Chapter 4
Operators

- The symbols which are used to perform logical and mathematical operations in a C program are called C operators.
- These C operators join individual constants and variables to form expressions.
- Operators, functions, constants and variables are combined together to form expressions.
- Consider the expression A + B * 5. where, +, * are operators, A, B are variables, 5 is constant and A + B * 5 is an expression.

TYPES OF C OPERATORS:

C language offers many types of operators. They are,

1. Arithmetic operators
2. Assignment operators
3. Relational operators
4. Logical operators
5. Bit wise operators
6. Conditional operators (ternary operators)
7. Increment/decrement operators
8. Special operators

CONTINUE ON TYPES OF C OPERATORS:

Click on each operator name below for detailed description and example programs.

Types of Operators	Description
Arithmetic_operators	These are used to perform mathematical calculations like addition, subtraction, multiplication, division and modulus
Assignment_operators	These are used to assign the values for the variables in C programs.
Relational operators	These operators are used to compare the value of two variables.
Logical operators	These

	operators are used to perform logical operations on the given two variables.
Bit wise operators	These operators are used to perform bit operations on given two variables.
Conditional (ternary) operators	Conditional operators return one value if condition is true and returns another value is condition is false.
Increment/decrement operators	These operators are used to either increase or decrease the value of the variable by one.
Special operators	&, *, sizeof( ) and ternary operators.

ARITHMETIC OPERATORS IN C:

C Arithmetic operators are used to perform mathematical calculations like addition, subtraction, multiplication, division and modulus in C programs.

Arithmetic Operators/Operation	Example
+ (Addition)	A+B
– (Subtraction)	A-B
* (multiplication)	A*B
/ (Division)	A/B
% (Modulus)	A%B

EXAMPLE PROGRAM FOR C ARITHMETIC OPERATORS:

In this example program, two values “40” and “20” are used to perform arithmetic operations such as addition, subtraction, multiplication, division, modulus and output is displayed for each operation.

```c
#include <stdio.h>

int main()
{
   int a=40,b=20, add,sub,mul,div,mod;
   add = a+b;
   sub = a-b;
   mul = a*b;
   div = a/b;
   mod = a%b;
   printf("Addition of a, b is : %d\n", add);
   printf("Subtraction of a, b is : %d\n", sub);
   printf("Multiplication of a, b is : %d\n", mul);
   printf("Division of a, b is : %d\n", div);
   printf("Modulus of a, b is : %d\n", mod);
}
```

OUTPUT:

```
Addition of a, b is : 60
Subtraction of a, b is : 20
Multiplication of a, b is : 800
Division of a, b is : 2
Modulus of a, b is : 0
```

ASSIGNMENT OPERATORS IN C:

In C programs, values for the variables are assigned using assignment operators.

- For example, if the value “10” is to be assigned for the variable “sum”, it can be assigned as “sum = 10;”
- There are 2 categories of assignment operators in C language. They are,
  1. Simple assignment operator (Example: =)
  2. Compound assignment operators (Example: +=, -=, *=, /=, %=, &=, ^=)

Operators	Example/Description
=	sum = 10; 10 is assigned to variable sum
+=	sum += 10; This is same as sum = sum + 10
-=	sum -= 10; This is same as sum = sum – 10
*=	sum *= 10;

	This is same as sum = sum * 10
/=	sum /= 10; This is same as sum = sum / 10
%=	sum %= 10; This is same as sum = sum % 10
&=	sum&=10; This is same as sum = sum & 10
^=	sum ^= 10; This is same as sum = sum ^ 10

EXAMPLE PROGRAM FOR C ASSIGNMENT OPERATORS:

- In this program, values from 0 – 9 are summed up and total "45" is displayed as output.
- Assignment operators such as "=" and "+=" are used in this program to assign the values and to sum up the values.

```c
# include <stdio.h>

int main()
{
int Total=0,i;
for(i=0;i<10;i++)
{
Total+=i; // This is same as Total = Toatal+i
}
printf("Total = %d", Total);
}
```

OUTPUT:

```
Total = 45
```

RELATIONAL OPERATORS IN C:

Relational operators are used to find the relation between two variables. i.e. to compare the values of two variables in a C program.

Operators	Example/Description
>	x > y (x is greater than y)
<	x < y (x is less than y)
>=	x >= y (x is greater than or equal to y)
<=	x <= y (x is less than or

	equal to y)
==	x == y (x is equal to y)
!=	x != y (x is not equal to y)

EXAMPLE PROGRAM FOR RELATIONAL OPERATORS IN C:

- In this program, relational operator (==) is used to compare 2 values whether they are equal are not.
- If both values are equal, output is displayed as " values are equal". Else, output is displayed as "values are not equal".
- Note : double equal sign (==) should be used to compare 2 values. We should not single equal sign (=).

```c
#include <stdio.h>

int main()
{
    int m=40,n=20;
    if (m == n)
    {
        printf("m and n are equal");
    }
    else
    {
        printf("m and n are not equal");
    }
}
```

OUTPUT:

```
m and n are not equal
```

LOGICAL OPERATORS IN C:

These operators are used to perform logical operations on the given expressions.

- There are 3 logical operators in C language. They are, logical AND (&&), logical OR (||) and logical NOT (!).

Operators	Example/Description
&& (logical AND)	(x>5)&&(y<5) It returns true when both conditions are true
|| (logical OR)	(x>=10)||(y>=10) It returns true when at-least one of the condition is true
! (logical NOT)	!((x>5)&&(y<5)) It reverses the state of the operand "((x>5) && (y<5))" If "((x>5) && (y<5))" is true, logical NOT operator makes it false

EXAMPLE PROGRAM FOR LOGICAL OPERATORS IN C:

```c
#include <stdio.h>

int main()
{
   int m=40,n=20;
   int o=20,p=30;
   if (m>n && m !=0)
   {
      printf("&& Operator : Both conditions are true\n");
   }
   if (o>p || p!=20)
   {
      printf("|| Operator : Only one condition is true\n");
   }
   if (!(m>n && m !=0))
   {
      printf("! Operator : Both conditions are true\n");
   }
   else
   {
      printf("! Operator : Both conditions are true. " \
      "But, status is inverted as false\n");
   }
}
```

OUTPUT:

```
&& Operator : Both conditions are true
|| Operator : Only one condition is true
! Operator : Both conditions are true. But, status is inverted as false
```

- In this program, operators (&&, || and !) are used to perform logical operations on the given expressions.
- **&& operator** – "if clause" becomes true only when both conditions (m>n and m! =0) is true. Else, it becomes false.
- || **Operator** – "if clause" becomes true when any one of the condition (o>p || p!=20) is true. It becomes false when none of the condition is true.
- **! Operator** – It is used to reverses the state of the operand.
- If the conditions (m>n && m!=0) is true, true (1) is returned. This value is inverted by "!" operator.
- So, "! (m>n and m! =0)" returns false (0).

BIT WISE OPERATORS IN C:

These operators are used to perform bit operations. Decimal values are converted into binary values which are the sequence of bits and bit wise operators work on these bits.

- Bit wise operators in C language are & (bitwise AND), | (bitwise OR), ~ (bitwise NOT), ^ (XOR), << (left shift) and >> (right shift).

TRUTH TABLE FOR BIT WISE OPERATION & BIT WISE OPERATORS:

x	y	x|y	x&y	x^y
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

BELOW ARE THE BIT-WISE OPERATORS AND THEIR NAME IN C LANGUAGE.

1. & – Bitwise AND
2. | – Bitwise OR
3. ~ – Bitwise NOT
4. ^ – XOR
5. << – Left Shift
6. >> – Right Shift

Consider x=40 and y=80. Binary form of these values are given below.

x = 00101000
y= 01010000

All bit wise operations for x and y are given below.

1. x&y = 00000000 (binary) = 0 (decimal)
2. x|y = 01111000 (binary) = 120 (decimal)
3. ~x = 11111111111111111111111111 11111111111111111111111111010111 = -41 (decimal)
4. x^y = 01111000 (binary) = 120 (decimal)
5. x << 1 = 01010000 (binary) = 80 (decimal)
6. x >> 1 = 00010100 (binary) = 20 (decimal)

NOTE:

- **Bit wise NOT :** Value of 40 in binary is 00000000000000000000000000000000 00000000000000000010100000000000. So, all 0's are converted into 1's in bit wise NOT operation.
- **Bit wise left shift and right shift :** In left shift operation "x << 1 ", 1 means that the bits will be left shifted by one place. If we use it as "x << 2 ", then, it means that the bits will be left shifted by 2 places.

EXAMPLE PROGRAM FOR BIT WISE OPERATORS IN C:

In this example program, bit wise operations are performed as shown above and output is displayed in decimal format.

```c
#include <stdio.h>

int main()
{
   int m = 40,n = 80,AND_opr,OR_opr,XOR_opr,NOT_opr ;
   AND_opr = (m&n);
   OR_opr = (m|n);
   NOT_opr = (~m);
   XOR_opr = (m^n);
   printf("AND_opr value = %d\n",AND_opr );
   printf("OR_opr value = %d\n",OR_opr );
   printf("NOT_opr value = %d\n",NOT_opr );
   printf("XOR_opr value = %d\n",XOR_opr );
   printf("left_shift value = %d\n", m << 1);
   printf("right_shift value = %d\n", m >> 1);
}
```

OUTPUT:

```
AND_opr value = 0
OR_opr value = 120
NOT_opr value = -41
XOR_opr value = 120
left_shift value = 80
right_shift value = 20
```

CONDITIONAL OPERATORS IN C:

- Conditional operators return one value if condition is true and returns another value is condition is false.
- This operator is also called as ternary operator.

Syntax : (Condition? true_value: false_value);

Example : (A > 100 ? 0 : 1);

- In above example, if A is greater than 100, 0 is returned else 1 is returned. This is equal to if else conditional statements.

EXAMPLE PROGRAM FOR CONDITIONAL/TERNARY OPERATORS IN C:

```c
#include <stdio.h>

int main()
{
   int x=1, y ;
   y = ( x ==1 ? 2 : 0 ) ;
   printf("x value is %d\n", x);
   printf("y value is %d", y);
}
```

OUTPUT:

```
x value is 1
y value is 2
```

Increment/decrement Operators in C:

- Increment operators are used to increase the value of the variable by one and decrement operators are used to decrease the value of the variable by one in C programs.
- Syntax:

  Increment operator: ++var_name; (or) var_name++;

  Decrement operator: – -var_name; (or) var_name – -;
- Example:

  Increment operator : ++ i ; i ++ ;

  Decrement operator : – – i ; i – – ;

EXAMPLE PROGRAM FOR INCREMENT OPERATORS IN C:

In this program, value of "i" is incremented one by one from 1 up to 9 using "i++" operator and output is displayed as "1 2 3 4 5 6 7 8 9".

```c
//Example for increment operators

#include <stdio.h>
int main()
{
     int i=1;
     while(i<10)
     {
         printf("%d ",i);
         i++;
     }
}
```

OUTPUT:

```
1 2 3 4 5 6 7 8 9
```

EXAMPLE PROGRAM FOR DECREMENT OPERATORS IN C:

In this program, value of "I" is decremented one by one from 20 up to 11 using "i–" operator and output is displayed as "20 19 18 17 16 15 14 13 12 11".

```c
//Example for decrement operators

#include <stdio.h>
int main()
{
    int i=20;
    while(i>10)
    {
        printf("%d ",i);
        i--;
    }
}
```

OUTPUT:

```
20 19 18 17 16 15 14 13 12 11
```

DIFFERENCE BETWEEN PRE/POST INCREMENT & DECREMENT OPERATORS IN C:

Below table will explain the difference between pre/post increment and decrement operators in C programming language.

Operator	Operator/Description
Pre increment operator (++i)	value of i is incremented before assigning it to the variable i
Post increment operator (i++)	value of i is incremented after assigning it to the variable i
Pre decrement operator (–i)	value of i is decremented before assigning it to the variable i
Post decrement operator (i–)	value of i is decremented after assigning it to variable i

EXAMPLE PROGRAM FOR PRE – INCREMENT OPERATORS IN C:

```c
//Example for increment operators

#include <stdio.h>
int main()
{
            int i=0;
            while(++i < 5 )
            {
                  printf("%d ",i);
            }
            return 0;
}
```

OUTPUT:

```
1 2 3 4
```

- Step 1 : In above program, value of "i" is incremented from 0 to 1 using pre-increment operator.
- Step 2 : This incremented value "1" is compared with 5 in while expression.
- Step 3 : Then, this incremented value "1" is assigned to the variable "i".
- Above 3 steps are continued until while expression becomes false and output is displayed as "1 2 3 4".

EXAMPLE PROGRAM FOR POST – INCREMENT OPERATORS IN C:

```c
#include <stdio.h>
int main()
{
            int i=0;
            while(i++ < 5 )
            {
                  printf("%d ",i);
            }
            return 0;
}
```

OUTPUT:

```
1 2 3 4 5
```

- Step 1 : In this program, value of i "0" is compared with 5 in while expression.
- Step 2 : Then, value of "i" is incremented from 0 to 1 using post-increment operator.
- Step 3 : Then, this incremented value "1" is assigned to the variable "i".
- Above 3 steps are continued until while expression becomes false and output is displayed as "1 2 3 4 5".

EXAMPLE PROGRAM FOR PRE – DECREMENT OPERATORS IN C:

```c
#include <stdio.h>
int main()
{
            int i=10;
            while(--i > 5 )
            {
                  printf("%d ",i);
            }
            return 0;
}
```

OUTPUT:

```
9 8 7 6
```

- Step 1 : In above program, value of “i” is decremented from 10 to 9 using pre-decrement operator.
- Step 2 : This decremented value “9” is compared with 5 in while expression.
- Step 3 : Then, this decremented value “9” is assigned to the variable “i”.
- Above 3 steps are continued until while expression becomes false and output is displayed as “9 8 7 6”.

EXAMPLE PROGRAM FOR POST – DECREMENT OPERATORS IN C:

```c
#include <stdio.h>
int main()
{
        int i=10;
        while(i-- > 5 )
        {
            printf("%d ",i);
        }
        return 0;
}
```

OUTPUT:

```
9 8 7 6 5
```

- Step 1 : In this program, value of i "10" is compared with 5 in while expression.
- Step 2 : Then, value of "i" is decremented from 10 to 9 using post-decrement operator.
- Step 3 : Then, this decremented value "9" is assigned to the variable "i".
- Above 3 steps are continued until while expression becomes false and output is displayed as "9 8 7 6 5".

SPECIAL OPERATORS IN C:

Below are some of the special operators that the C programming language offers.

Operators	Description
&	This is used to get the address of the variable. Example : &a will give address of a.
*	This is used as pointer to a variable. Example : * a where, * is pointer to the variable a.
Sizeof ()	This gives the size of the variable. Example : size of (char) will give us 1.

EXAMPLE PROGRAM FOR & AND * OPERATORS IN C:

In this program, “&” symbol is used to get the address of the variable and “*” symbol is used to get the value of the variable that the pointer is pointing to. Please refer **C – pointer** topic to know more about pointers.

```c
#include <stdio.h>
int main()
{
int *ptr, q;
q = 50;
/* address of q is assigned to ptr */
ptr = &q;
/* display q's value using ptr variable */
printf("%d", *ptr);
return 0;
}
```

OUTPUT:

```
50
```

EXAMPLE PROGRAM FOR SIZEOF() OPERATOR IN C:

sizeof() operator is used to find the memory space allocated for each C data types.

```c
#include <stdio.h>
#include <limits.h>

int main()
{
int a;
char b;
float c;
double d;
printf("Storage size for int data type:%d \n",sizeof(a));
printf("Storage size for char data type:%d \n",sizeof(b));
printf("Storage size for float data type:%d \n",sizeof(c));
printf("Storage size for double data type:%d\n",sizeof(d));
return 0;
}
```

OUTPUT:

```
Storage size for int data type:4
Storage size for char data type:1
Storage size for float data type:4
Storage size for double data type:8
```