# 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);

}

What is the output of the above program?

#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");

   }

}

What is the output of the above program?

#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);

}

Syntax     :        (Condition? true\_value: false\_value);

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

#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);

}

* Syntax:  
  Increment operator: ++var\_name; (or) var\_name++;  
  Decrement operator: – -var\_name; (or) var\_name – -;
* Example:  
  Increment operator :  ++ i ;    i ++ ;  
  Decrement operator :  – – i ;   i – – ;

#include <stdio.h>

**int** main()

{

**int** i=0;

**while**(++i < 5 )

           {

**printf**("%d ",i);

           }

**return** 0;

}

#include <stdio.h>

**int** main()

{

**int** i=0;

**while**(i++ < 5 )

           {

**printf**("%d ",i);

           }

**return** 0;

}

operator Precedence and Associativity in C

**Operator precedence** determines which operator is performed first in an expression with more than one operators with different precedence.

|  |  |  |
| --- | --- | --- |
| **Category** | **Operator** | **Associativity** |
| Postfix | () [] -> . ++ - - | Left to right |
| Unary | + - ! ~ ++ - - (type)\* & sizeof | Right to left |
| Multiplicative | \* / % | Left to right |
| Additive | + - | Left to right |
| Shift | << >> | Left to right |
| Relational | < <= > >= | Left to right |
| Equality | == != | Left to right |
| Bitwise AND | & | Left to right |
| Bitwise XOR | ^ | Left to right |
| Bitwise OR | | | Left to right |
| Logical AND | && | Left to right |
| Logical OR | || | Left to right |
| Conditional | ?: | Right to left |
| Assignment | = += -= \*= /= %=>>= <<= &= ^= |= | Right to left |
| Comma | , | Left to right |

#include <stdio.h>

main() {

int a = 20;

int b = 10;

int c = 15;

int d = 5;

int e;

e = (a + b) \* c / d; // ( 30 \* 15 ) / 5

printf("Value of (a + b) \* c / d is : %d\n", e );

e = ((a + b) \* c) / d; // (30 \* 15 ) / 5

printf("Value of ((a + b) \* c) / d is : %d\n" , e );

e = (a + b) \* (c / d); // (30) \* (15/5)

printf("Value of (a + b) \* (c / d) is : %d\n", e );

e = a + (b \* c) / d; // 20 + (150/5)

printf("Value of a + (b \* c) / d is : %d\n" , e );

return 0;

}

1. Write a C program to perform all the operations of a calculator.
2. Write a c Program to find the sum of N natural numbers
3. Write a C program to find the prime numbers from ‘ N’ numbers