Which to Use When

Dennis Ritchie has made available to the C programmer a number of storage classes with varying features, believing that the programmer is in a best position to decide which one of these storage classes is to be used when. We can make a few ground rules for usage of different storage classes in different programming situations with a view to:

(a) economise the memory space consumed by the variables
(b) improve the speed of execution of the program

The rules are as under:

- Use **static** storage class only if you want the value of a variable to persist between different function calls.
- Use **register** storage class for only those variables that are being used very often in a program. Reason is, there are very few CPU registers at our disposal and many of them might be busy doing something else. Make careful utilization of the scarce resources. A typical application of **register** storage class is loop counters, which get used a number of times in a program.
- Use **extern** storage class for only those variables that are being used by almost all the functions in the program. This would avoid unnecessary passing of these variables as arguments when making a function call. Declaring all the variables as **extern** would amount to a lot of wastage of memory space because these variables would remain active throughout the life of the program.
- If you don't have any of the express needs mentioned above, then use the **auto** storage class. In fact most of the times we end up using the **auto** variables, because often it so happens that once we have used the variables in a function we don't mind loosing them.

Summary

(a) We can use different variations of the primary data types, namely **signed** and **unsigned char**, **long** and **short int**, **float**, **double** and **long double**. There are different format specifications for all these data types when they are used in **scanf( )** and **printf( )** functions.

(b) The maximum value a variable can hold depends upon the number of bytes it occupies in memory.

(c) By default all the variables are **signed**. We can declare a variable as **unsigned** to accommodate greater value without increasing the bytes occupied.

(d) We can make use of proper storage classes like **auto**, **register**, **static** and **extern** to control four properties of the variable—storage, default initial value, scope and life.