Arrays

10-15-2003

Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignment (true it isn’t due for a while, but the next one is posted)?

Storing Multiple Values

- When trying to solve problems with a computer program, it is quite common that you want to be able to store data related to many different real world entities without knowing exactly how many you will have.
- With what you have learned, can you do this? How?
Arrays

- The most direct way to store, access, and work with multiple values in C is with arrays.
- Arrays allow you to use integer indexes to access multiple pieces of data of the same type. The index can be any int type expression.
  ```c
  int a[5];
  a[2]=3;
  printf("%dn",a[4]);
  ```

Indexing into Arrays

- Arrays in C are "zero-referenced". This means that the first element of the array has an index of zero. If there are n elements, the last index of the array has index n-1.
- So on the previous slide, the valid expressions with a are a[0], a[1], a[2], a[3], and a[4].

Types of Arrays

- Arrays can be declared for any type in C so you can have arrays of ints, doubles, floats, chars, etc.
- You get a chunk of memory big enough to hold the proper number of that type.
- Strings in C are actually arrays of chars and the end of the string is denoted with a zero stored in the array. These are called null terminated strings.
Initializing Arrays

- When an array is declared, it has garbage in it unless initialized, just like a normal variable.
- You can initialize with the declaration, or using a loop.
  ```c
  int j, grades[] = {95, 87, 89, 94};
  double positions[30];
  for (j = 0; j < 30; j++) {
    positions[j] = 0.0;
  }
  ```

Passing Arrays to Functions

- You can pass arrays in as arguments to a function much like you would any normal value.
- Arrays in C don’t store how long they are though so typically you have to pass in an int giving the length as well.
  ```c
  int SumArray(int a[], int len);
  ```

Summing Arrays

- A common operation to perform on an array is to sum the elements.
- Let’s write the function whose signature was shown on the previous slide to sum the elements of an array.
- What do we need to do to be able to return the sum?
**Code Involving Arrays**

- Now let’s write a different function called `FillArray` that takes an array of ints, a length, and a value `int`, then assigns all the elements to be that value.
- Let’s put a call to this function in our `main` and look at the values in the array before and after the call.

**Pass By Reference**

- What we just saw is an example of pass-by-reference which we talked about last week today. What does this tell you about the arrays and what gets passed?
- Arrays are always passed by reference so you have to be careful when passing them to functions that the function doesn’t cause unwanted side effects.
- Comments on functions should document the side effects.

**Minute Essay**

- Write a function that takes an integer array and the length of the array. Set each element of the array equal to its index.
- I’ll post the midterm grades on the submission application as soon as they are done.