C Programming

Trainer : Smita Kadam

Email ID : smita@sunbeaminfo.com

Bitwise Operators

- Bitwise Binary Operators

&	bitwise AND operator
|	bitwise OR operator
^	bitwise XOR operator
<<	bitwise left shift operator
>>	bitwise right operator

- Bitwise Unary Operators

~	bitwise complement operator

Bitwise Operators

&	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0
	0	0	0	0	0	1	1	1
	0	0	0	0	1	1	0	1
	0	0	0	0	0	1	0	1

7 & 13 = 5

|	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0
	0	0	0	0	0	1	1	1
	0	0	0	0	1	1	0	1
	0	0	0	0	1	1	1	1

7 | 13 = 15

^	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0
	0	0	0	0	0	1	1	1
	0	0	0	0	1	1	0	1
	0	0	0	0	1	0	1	0

7 ^ 13 = 10

~	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0
	0	0	0	0	0	1	1	1
(n=7)	1	1	1	1	1	0	0	0
	1	1	1	1	1	0	0	0
	0	0	0	0	0	1	1	1
								1
-(n-1)	0	0	0	0	1	0	0	0

~7 = -8

2's Compliment number + 1

13 << 1	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0
	0	0	0	0	1	1	0	1
0	0	0	0	1	1	0	1	0
	2^4 + 2^3 +~~2^2~~ +2^1+~~2^0~~ = 26							

13<<1 = 26

13 >> 1	B^7	B^6	B^5	B^4	B^3	B^2	B^1	B^0	
	0	0	0	0	1	1	0	1	
	0	0	0	0	0	1	1	0	1
	2^2 + 2^1 +~~2^0~~ = 6								

13>>1 = 6

Decision Control - switch case

Syntax :

```
switch(<expression>)
{
        case <integer constant>:
                                <statements>
                                break;
        case <integer constant>:
                                <statements>
                                break;
        …
        default:
                                <statements>
                                break;
}
```

Decision Control - switch case

Points To Note:

1. Each case should be followed by integer constant.

2. Can not add duplicate cases.

3. Use of break is suggested in each case at the last. Else execution control is given to next case even though next case is not satisfied.

4. Use of default case is optional.

5. Sequence of case does not matter.

typedef

Syntax:

typedef <existing data type> <another name/alias> ;

Points To Note:

1. Helps to give another name to existing data type.

2. Improves readability of source code.
3. Helps to port code across multiple architecture/platforms.
4. Helps to simplify complicated declarations

User Defined Data Type : enum

Syntax:

enum <tag name> {[<enumated fields>,..]} ; e.g enum colors{RED,BLUE,GREEN};

1. Helps to define new data type.
2. Collection of enumerated fields.
3. Each enumerated field represent integer constant and also can be initialised with integer constant.
4. Helps to improve readability of source code.
5. If we do not initialise first field by default it represents integer 0 value. And next fields carry step 1 value ahead of previous.

Thank you!