Relational opperators have a lower precedence than arithmetic operators, so an expression like i<lim-1 is taken as i < (lim-1)

**Data types and sizes**

Variables and constants Are the basic data objects manipulated in a program.

There are signed and unsigned forms of all integer types, and notations for unsigned constants and hexadecimal character constants.

Floating point operations may be done in single precision. There is also a long double type for extended precision. String constants may be concatenated at compiler time.

Objects may be declared const, which prevents them from being changed.

Short -16 bits (2 bytes)

Long – 32 bits (4 bytes)

Whether plain *char* are signed or unsigned, is machine dependent, but printable characters are always positive.

## Enumeration

Enumeration provide a convenient way to associate constant values with names, an alternative to #define wit the advantage that the values can be generated fro you. Although variables of enum types may be declared, compilers need not check that you store in such a variable is a valid value for the enumeration. Nevertheless, enumeration variables fofer the chance fo checking, and so are often better than #defines. In adduition, a debugger was able to print values of enumeration in their symbolic form.

Declaration

* Global and static variables are initialized to static by default

Type conversion

In general, the only automatic conversion are those that convert a “narrower” operarand into a “wider” one without losing information, such as converting an integer to floating point in an expression like f+1.

* A *char*  is just an small *int*

Implicit arithmetic conversions work much as expected. In general, if an operator takes 2 opernds of different arithmetic types, the “lower” type will be converted to the “higher” type.

* If either operand is long double, convert the othert to long double
* Otherwise, if either operand is double, convert the other to double
* Otherwise, if either operand is float, convert the other to float
* Otherwise, convert char and short to int
* Then if , either operand is long, convert the other to long

Since an argument of a function call is an expression, type conversions also take place when arguments are passed to functions. In absence of a function prototype, char and short become int, and float becomes double. This is why we have declared function arguments to be int and double even when the function is called with char and float.

Assignment operators and expressions

\

D:\c\_programming\_tutorial\range.c

D:\c\_programming\_tutorial\atoi.c

D:\c\_programming\_tutorial\detab.c