Midterm Test

READ THIS PAGE FIRST. *Please do not discuss this exam with people who haven’t taken it.* Your exam should contain 5 problems on 11 pages. Officially, it is worth 40 points (out of a total of 200).

This is an open-book test. You have two hours to complete it. You may consult any books, notes, or other inanimate objects available to you. You may use any program text supplied in lectures, problem sets, or solutions. Please write your answers in the spaces provided in the test. Make sure to put your name, login, and lab section in the space provided below. Put your login and initials *clearly* on each page of this test and on any additional sheets of paper you use for your answers.

Be warned: my tests are known to cause panic. Fortunately, this reputation is entirely unjustified. Just read all the questions carefully to begin with, and first try to answer those parts about which you feel most confident. Do not be alarmed if some of the answers are obvious. Should you feel an attack of anxiety coming on, feel free to jump up and run around the outside of the building once or twice.

Your name: ___________________________ Login: ____________

1. ____________/10

Login of person to your Left: ___________ Right: ___________

2. ____________/10

Discussion section number or time: _______________________

3. ____________/10

Discussion TA: __________________________

4. ___________

Lab section number or time: _______________________

5. ____________/10

Lab TA: __________________________

TOT ____________/40
Throughout this test, assume that the following definition is available.

class IntList {
    public int head;
    public IntList tail;

    /** The IntList whose head is HEAD and whose tail is TAIL. */
    public IntList (int head, IntList tail) {
        this.head = head; this.tail = tail;
    }
}

1.  [10 points] For each of the following assertions, indicate whether it is true or false. Give a clear and brief argument for your answer (not a formal proof; for example, it is acceptable to show that the question is a variation of some example from lectures or readings).

   a. \( 1 + \frac{\sin x}{x} \in \Theta(0.01) \)

   b. \( 1000 \sin x \in O(1) \)

   c. \( \sqrt{n} \in \Omega\left(\frac{n^2}{2^n}\right) \).

   d. The worst-case time for executing the call \( f(n) \) is \( \Theta(n) \). Assume that each call to \( h \) and \( g \) executes in worst-case constant time.

```
void f (int n) {
    int u;
    u = 8;
    for (int i = 1; i <= n; i += 1) {
        if (i > u) {
            for (int j = 0; j < u; j += 1)
                h (j);
            u = 2*u;
        }
        g (i);
    }
}
```
e. The worst-case time for executing the call $F(A)$ is $\Omega(2^n)$, where $n$ is the size of array $A$.

```java
void F(int[] A) {
    if (A.length <= 1)
        return;
    int[] A1 = new int[A.length/2],
    A2 = new int[A.length/2];
    for (int i = 0; i < A1.length; i += 1) {
        A1[i] = A[i];
        A2[i] = A[A.length - i];
    }
    F(A1);
    F(A2);
}
```
2. [10 points] The methods in this problem take two `IntList` arguments: $P$, which consists only of 1’s and 0’s, and $L$, an arbitrary list. They produce what we’ll call “$L$ masked by $P$,” defined as follows: $L$ masked by $P$ is a subsequence of $L$ that contains element number $i \geq 0$ of $L$ iff $P$ has an element number $i$ and that element is 1. Thus, the elements of $L$ that are included in $L$ masked by $P$ occur in the same order as in $L$. For example, if $P$ contains $[0, 1, 1, 0, 0, 1]$ and $L$ contains $[1, 2, 3, 4, 5, 6, 7]$, then $L$ masked by $P$ is $[2, 3, 6]$.

a. [4 points] Fill in the function below to satisfy its specification.

```java
/** Returns $L$ masked by $P$ non-destructively. */
static IntList mask (IntList P, IntList L) {
    // FILL IN (about 6 lines)
}
```

b. [5 points] Fill in the function below to satisfy its specification. Give an iterative solution that contains no method calls (recursive or otherwise). Although the result may contain no new `IntList` nodes, you may create one extra `IntList` node for internal use (it is not necessary to do so, but might make your solution simpler).

```java
/** Returns $L$ masked by $P$. The result contains no new IntList nodes.
* The original list referenced by $L$ may be destroyed. */
static IntList dmask (IntList P, IntList L) {
    // FILL IN (about 10 lines)
}
```

c. [1 point] Show how you would create one list that could be used as a mask ($P$) argument to select every other value of any list, $L$ (that is, the same $P$ would select items #0, 2, 4, ..., of any $L$).
3. [10 points] You’ve been asked to help out with a certain typography system. The basic type is the **Box**, which is intended to represent something that has a width and a horizontal position on a line and that can be printed. A Box may have *constituents*: Boxes that are contained within it and contribute to its width. There are several kinds of **Box**.

**Glyph** objects represent single characters. Printing them calls a primitive static method `Page.print(c, x)`, which prints character `c` at position `x` on a line. They contain no constituents.

**Leading** objects represent blank space. They print nothing, but have a width. They also have no constituents. (The name comes from the fact that in old-fashioned typography, blank space is represented by blank lead slugs.)

**Hbox** objects simply contain sequences of other **Boxes**. Changing the position of an **Hbox** moves all its constituent Boxes; and printing it prints its Boxes. Its width is the sum of the widths of its constituents; its position is the same as that of its leftmost constituent.

Usually, one represents a line of type with an **Hbox**, which contains all the Glyphs, Leading, and smaller Hboxes that make up a line. After assembling them all into a line, a program will typically set the position of the line (which is also the position of its leftmost constituent box) and then *print* the line, which ultimately prints each constituent Glyph at its proper place.

For example, the following diagram illustrates an **Hbox** (rectangles with rounded corners) containing two smaller Hboxes separated by Leading (dashed box). The two smaller Hboxes contain the Glyphs (rectangles) for “Hello world!” The position of the large Hbox has been set to 50. Each Box has its width indicated on top and the position of its left-hand corner indicated below.

```
```

a. Read the restrictions here carefully! Fill in the missing parts of the implementation on the next three pages. Some of the blanks should be left blank. Mark these explicitly with a little comment (`/* - */`) so that we can see you didn’t get them right by accident! The only explicit looping in your program should be in the bodies of the `traverse` methods. For all other iteration, call `traverse` with appropriate arguments. Do not introduce auxiliary methods; that is, you may override methods, as needed, but don’t introduce new method names or overload methods with others having different types of parameters.

b. Suppose that all subtypes of **Box** are in the package `typesetting`. What is the most restrictive access modifier (`public`, `private`, etc.) that I can replace the two instances of `ℵ` with?

c. If I might create subtypes of **Box** outside of the `typesetting` package, what is the most restrictive access modifier I can replace `ℵ` with?
package typesetting;

abstract public class Box {
    /** My width. */
    abstract public int getWidth ();
    /** The current horizontal position of my left edge on the line. */
    public int getPosn () { return posn; }
    /** Set my current horizontal position to X. (Default implementation) */
    public void setPosn (int x) { posn = x; }

    /** Print me at my current position on the Page. (Default implementation) */
    public void print () {
    }

    /** Add B as my rightmost constituent. Throws an exception if
     * I cannot contain constituents. (Default implementation) */
    public void add (Box b) {
    }

    /** Apply V.apply to each of my constituents (if any). (Default implementation). */
    N void traverse (Visitor v) {
    }

    private int posn;
}
public class Glyph extends Box {
    /** A new Box whose width is WIDTH that will print the character
     * C at its current position. */
    public Glyph (char c, int width) {
        super();
    }
    // FILL IN OTHER METHODS AND FIELDS, IF NEEDED.
}

public class Leading extends Box {
    /** A new Box that has width WIDTH, but prints nothing. */
    public Leading (int width) {
        super();
    }
    // FILL IN OTHER METHODS AND FIELDS, IF NEEDED.
}

public class Hbox extends Box {
    /** A new Box that consists of a sequence of other Boxes
     * ("constituents"). Its width is the total width of its
     * constituents. Setting the position of an Hbox changes
     * the positions of all its subboxes, and printing it
     * prints all its subboxes. */
    public Hbox () {
    }
    public setPosn (int x) {
        super.setPosn (x);
        // FILL IN (remember: no loops!)
    }
}

Continues on next page.
\begin{verbatim}
N traverse (Visitor v) {
    // FILL IN
}
// FILL IN OTHER METHODS AND FIELDS OF Hbox, AS NEEDED (no loops!)

// PUT ANY ADDITIONAL CLASSES HERE (continue on back if needed).
\end{verbatim}
4. [1 point] In what city would you most likely have found the author of the following quotation?

“If we take in our hand any volume of divinity or school metaphysics, for instance, let us ask, Does it contain any abstract reasoning concerning quantity or number? No. Does it contain any experimental reasoning concerning matter of fact and existence? No. Commit it then to the flames, for it can contain nothing but sophistry and illusion.”

5. [10 points] Give short, concise answers to the following:

a. Give three (type int) values of $x$ for which the following is true:

$$(((x-1) \land x) \land x) == x$$

b. Give a regular expression that matches all and only strings of the form “$(i_1, i_2, \ldots, i_n)$” for $n \geq 0$, where the $i$’s are all decimal integers (possibly negative). So the expression should match $(1,2,-42)$ and $(13)$, but not $(1,2-1,3)$ or $(jack,jill)$.

c. By hand, I can create a file `commands.c` from a file `commands.in` using the command

```
convertCommands commands.in > commands.c
```

and I can create the file `commands.o` from the two files `commands.c` and `commands.h` using

```
gcc -g -c commands.c
```

Finally, I can create the executable file `editor` from `commands.o` and `editor.o` using

```
gcc -o editor -g editor.o commands.o
```

What could I put into a makefile so that `gmake editor` would execute only the commands that are needed to produce a version of `editor` that is up to date with any changes to the files `commands.h`, `commands.in`, and `editor.o`?
d. The following program fragment cannot do what the comment suggests. Why not? No, we are not going to tell you what’s inside \texttt{sliceOut}. It is \textit{not} sufficient just to answer that “array objects cannot shrink once they are created” (although that statement is true).

\begin{verbatim}
    int[] L;
    L = new int[] { 1, 2, 3, 4, 5, 6, 7, 8, 9 };  
    sliceOut(L, 3, 7);
    /* The array in L is now [ 1, 2, 3, 8, 9 ] */
\end{verbatim}

e. Your program blows up with a \texttt{NullPointerException}. Unfortunately, it is a rather large program and executes several million instructions before throws this exception. How could you use \texttt{gjdb} to track down the problem, without modifying the program? Please give specific advice in your answer (as opposed to vague suggestions like “Set a breakpoint.”).