Arrays: Part 2

10-18-2003

Opening Discussion

- What did we talk about last class?
- What do you think the picture of memory looks like when we declare an array? We saw last time that arrays are passed by reference. What does that imply? Does it change this picture?

Recap of Arrays

- Last time we introduced arrays and saw that they gave us the ability to store, access, and manipulate numerous pieces of data of the same type using integer indexes.
- Arrays are denoted by the square brackets that follow a variable. When declaring an array the number in brackets gives the size, for use it gives the index, when passing arrays they are left empty.

Arrays in Memory

- What we didn't really talk about is what happens in memory when we declare an array. I said you get a chunk of memory large enough to hold the elements, but there is more.
- The variable for the array name is actually associated with a pointer that points to the block of memory of the proper size.

Array Syntax vs. Pointer Syntax

- If `int a[5];` actually declares a pointer to a block of memory, it is reasonable to ask how the [] notation works with that.
- Just the name, **a**, is the pointer to the memory block.
- As it turns out, the brackets are equivalent to adding an offset and dereferencing.

```
a[i]  <==>  *(a+i)
```

Using const with Function Arguments

- We mentioned yesterday that there is a problem with arrays being passed by reference, mainly that we can mess them up if we aren't careful. This can be avoided. Array and pointer arguments can be preceded with the const keyword which says they can't be modified in the function.

```
int sumArray(const int a[],int len);
```

Strings in C

- Last class I mentioned that strings in C are null terminated arrays of characters. So strings can be treated/passed as arrays of chars or pointers to chars.
- printf and scanf can work with strings using the %s symbol.

```
char buf[20];
scanf("%s",buf);
printf("%s\n",buf);
```

Code

- Now let's write some code to explore the relationship between arrays and pointers and a bit about strings in C.

Minute Essay

- If you have a lower case letter, you can make it upper case by adding the value ('A'-'a'). Given this, write a chunk of code the will make a char array, buf, all upper case.
- Quiz #4 is next class. It will cover pointers and arrays. Assignment #5 is due on Monday.