) method call, and that’s all. In addition to `String`, you are free to use any library classes whose names contain the word `Reader` (check the on-line documentation), but no others. See the template file `~cs61b/hw/hw3/Translate.java`.

```
/** The String S, but with all characters that occur in FROM changed
 * to the corresponding characters in TO. */
static String translate (String S, String from, String to)
{
    char[] buffer = new char[S.length ()];
    // NOTE: This try {...} catch is a technicality to keep Java happy.
    try {
        // FILL IN
    } catch (IOException e) { return null; }
}
```

3. Fill in the Java classes on the next page to agree with the comments. However, do *not* use any `if`, `switch`, `while`, `for`, `do`, or `try` statements, and do not use the `?:` operator. The `WeirdList` class may contain only private fields. The methods in `User` should *not* use recursion. *DO NOT FIGHT THE PROBLEM STATEMENT!* I really meant to impose all the restrictions I did in an effort to direct you into a solution that illustrates object-oriented features. You are going to have to think. The answers are quite short. See the templates in `WeirdList.java` and `IntUnaryFunction.java`, in directory `~cs61b/hw/hw3`.

```
/** An IntUnaryFunction represents a function from
 * integers to integers. */
public interface IntUnaryFunction {
    /** The result of applying this function to X. */
    int apply (int x);
}

/** A WeirdList holds a sequence of integers. */
public class WeirdList {
    /** The empty sequence of integers. */
    public static WeirdList EMPTY = // FILL IN;

    /** A new WeirdList whose head is HEAD and tail is
     * TAIL. */
    public WeirdList (int head, WeirdList tail) { /* FILL IN */ }

    /** The number of elements in the sequence that
     * starts with THIS. */
    public int length () { /* FILL IN */ }

    /** Apply func.apply to every element of THIS WeirdList in
     * sequence, and return a WeirdList of the resulting values. */
    public WeirdList map (IntUnaryFunction x) { /* FILL IN */ }

    /** Print the contents of THIS WeirdList on the standard output
     * (on one line, each followed by a blank). Does not print
     * an end-of-line. */
    public void print () { /* FILL IN */ }

    // FILL IN WITH *PRIVATE* FIELDS ONLY.
    // You should NOT need any more methods here.
}

// FILL IN OTHER CLASSES HERE (HINT, HINT).

class User {
    /** The result of adding N to each element of L. */
    static WeirdList add (WeirdList L, int n) { /* FILL IN */ }

    /** The sum of the elements in L */
    static int sum (WeirdList L) { /* FILL IN */ }
}
```