**SCOPE RULES**

The scope of a variable is the part of the program within which the variable can be used. For an automatic variable declared at the beginning of a function, the scope is the function in which the variable is declared. Local variables of the same name in different functions are unrelated. The same is true of the parameters of the function, which are in effect local variables.

The scope of an external variable or a function lasts from the point at which it is declared to the end of the file being compiled. If an external variable is to be referred to before it is defined, or if it is defined in a different source file from the one where it is being used, then an extern declaration is mandatory.

It is important to distinguish between the ***declaration*** of an external variable and its ***definition***.

If the lines

int sp;

double val[MAXVAL];

appears outside of any function, they ***define*** the external variables sp and val, causes storage to be set aside, and also serve as the declaration for the rest of that source file.

On the other hand, the lines

extern int sp;

extern double val[];

***declare*** for the rest of the source file that ***sp*** is an int and that ***val*** is a double array (whose size is determined elsewhere), but they do not create the variables or reserve storage for them.

There must be only one definition of an external variable among all the files that make up the source program; other files may contain extern declarations to access it. Array sizes must be specified with the definition, but are optional with an extern declaration.

Initialization of an external variable goes only with the definition.