Part I

ANSI C

“Never give in, never give in, never give in—in nothing great or small, large or petty—never give in except to convictions of honor and good sense”

—Winston Churchill

4

“A good person gives life to others.”

ANSI C - Prelude

When C language was developed, it took its popularity and many changes have been done on the language by other people. It necessitates the need for a good standard for C. Thus in 1983 American National Standards Institute (ANSI) established a committee to “standardize” the C language. The main objective of ANSI was to provide portability to C. (*Portability* is nothing but how far your code is portable, i.e. how far your code can be transported between different machines & different operating systems). The result of the committee’s work was completed by the end of 1988. And the result is the ANSI standard or ANSI C.

Note

As Part I, fully concentrates on ANSI C, choose ANSI C from your Turbo C++3.0 IDE (Options > Compiler > Source > ANSI C) to let your standard to ANSI C.

Thus the word ‘C’ directly or indirectly refers to ANSI C. Indian Programmers very often misunderstand that *DOS programming* is *C programming*. There is a vast difference between DOS programming and C programming. C programming always refers to ANSI C standard.

ANSI C was accepted by ISO too. Thus ANSI C is the international standard for C.

Caution

ANSI C does not have `getch( ), dos.h` and other DOS based functions. If you are not sure about the functions, place the cursor over the function and press Ctrl+F1 and check the documentation, whether it is acceptable in ANSI standard or not.

4.1 Myth & Mistakes

Q: *Is there any difference between “C programming” and “DOS programming”?*

A: Yes. There is a lot of difference between the two. The term “C programming” always refers to ANSI C. The main objective of ANSI C is to provide portable C code. If you write a code that can run *only on DOS*, then it is a DOS program (not C program) and you will be referred as “DOS Programmer”!

You have to understand that C (ANSI C) programs are 100% portable and those programs can run on any operating systems and on any machines.

So if you develop a C program that can run only on DOS, it is DOS programming. The right term in this context is “*DOS programming with C*”.

Q: *I am working with UNIX. Am I working with ANSI standard?*

A: Yes. As far as I know all the UNIX based compilers follow ANSI standards. But the DOS or Windows based compilers use their own standards.

Q: *Many people refer "C is Sea". Is C big enough with number of functions?*

A: No. According to K&R ("K&R" and "White book" are the nick names for "The C Programming Language", the book written by Kernighan and Ritchie) "C is not a big language…C is pleasant, expressive…" But we can widen the C library with our own functions.

Q: *Are all software being written in C?*

A: May not be. K & R says that all UNIX application programs are written in C. But other operating system developers haven't said so. According to me most of the DOS based applications are written in Assembly than in C. So this question doesn't have any valid answer.

4.2 Tips for better Programming

4.2.1 Coding Style

Readability is a must for every program. So I ask you to use WAR coding style. The rule(j) of WAR coding style, which says, "Don't use more than one `return( )` on a single function" may be little bit hard for you. But if you code with WAR coding style, your code would get more readability than with other coding styles.

Usually programmers uses the following style for `strcmp( )` function:

```
/* strcmp( ) without WAR coding style */
int strcmp(char *s, char *t)
{
   while(*s==*t)
   {
      if(*s=='\0')
         return(0);
      ++s;
      ++t;
   }
   return(*s-*t); /* more than one return statement */
}
```

But if you code with WAR coding rules your code will be more readable. The following code fragments use WAR coding style for the same `strcmp( ).`

```
/* strcmp( ) with WAR coding style */
int strcmp( char *s, char *t )
{
       int n;
```

```
        while ((n = *s - *t++) == 0 && *s++)
             ;
        return( n );
}
```

Now you might have found that how far WAR coding style is better than other coding styles.

4.2.2 Boolean Variables

In C, ‘0’ refers to ‘False’ and any other number refers to ‘True’. But however, we don’t have separate data type for Boolean. But it is wise to have Boolean, for better programming.

Boolean can be defined like:

Version 1

```
enum BOOLEAN
{
      FALSE, TRUE
};
```

Version 2

```
enum BOOLEAN
{
      FALSE = 0,TRUE = 1
};
```

Version 3

```
enum BOOLEAN
{
      FALSE=0, TRUE
};
```

Version 4

```
enum BOOLEAN
{
      TRUE=1,FALSE=0
};
```

All the above four versions use `enum.` But programmers rarely use `enum.`

Some people use

Version 5

```
typedef char BOOLEAN;
#define TRUE  (1)
#define FALSE (0)
```

Version 5 uses `typedef` to define `BOOLEAN`. It is efficient in terms of space (memory) to use `char`. But `char` is slower than `int`.

So let’s see another version.

Version 6

```
typedef int BOOLEAN
# define TRUE  (1)
# define FALSE (0)
```

Version 6 uses int for `BOOLEAN`. Since `int` is the fastest data type in C, version 6 is better than any other implementations. Also `FALSE` & `TRUE` are defined with macro `#define`. So it is the fastest implementation of `BOOLEAN.` So I recommend you to use version 6 for `BOOLEAN` implementation.

4.2.3 How to code better?

Beginners usually ask the question: How to develop programming skills? According to me, the programs related to ‘Calendar’ will help you to develop programming skills. You must remember to use all features of the language when you program.

The following points will help you to program better:

a) Your code should be efficient. ‘Efficient’ refers to less in code size and faster in execution.
b) Your code should have good readability.
c) Your code should use all the good features of the language

Try to rewrite your code. It will help you to reduce the size of the code and to increase readability.