Intro to C Programming

9-17-2003

Opening Discussion

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

Minute Essays Replies

- Yes, everything on a computer is a mathematical operation. The whole theoretical basis of computers comes from notions of formal systems in math.
- Computer memory is a single long strip of bits. The processor requests them in certain sized chunks. "Op codes" tell what instruction is.
- The sign on a floating point is a bit. 1=negative.
- Xor only returns 1 if two values are different.
- $273_8=2*8^2+7*8+3=128+56+3=187$

More Minute Essay Replies

- Only difference between negative number and large positive is how you interpret it.
- This material will only directly appear in this class if you do some graphics options. Hex is one exception because we typically print pointers as hex.
- C++ is almost a superset of C. It pastes object oriented features on C. However, good C++ programs are very different from good C programs. C++ allows operator overloading so << and >> are used for input and output.

More Minute Essay Replies

- When converting binary to hex figure out what the decimal value of the 4 binary digits is first. Then if it's greater than 9 figure out letter equiv.
- You will need to understand the different types in C and we will talk more about them as the semester goes on. Some conversions are implicit, but not all (signed to unsigned isn't).
- You could use << to find powers of two. For floating point numbers it is done in hardware instead.
- Binary fractions.

A First Example

- We will now look at some code that we started to write and add a bit more to it.
- At this point I would like to note that in many ways programming is a creative endeavor. When you write a program you bring something into existence. You take your thoughts and formalize them into something of substance. Unlike static art, this medium is nearly always dynamic.

Compiling and Executing

- Computers don't understand C code, they only understand machine language. For this reason we have compilers. They are programs that convert high level languages to machine code.
- The compiler you will use in this class is gcc. A man on gcc will give you the many options possible. Here are some:
  - -o: Executable output name. a.out is used be default.
  - -g: Include debug information.
  - -Wall: Print all warnings.
  - -pedantic: Only accepts tighter code.

Parts of a C Program

- #include lines: These lines tell the compiler what libraries you will be using and basically paste their code at the top.
- main: All C programs have a main and this is where the execution of the program begins. Main is a function. We will talk more about functions later. The body is inside curly brackets: '{' and '}'.
- Comments: All text between a /* and a */ is comments and are ignored by the compiler. They can span lines.

Statements

- The body of the main function consists of a series of statements that are executed in order, top-down and left-right.
- All statements end with a ';'. What comes before the ';' must be a valid C expression.
- Note that the statements in main are indented beyond the declaration of main. C ignores extra white space in the program.

Expressions

- An expression can be one of the following:
  - A number
  - A string literal
  - A variable name
  - A function call
  - Two expressions separated by an operator. For this, C must be able to perform the operation on the types of the expressions. Can include parentheses.
- An "atomic" elements is called a token. The first three are tokens and adding white space changes their meaning.

Types and Variable Declarations

- C is a typed language so all expressions in C have a type. C has the following types:
  - char, short, int, long. These all represent integers and can be signed or unsigned.
  - float, double, long double. These represent floating point numbers.
- When you want to keep track of a value in C you declare a variable of the correct type. A variable declaration has the form of "type name1;".

Operators

- Complex expressions in C are built with operators. Here are the numeric operators available that take two arguments.
  - +, -, *, /: Do what you would expect.
  - %: Modulo, the remainder after division.
  - <<, >>: Bit shifting operators.
  - &, |, ^: Bitwise and, or, and xor.
- Here are operators that take one argument.
  - -: Negative.
  - ~: Bitwise negation.
- Tertiary Operator, ?:, takes 3 arguments.

Assignment

- There is also an operator '=' that is an assignment operator. It stores the value of the expression on the right hand side into the memory for what is on the left hand side.
- For the time being the only thing that will ever appear on the left hand side is a variable.
- You can do this in a variable declaration.

Functions

- The last somewhat atomic type of expression listed was a call to a function. You can use functions that exist in other libraries, like printf in stdio, right now. A bit later, we will learn how to define our own functions to help break up problems into smaller pieces.
- Function calls give the name of the function followed by an argument list in parenthesis.

printf the Short Version

- The printf function is how you will print things to screen. It allows you to do formatted output.
- For now you can think of its argument list as a format string followed by values to be put into that string at the %? tokens.
  - %d is for decimal integer
  - %f is for float
  - %c is for character
  - %X prints an integer in hex

Minute Essay

- Write a short C program that declares one integer variable (you pick the name) and stores the value of 2+2 in it.
- Read chapter 3 for next class. The second half of the assignment could prove very challenging if your only information comes from lecture. Come to the next class armed with questions.