

[Home](#) / [Tutorial](#) / [C++](#) / [Operator Bitwise](#)

Operator Bitwise

Posted on Januari 23, 2018 | Last Modified Januari 23, 2018



Operator bitwise adalah operasi matematika yang mengoperasikan suatu nilai dalam bilangan biner. Operator bitwise akan bekerja dengan mengubah bilangan bulat menjadi bilangan biner, setelah itu akan melakukan operasi berdasarkan operator bitwise yang digunakan. Dan pada akhirnya nilai tersebut akan diubah menjadi bilangan bulat kembali.

Operator	Asm equivalent	Keterangan
&	AND	Bitwise DAN
	OR	Bitwise OR Inklusif
^	XOR	Bitwise OR Eksklusif
~	NOT	Akan membalikan nilai
<<	SHL	Penggeseran bit ke kiri
>>	SHR	Penggeseran Bit ke Kanan

Macam-macam Operator Bitwise

Shift Left (<<)

Operator Bitwise Shift Left (<<) adalah operator yang akan menggeser nilai dalam bentuk bilangan biner ke kiri. Operator ini akan mengubah bilangan bulat menjadi bilangan biner lalu menggeser angka dari bilangan biner tersebut ke kiri dan setelah itu akan mengubahnya kembali ke bilangan bulat.

```
0000 0000 1100 1001 = 201
//menggeser 1 bit ke kiri, Dibagian kanan disisipkan 0, sebanyak
bit yang digeser
0000 0001 1001 0010 = 402
```

Bentuk Penulisan :

```
Nilai << jumlah;
```

Contoh Program :

```
#include <iostream>
using namespace std;

int main(){
    int x;

    cout<<"Masukan Nilai X = "; cin>>x;
    cout<<"Nilai Awal : "<<x<<endl;

    x = x << 1;
    cout<<"Hasil dari Geser 1 Bit Ke Kiri = "<<x<<endl;

    return 0;
}
```

Shift Right (>>)

Operator Bitwise Shift Right (>>) adalah operator yang akan mengeser nilai dalam bentuk bilangan biner ke kanan. Operator ini akan mengubah bilangan bulat menjadi bilangan biner setelah itu menggeser angka dari dari bilangan biner tersebut ke kanan dan pada akhirnya akan mengubahnya kembali ke bilangan bulat.

```
0000 0000 1100 1001 = 201
```

```
//digeser 1 bit ke kanan, Dibagian kanan disisipkan 0, sebanyak  
bit yang digeser dan pada bagian paling kiri akan hilang
```

```
0000 0000 0110 0100 = 100
```

Bentuk Penulisan :

```
Nilai >> jumlah;
```

Contoh Program :

```
#include <iostream>
using namespace std;

int main(){
    int x;

    cout<<"Masukan Nilai X = "; cin>>x;
    cout<<"Nilai Awal : "<<x<<endl;

    x = x >> 1;
    cout<<"Hasil dari Geser 1 Bit Ke Kanan = "<<x<<endl;

    return 0;
}
```

AND (&)

Operator Bitwise AND (&) adalah operator yang akan membandingkan dua operand dalam bentuk bilangan biner. Operator ini akan mengubah bilangan bulat menjadi bilangan biner lalu

membandingkannya dengan operand dari sisi lain, perbandingan akan berlangsung dari bit ke bit (angka ke angka), membandingkan berdasarkan sifat dari AND yaitu untuk menghasilkan nilai 1 (true) kedua operand harus bernilai 1(true) jika tidak akan menghasilkan nilai 0 (false). Dan setelah perbandingan selesai dilakukan, hasil dari perbandingan akan diubah ke bentuk bilangan bulat.

Tabel Operator Bitwise AND

Bit Operand 1	Bit Operand 2	Hasil Operand
1	1	1
1	0	0
0	1	0
0	0	0

Contoh :

```
1100 1001 = 201
0110 0100 = 100
----- AND
0100 0000 = 64
```

Contoh Program :

```
#include <iostream>
using namespace std;

int main(){
    int hasil, x, y;
    cout<<"Masukan Nilai X = "; cin>>x;
    cout<<"Masukan Nilai Y = "; cin>>y;
    hasil = x & y;
    cout<<"Hasil dari "<<x<<" & "<<y<<" = "<<hasil<<endl;
    return 0;
}
```

OR (|)

Operator Bitwise OR (|) adalah operator yang akan membandingkan dua operand dalam bentuk bilangan biner. Operator ini akan mengubah bilangan bulat menjadi bilangan biner lalu membandingkannya dengan operand dari sisi lain, perbandingan akan berlangsung dari bit ke bit (angka ke angka), membandingkan berdasarkan sifat dari OR yaitu untuk mendapatkan nilai 1 (true) maka salah satu atau semua operand harus bernilai 1 (true), jika semua operand bernilai 0 (false) maka akan mendapatkan nilai 0 (false). Dan setelah perbandingan selesai dilakukan, hasil dari perbandingan akan diubah ke bentuk bilangan bulat.

Tabel Operator Bitwise OR

Bit Operand 1	Bit Operand 2	Hasil Operand
1	1	1
1	0	1
0	1	1
0	0	0

Contoh :

```
1100 1001 = 201
0110 0100 = 100
-----OR
11101101 = 237
```

Contoh Program :

```
#include <iostream>
using namespace std;

int main(){
    int hasil, x, y;
    cout<<"Masukan Nilai X = "; cin>>x;
    cout<<"Masukan Nilai Y = "; cin>>y;
    hasil = x | y;
    cout<<"Hasil dari "<<x<<" | "<<y<<" = "<<hasil<<endl;
    return 0;
}
```

XOR (^)

Operator Bitwise XOR (^) merupakan singkatan dari “eXclusive OR” adalah operator yang akan membandingkan dua operand dalam bentuk bilangan biner. Operator ini akan mengubah bilangan bulat menjadi bilangan biner lalu membandingkannya dengan operand dari sisi lain, perbandingan akan berlangsung dari bit ke bit (angka ke angka), membandingkan berdasarkan sifat dari XOR yaitu untuk mendapatkan nilai 1 (true) maka kedua operand harus memiliki nilai yang berbeda, jika kedua operand memiliki nilai yang sama maka akan mendapatkan nilai 0 (false). Dan setelah perbandingan selesai dilakukan, hasil dari perbandingan akan diubah ke bentuk bilangan bulat.

Tabel Operator Bitwise XOR

Bit Operand 1	Bit Operand 2	Hasil Operand
1	1	0
1	0	1
0	1	1
0	0	0

Contoh :

```

1100 1001 = 201
0110 0100 = 100
-----XOR
1010 1101 = 137

```

Contoh Program :

```

#include <iostream>
using namespace std;
int main( )
{
    int a, b, x, y;

    cout<<"Masukan Nilai X = "; cin>>x;
    cout<<"Masukan Nilai Y = "; cin>>y;
    a = x ^ y;
    b = x ^ y ^ x;
    cout<<"Hasil dari "<<x<<" ^ "<<y<<" = "<<a<<endl;
    cout<<"Hasil dari "<<x<<" ^ "<<y<<" ^ "<<x<<" = "<<b<<endl;

    return 0;
}

```

NOT (~)

Operator Bitwise NOT (~) adalah operator bersifat unary yang akan membalikan nilai di dalam bentuk bilangan biner. Operator ini akan mengubah bilangan bulat menjadi bilangan biner lalu membalikan nilai dari bit ke bit (angka ke angka), jika bit tersebut memiliki nilai 1 (true) maka akan dibalik menjadi 0 (false) begitu pula sebaliknya. setelah operasi selesai maka hasil akhir akan di kembalikan ke bentuk bilangan bulat.

Tabel Operator Bitwise NOT

Bit Operand	Hasil
0	1
1	0

Contoh :

```
0000 1000 = 8  
menjadi  
1111 0111 = 247
```

Contoh Program :

```
#include <iostream>  
using namespace std;  
  
int main(){  
    int a, x;  
  
    cout<<"Masukan Nilai X = "; cin>>x;  
    a = ~x;  
  
    cout<<"Hasil dari ~ "<<x<<" = "<<a<<endl;  
    return 0;  
}
```

mungkin pada contoh-contoh di atas hasil penghitungan yang penulis contohkan tidak selalu akurat seperti apa yang telah anda coba dalam program, operator-operator di atas dalam melakukan operasi tergantung pada seberapa besar tipe data yang digunakan, beberapa contoh di atas penulis contohkan dengan menggunakan penyimpanan sebesar 8 bit.

Contoh Program :




```
#include <iostream>
using namespace std;

int main (){
    //Variabel
    int a=10, b=12;
    int hasilAND, hasilOR, hasilXOR, hasilNOT, hasilSHL,
    hasilSHR;

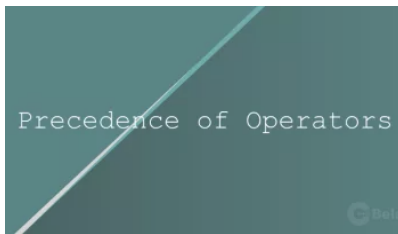
    //Operasi Bitwise
    hasilAND=a&b;
    hasilOR=a|b;
    hasilXOR=a^b;
    hasilNOT=~a;
    hasilSHL=a<<1;
    hasilSHR=a>>1;

    //Output
    cout<<hasilAND<<endl;
    cout<<hasilOR<<endl;
    cout<<hasilXOR<<endl;
    cout<<hasilNOT<<endl;
    cout<<hasilSHL<<endl;
    cout<<hasilSHR<<endl;

    return 0;
}
```

 Share Tweet Share Pin Share

Terkait



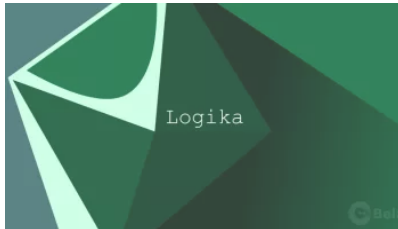
Precedence of Operators
Januari 23, 2018
dalam "C++"



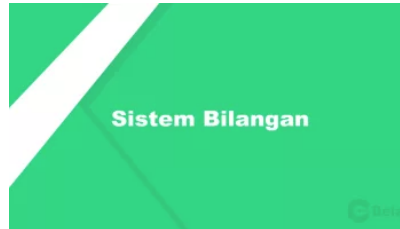
Operator Aritmetika
Januari 23, 2018
dalam "C++"



Operator
Januari 22, 2018
dalam "C++"



Operator Logika
Januari 23, 2018
dalam "C++"



Sistem Bilangan
Januari 22, 2018
dalam "Tutorial"



Operator Peningkatan dan
Penurunan
Januari 23, 2018
dalam "C++"