Functions

9/10/2007

Opening Discussion

- Let's look at some solutions to the interclass problem.
- Minute Essay Questions
  - Ctrl-C terminates a process (like a program that won't quit).
  - Difference between int and double.
  - Purpose of different operators: sizeof, &, and ++.
  - Looking up Linux commands and uses of Linux commands (like grep) or slashes in paths. Difference between > and >> when redirecting output.
  - Hair care.
  - void main(void)?
  - Importance of Calculus.
  - ssh
  - Why isn't C like Java?

More Minute Essay Questions

- Do we really do binary conversion in C?
- Are basic C concepts central of other programming languages?
- Will C ever give a “possible loss of precision” error?
- Why doesn't C have heat breath?
- Goto stigma.
- Do I like fishing?

Problem Decomposition

- One of the most significant skills you should learn in this course is how to break problems into smaller, more manageable parts.
- This process is called problem decomposition and it is really an art. Good problem decomposition leads to good programs. It is also a skill that you can use in all facets of your life.
- The best problem decomposition gives you pieces that are flexible and reusable.
- The term top-down design is used to describe how we often decompose problems. We start with the full problem and break it into meaningful pieces repeatedly.

Functions in C

- In C we decompose problems by breaking our program up into multiple functions instead of sticking the whole thing in main.
- You have already written a function in C. That is what main really is. The main function is just a special function that the program always starts with.
- When a function is called, we go into that function and execute the lines in that function. When it returns we jump back to where the function was called from.

Making Your Own Functions

- In C functions can only be written at the global scope. That means you can't put functions in other functions or anything else.
- A function declaration has the following form.
  - returnType functionName(type1 param1,type2 param2, ...)
- This can be followed by a ; to declare a function or by a code block to define the function.
- I will only include function declarations when they are required.
- Functions must be declared or defined before they can be called.

Function Parameters

- You can think of functions just like the things you are used to from math. The only difference is that C functions can have side effects.
- We get information into functions through parameters. If we don't need to tell a function anything we can give void as the parameter list.
- The list specified in the function definition is called the formal parameter list. The values are determined when it is called and the passed values are called actual parameters. The names of formal and actual parameters don't have to be related.
- Let's make a function to see how this works.

Pass-by-Value

- Function arguments are passed by value in C. This means that the function gets a copy of what is passed.
- Changes to the copy have no impact on the original.

Function Returns

- A function gets information back to where it was called by a return value. If a function doesn't return anything we can specify void as the return type.
- In C a function can only directly return one value. We'll see ways of getting around this later in the semester. Your book describes one method, pass-by-reference, in this chapter.
- A return statement needs to be put in any function that doesn't return void and it includes an expression with the value to be returned.

Function Expressions

- Because functions return values, they are valid C expressions and can be used in any place where we expect an expression to occur.
- When the function call is reached, control goes into the function and it executes until it reaches a return statement.
- The returned value is used as the value of the expression.

Minute Essay

- Write a function called timesTwo that takes a single double as an argument and returns twice that number.
- Interclass Problem – Write two functions. One returns the sum of three numbers and the other returns the product of three numbers. Include a main to test that they work.