Jan 29

Code will be in jan29.c

Enumeration Type

- Keyword is **enum**. This is equivalent in size to the int.

Typedef

- Ability to define an alias for an existing type. Anything can then be a size. Size can be an int, and an int can be a size.
- Characteristic of well written code if the typedef is nice.

Pointers

- Contains the address of other variables or an address in memory. Everything passed in C is passed by **value**, which is a copy. If you want to change something you must pass the **address**. What about swapping the pointers? This is also a problem. The void pointer is **universal**, that is it works with every kind.

What can we do with pointers?

- Increment by postfix or prefix, add (ex add 17. This does the entity and 17 times it). What happens if we subtract?
- We can assign anything to a void pointer without casting, but assigning a void pointer to something requires *casting*.

Arrays

- Can use subscripts with pointers.
- `char **argv` is an array of pointers.

Pointer Arithmetic

- `p+1` gets the size, and adding goes to the next location in memory. It adds size of what is being pointed to. For example, a `struct+1` adds the entire size of the struct again.
- size of an array over size of elemnts gives number of elements

Strings

- Not first-class data types. There is no `string` data type. There are character arrays. All character "strings" are terminated by a `null` byte.
- The size of a character array is the `size+1`. The extra one is from the null terminating end character.