**PRE INCREMENT AND POST INCREMENT OPERATOR**

**1)Pre-increment operator**: A pre-increment operator is used to increment the value of a variable before using it in a expression. In the Pre-Increment, value is first incremented and then used inside the expression.

The Pre-increment operator increases the value of the variable by 1 before using it in the expression, i.e. the value is incremented before the expression is evaluated.

**Syntax:** 

a = ++x;

Here, if the value of ‘x’ is 10 then value of ‘a’ will be 11 because the value of ‘x’ gets modified before using it in the expression.

**Write a program to demonstrate prefix increment**

#include<stdio.h>

Void main()

{

Int a,result;

a=5;

result=++a;

printf(“result is%d”,result);

printf(“\n value of a:%d”,a);

}

**Output:**

Result=6

Value of a=6

**2)Post-increment operator**: A post-increment operator is used to increment the value of variable after executing expression completely in which post increment is used. In the Post-Increment, value is first used in a expression and then incremented.

The Post-increment operator increases the value of the variable by 1 after using it in the expression, i.e. the value is incremented after the expression is evaluated.

**Syntax:** 

a = x++;

Here, suppose the value of ‘x’ is 10 then value of variable ‘b’ will be 10 because old value of ‘x’ is used.

**Write a program to demonstrate postfix increment**

#include<stdio.h>

Void main()

{

Int a,result;

a=5;

result=a++;

printf(“result is%d”,result);

printf(“\n value of a:%d”,a);

}

**Output:**

Result=5

Value of a=6

**Difference between pre-increment and post-increment operators**

Since both are used to increase the value of the variable by 1. The pre-increment increased the value before the expression is evaluated and the post-increment increases the value after the expression is evaluated.

**Program to demonstrate the use of pre and post increment operators**

#include<stdio.h>

**int** main()

{

**int** A = 10, B = 20, C = 0;

C = A++ + ++B \* 10 + B++;

Printf(“%d %d %d”,A,B,C);

**return** 0;

}

**Output:**

11, 22, 241

**Explanation:**

Based on the above discussion, the express will be evaluated like this,

Values are,

A = 10

B = 20

C = 0

Expression is,

C = A++ + ++B \* 10 + B++;

C = 10 + 21 \* 10 + 21

= 10 +210 +21

= 241

Finally, the values of A, B and C will be,

A = 11

B = 22

C = 241

# Post-decrement and Pre-decrement Operator:

**In pre-decrement**, first the value of the variable is decremented after that the assignment or other operations are carried Pre-decrement unary operator is used to decrease the value of variable by one before using in the expression. In the pre-decrement operator concern value is first decremented and then it used inside the expression with final updated value.

**Note:**  
a- -;  
– -a;  
a = a – 1;  
a -= 1;

are all same if used independently.

### Pre-decrement Operator: C Program

1. #include < stdio.h >
3. **int** main()
4. {
5. **int** a = 10, b;
7. b = --a;
9. printf("b = %d\n\n", b);
10. printf("a = %d\n", a);
12. **return** 0;
13. }

**Output:**  
b = 9  
a = 9

Here first the value of a decrements and then is assigned to variable b. So both a and b value will be 9.

**ANOTHER PRE DECREMENT PROGRAM**

1. #include < stdio.h >
3. **int** main()
4. {
5. **int** a = 10;
7. printf("a = %d\n\n", --a);
8. printf("a = %d\n", a);
10. **return** 0;
11. }

**Output:**  
a = 9  
a = 9

Here in the first printf statement a value gets decremented and then printed out.

**Post-decrement unary operator** is used to decrement the value of variable as soon as after executing expression completely in which post decrement is used individually. In the post-decrement, value is first used in an expression and then decremented.

. In post-decrement, first assignment or other operations occur, after that the value of the variable is decremented.

### Post-decrement Operator: C Program

1. #include < stdio.h >
3. **int** main()
4. {
5. **int** a = 10, b;
7. b = a--;
9. printf("b = %d\n\n", b);
10. printf("a = %d\n", a);
12. **return** 0;
13. }

**Output:**  
b = 10  
a = 9

Here first value of a(i.e., 10) is assigned to b and then value of a is decremented. So b = 10 and a = 9 is printed.

**ANOTHER POST DECREMENT PROGRAM**

1. #include < stdio.h >
3. **int** main()
4. {
5. **int** a = 10;
7. printf("a = %d\n\n", a--);
8. printf("a = %d\n", a);
10. **return** 0;
11. }

**Output:**  
a = 10  
a = 9

Here in the first printf statement a value gets printed after that its value gets decremented, which is shown in second printf statement

**DIFFERENCE BETWEEN PRE AND POST DECREMENT OPERATOR**

**Pre-decrement (--i)** − Before assigning the value to the variable, the value is decremented by one.

**Post-decrement (i--)** − After assigning the value to the variable, the value is decremented.

The following is the syntax of pre and post increment.

--variable\_name; // Pre-increment

variable\_name--; // Post-increment

Here,

**variable\_name** − Any name of the variable given by user.