Understanding Variable Scope

scope- refers to the extent to which a variable is visible/accessible.

important because structured approach divides program into **INDEPENDENT** functions

external variable- variable defined outside of any function- also called global variable

use external variables rarely- violate principle modular independence

only make external when all or most of the program's functions need access

good practice: *extern type name*

local variables are **automatic** by default. that is they are created anew each time a function is called and destroyed when complete

to retain value of local variable between function calls use **static**

local scope- variable that is contained in a functions heading parameter

parameter variables always start with the value passed to the corresponding argument

ordinary external variable visible to other files static external variable only visible to functions in own file below point of definition

register variable- store variable in cpu register. suggestion if possible. **register** keyword. good for variables frequently used such as counter variable in loop

only be used simple numeric variables not arrays or structures

programs defined in main() function created when program begins and die when it ends

no difference automatic and static variables in main() function

can define variables local to any block {}

Guidelines for picking storage classes (chart pg 297)

```
1. give each variable an automatic local storage class to begin with
2. if the variable will be manipulated frequently provide *register* keyword to definition
3. in functions other than main() make a variable static if its value must
be retained between calls to the function
4. if a variable is used by most or all of the program's functions then
define it with an external storage class
```

global variables take up memory as long as program is running

if local variable and global variable have same name program will ignore global over local