The Power of Recursion

10-28-2002

Opening Discussion

- What did we talk about last class?
- Do you have questions about the assignment?

Revisiting Recursion

- A recursive function is a function that calls itself. The fact that the stack keeps distinct memory locations for each call makes this method very powerful.
- We looked at how a simple recursive function could be make to execute a loop and used that to calculate factorials. We also saw that it was at least slightly more powerful than a basic loop and could print inputs in reverse order.

Tracing a Recursive Loop with Printing

- Just to make it completely clear what happens with the flow of control in a recursive program, let's look at out simple loop function, but put in printf statements that show us more explicitly what is happening with the control flow in the program.

Recursion in Multiple Directions

- Recursion is more powerful (and more useful) when the function contains two or more possible calls of itself. In this case, a loop can't be easily used as a substitute without major logic modifications.
- Drawing out the call path of these types of functions produces what computer scientists call a tree.
- Normally I do Fibonacci numbers for this.

Recursive Fibonacci Numbers

- This is the solution to the last minute essay. Notice how incredibly similar it is to the recurrence relationship that we started with. Unfortunately, in this case recursion is very inefficient because work is repeated frequently.

```
int recurFib(int n) {
     if(n<3) return 1;
     return f(n-1)+f(n-2);
}
```

Flood Fill

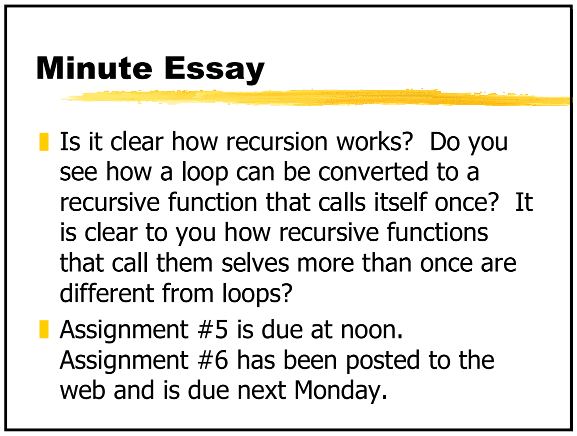
- If you have used a painting package, you are familiar with the idea of "filling" the area around a specific location with a certain color.
- One way of doing this easily in code is with a recursive function that calls itself up to 4 times if adjacent pixels are blank.
- Let's write a bit of code for this.

Debuggers

- You can get help with finding runtime errors, and to some extent logic errors as well. Hopefully you have already used the practice of putting extra printf statements in code to help you figure out what it is doing. If you compile with the -g option you can also use a debugger.
- The debugger on these systems in gdb. You run "gdb a.out" (or whatever your program name is). Typing run at the prompt starts your program. You can pipe things in there too.

More on the Debugger

- When a fault occurs (could be you hitting ctrl-C) you can type in where to get a stack trace including line numbers.
- The command print will print out variable values.
- Use quit to get out.
- There are also many other powerful features in the debugger that allow you to set break points, return from functions, continue after breaks, etc. Use help to see these.



Minute Essay

- Is it clear how recursion works? Do you see how a loop can be converted to a recursive function that calls itself once? It is clear to you how recursive functions that call them selves more than once are different from loops?
- Assignment #5 is due at noon. Assignment #6 has been posted to the web and is due next Monday.