Exam 2, CSCI 210, Spring 2004

Name: _____

1. **a.** [5 pts] Draw a logic circuit corresponding to the Boolean expression $\overline{x + yz}$.

b. [5 pts] Reduce the Boolean expression $\overline{x + yz}$ to sum-of-products form. (You need not show intermediate steps.)

2. [10 pts] Suppose we are designing the mechanism for communication between a CD-ROM drive and the CPU. Since a CD-ROM drive takes so long to respond to a request, we want to design it to allow an operating system to use the response time entirely for computing, without resorting to forcing the OS to poll the drive periodically to check whether a response is ready yet. How do hardware designers allow for this? (Your answer should amount to more than a single-word answer.)

3. [10 pts] Suppose we were to run the C program at right on a Unix computer. Which of the following outputs might occur? (Place a checkmark to each possible output.)

__ **a.** `1 1 1`
__ **b.** `1 1 1 1`
__ **c.** `1 2 1 2`
__ **d.** `1 2 2 3`
__ **e.** `1 2 3`
__ **f.** `2 2 2`
__ **g.** `2 2 2 2`
__ **h.** `3 2 1`

```
#include <stdio.h>

int i = 1;

int main() {
    if(fork() == 0) i++;
    if(fork() == 0) i++;
    printf("%d ", i);
    exit(0);
}
```

4. [15 pts] For the Java program at right, suppose we have two threads, `a` and `b`, where `a` executes the `runA` method and `b` executes the `runB` method. Give an example of code for `runA` and `runB` which could result in deadlock between the two threads.

```
public class Deadlock {
    Thread a;
    Thread b;

    public void runA() {




    }
    public void runB() {




    }
}
```

5. [10 pts] Language-defined *library functions* (such as `printf`) are less efficient than OS-defined *system calls* (such as `write`), since the library functions must translate their arguments into a series of system calls. Why, then, do programmers usually prefer to design programs using library functions rather than the system calls? (One reason is sufficient.)

6. [20 pts] Modify the two classes below so that when the user runs `Main`'s `main` method, it allows the user to type "begin" to initiate a `Counter` thread and "end" to stop the currently running `Counter` thread. (Don't worry about the possibility that the user might type "begin" while a `Counter` thread is going or "end" when no `Counter` thread is going.) The `Counter` thread should display a counter as it increments each second. The following is code to accomplish this.

```
while(true) {
    ++counter;
    System.out.println(counter);
    try { Thread.sleep(1000); }
    catch(InterruptedException e) {}
}
```

```
begin
1
2
3
4
5
end
begin
1
2
3
end
```

The transcript at right illustrates how the program should work.
(Your program should not use any deprecated methods from the `Thread` class. If you stick to what we discussed in class, this won't be a problem for you.)

```
public class Counter extends Thread {
}
```

```
import java.io.*;
public class Main {
    public static void main(String[] args) {
        BufferedReader user = new BufferedReader(
            new InputStreamReader(System.in));


        while(true) {
            try {
                String line = user.readLine();
                if(line.equals("begin")) {



                } else {



                }
            } catch(IOException e) {}
        }
    }
}
```

7. [5 pts] Approximate 2^{64} in the form $x \times 10^y$, with x and y both being base-10 numbers. (Your answer need not be normalized.)