Functions

9-30-2002

Opening Discussion

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

Minute Essay Responses

- Strings, Files, Sorting? Later.
- Stopping a loop. Can you do it without a counter?
- A second return statement?
- What the initializer does. Write a loop to print out some numbers.
- When do you have to Ctrl-C?
- Goto==Bad

More Minute Essays

- Incrementing through odds, evens, or random collections of numbers.
- ++ and -- operators are just shortcuts.
- When to use do-while.
- Programming in other languages?
- Break in loops.
- Drill problems.

Loops Inside of Loops

- Just like conditionals, loops can be nested.
- Doing this results in code that executes a large number of times. When programs run slowly, typically the code you need to optimize is just what is in the inner most loops.

Breaking Problems into Pieces

- A key to solving any complex problem is to break it into pieces that are of a more manageable size and solve each of those, then bring the elements together. This is like the outline view of an algorithm.
  - Top-Down Design: Starting with the whole problem, repeatedly break it into pieces going down to what you can handle.
  - Bottom-Up Design: Starting will little pieces work up to the full problem.

Functions in C

- The way we break problem up in C is to break our code into separate functions.
- You have already seen how we can call functions in our code. Now we look at how to write them.
- The first part of any function is it's signature:

```
return-type name(type a1,type a2,…);
```

- C uses a "single pass" compiler so it must know about methods before they are called. Either put a signature, or the whole function at top.

Parts of a Signature

- C functions are much like normal math functions, but type matters.
- Return type: This is the type of what the function returns. It will be void if the function doesn't return anything.
- Arguments: This is a list of types and names for what is passed into the function. Again, if nothing is passed in it is void.

Function Body

- A method can be declared with a signature followed by a ';'. The function definition will have the signature followed by curly braces enclosing the statements that should be executed.
- As with main, you can put any type of C statements you want in there.

The return Statement

- When the statement "return" is reached, execution returns from the function to the point it was called from.
- For a void function you can use return; in the middle and don't need any at the end.
- For a non-void function return should be followed by an expression of the correct type for what should be returned.

Formal vs. Actual Arguments

- Inside a function arguments act like variables. You don't know what values they will have in the function itself. Those are the formal arguments.
- At the point where the function is called, values are "passed" to the function. These are the actual arguments and determine what the formal arguments will be initialized to.

Minute Essay

- Write a function called minI that takes two integer arguments and returns the smaller of the two.
- Assignment #3 is due today and quiz #3 will be at the beginning of next class.