Jika sebelumnya kita telah belajar bagaimana menggunakan variabel dalam program sederhana kita, pada bab ini kita akan belajar macam-macam operator pada C++.

Berikut pengertian operator dalam bahasa pemrograman:

**Operator** adalah simbol khusus yang biasa dilibatkan dalam pembuatan program untuk melakukan suatu operasi matematika maupun manipulasi logika.

Pada dasarnya, ada tiga jenis operator dalam pemrograman C++:

* **Operator Unary** - operator yang dikenakan untuk satu buah nilai (operand).
* **Operator Binary** - operator yang dikenakan untuk dua buah nilai (operand).
* **Operator Ternary** - operator yang dikenakan untuk tiga buah nilai (operand)

Dengan catatan jika kita hanya melihat dari jumlah operand.

Berikut ini adalah contohnya:

- c //Binary

a + b //Unary

(a ? b : c) //Ternary

Namun yang akan kita bahas kali ini bukan itu, kita akan mempelajari macam-macam operator yang ada dalam pemrograman C++ berserta fungsi dan contoh programnya.

Berikut adalah daftar operator yang dapat anda jadikan acuan pembuatan program.

**Table Of Contents**

* [#1 Assignment Operators](https://kodedasar.com/operator-cpp/#1-assignment-operators)
* [#2 Arithmetic Operators](https://kodedasar.com/operator-cpp/#2-arithmetic-operators)
* [#3 Compound Assignment](https://kodedasar.com/operator-cpp/#3-compound-assignment)
* [#4 Increment and Decrement Operators](https://kodedasar.com/operator-cpp/#4-increment-and-decrement-operators)
* [#5 Relational and Comparison Operators](https://kodedasar.com/operator-cpp/#5-relational-and-comparison-operators)
* [#6 Logical Operators](https://kodedasar.com/operator-cpp/#6-logical-operators)
* [#7 Conditional Ternary Operator](https://kodedasar.com/operator-cpp/#7-conditional-ternary-operator)
* [#8 Comma Operator](https://kodedasar.com/operator-cpp/#8-comma-operator)
* [#9 Bitwise Operators](https://kodedasar.com/operator-cpp/#9-bitwise-operators)
* [#10 Operator Lainnya](https://kodedasar.com/operator-cpp/#10-operator-lainnya)
* [Menarik Kesimpulan](https://kodedasar.com/operator-cpp/#menarik-kesimpulan)

**#1 Assignment Operators**

Yang pertama kita akan mengenal *Assignment Operator*.

Sesuai namanya, operator ini berfungsi untuk memberikan nilai kepada variabel. Kita akan menggunakan = ( *tanda sama dengan* ) untuk memberi nilai.

Berikut cara penggunaanya:

nama = "Damas";

umur = 22;

Berikut contoh program sederha penggunaan *assignment operator* pada C++:

#include <iostream>

using namespace std;

int main ()

{

int a, b; // a:?, b:?

a = 10; // a:10, b:?

b = 4; // a:10, b:4

a = b; // a:4, b:4

b = 7; // a:4, b:7

cout << "a:";

cout << a;

cout << " b:";

cout << b;

}

Lalu outputnya adalah:

a:4 b:7

Mengapa hal itu bisa terjadi demikian?

Jika kita perhatkan, kita mendeklarasikan variabel a dan b. Lalu kita memberikan nilai 10 untuk a dan 4 untuk b pada awalnya. Lalu kita memberikan nilai b terhadap variabel a.

Perhatikan bagaimana sebuah variabel tidak terpengaruh oleh modifikasi akhir dari b, meskipun kita menyatakan a = b sebelumnya, anda pasti paham maksud saya.

Karena sebuah operasi akan selalu berlangsung dari kanan ke kiri.

Dan variabel hanya akan menyimpan data atau nilai terakhir yang diberikan.

*(Silahkan baca kembali:*[*Belajar Algoritma Pemrograman*](https://kodedasar.com/belajar-pascal/)*)*

**#2 Arithmetic Operators**

Yang kedua adalah operator aritmetika.

Operator ini digunakan untuk melakukan sebuah opearasi penghitungan atau aritmatika dalam bahasa pemrograman. Tugasnya untuk memecahkan masalah matematika.

Operator aritmatika yang sering digunakan dalam C++ adalah:

* + untuk penjumlahan
* - untuk pengurangan
* \* untuk perkalian
* / untuk pembagian
* % untuk sisa bagi (modulo)

Berikut adalah contoh penggunaanya:

x = 11 % 3;

Dan tentunya hasil opeasi diatas dapat ditebak, ya hasilnya adalah 2, karena disini kita mencari sisa bagi dimana hasil bagi 11 dan 3 = 3, sedangkan sisa baginya adalah 2.

**#3 Compound Assignment**

Operator ini memodifikasi nilai variabel saat ini dengan melakukan operasi di atasnya.

| **Operator** | **Keterangan** | **Contoh Penggunaanya** |
| --- | --- | --- |
| += | Penjumlahan | z += x sama dengan, z = z + x |
| -= | Pengurangan | z -= x sama dengan, z = z - x |
| \*= | Perkalian | z *= x sama dengan, z = z*x |
| /= | Pembagian | z /= x sama dengan, z = z / x |
| >>= | Right shift AND | z >>= 2 sama dengan, z = z >> 2 |
| <<= | Left shift AND | z <<= 2 sama dengan, z = z << 2 |
| &= | Bitwise AND | z &= 2 sama dengan, z = z & 2 |
| ^= | Bitwise exclusive OR | z ^= 2 sama dengan, z = z ^ 2 |

Berikut contoh program c++ menggunakan *Compound Assignment*.

#include <iostream>

using namespace std;

int main ()

{

int a, b=3;

a = b;

a += 2; // sama dengan a = a + 2

cout << a;

}

Intinya: "menggunakan dua operand dan memberikan hasilnya ke operand kiri".

**#4 Increment and Decrement Operators**

Kedua operator ini sangatlah penting:

Operator Increment (++) dan operator decrement (--) digunakan untuk meningkatkan atau mengurangi satu nilai yang tersimpan dalam sebuah variabel.

Maksudnya seperti ini:

x = x + 1 ;

y = y - 1 ;

Jika sebelumnya anda menulis kode seperti datas..

..maka akan sama hasilnya dengan kode berikut:

++ x ;

-- y ;

//atau

x ++ ;

y -- ;

**TIPS**: Operator Increment adalah operator yang digunakan untuk menaikan nilai variabel sebesar 1, sedangkan decrement digunakan untuk menurunkan nilai sebesar 1.

Berikut contoh program Increment dalam C++:

#include <iostream>

using namespace std;

int main ()

{

int a, b;

a = 3;

b = ++a;

cout << "a:" << a;

cout << " b:" << b;

}

Jika anda paham konsepnya anda pasti tahu hasil akhir dari variabel a akan bernilai 4, dan b bernilai 4 karena nilai dari b adalah nilai a yang telah ditingkatkan.

Kemudian nantinya anda akan mendengar istilah:

* Pre Increment (++c)
* dan Post Increment (c++).

Yang membedakannya adalah:

Jika Pre-increment akan menenambahkan nilai 1 sebelum operasi lain dijalankan, maka Post-increment akan menambahkan 1 nilai sesudah proses lain dijalankan.

Silahkan coba kode program berikut ini:

#include <iostream>

using namespace std;

int main(){

// Mendeklarasikan Variabel KD

int KD;

// Mengisi nilai kedalam Variabel KD

// dengan nilai 10

KD = 10;

// Melakukan Pre-Increment

cout<<"Nilai KD awal : "<<KD<<endl;

cout<<"Nilai ++MD : "<<++KD<<endl;

cout<<"Nilai KD : "<<KD<<endl;

// Mengubah nilai yang terdapat dalam variabel KD

// dengan nilai 20

KD = 20;

// Melakukan Post-Increment

cout<<"\nNilai KD awal : "<<KD<<endl;

cout<<"Nilai KD++ : "<<KD++<<endl;

cout<<"Nilai KD : "<<KD<<endl;

return 0;

}

Anda akan sering menemukannya operator tersebut dalam program looping.

**#5 Relational and Comparison Operators**

Operator selanjutnya adalah operator relasi - operator ini akan banyak kita gunakan dalam suatu statement bersyarat yang selalu menghasilkan nilai true atau false.

Nantinya kita akan sering menggunakan operator ini untuk menentukan kondisi.

| **Operator** | **Keterangan** |
| --- | --- |
| == | Sama dengan |
| != | Tidak sama dengan |
| > | Lebih dari |
| < | Kurang dari |
| >= | Lebih dari atau sama dengan |
| <= | Kurang dari atau sama dengan |

Operator ini digunakan untuk menguji hubungan antara nilai dan atau variabel.

Berikut contoh penggunaanya dalam program:

#include <iostream>

using namespace std;

void main() {

bool nilai;

nilai = 3 > 2 ; // hasil ungkapan : benar

cout << "Nilai = " << nilai;

nilai = 2 > 3 ; // hasil ungkapan : salah

cout << "Nilai = " << nilai;

}

Dan berikut adalah hasil akhirnya:

* Nilai = 1
* Nilai = 0

**#6 Logical Operators**

**Operator Logika** adalah operator yang digunakan untuk membandingkan dua nilai variabel atau lebih. Hasil dari operasi ini adalah nilai boolean true atau false.

| **Operator** | **Keterangan** |
| --- | --- |
| && | **AND** - Jika semua operand bernilai benar (TRUE) maka kondisi bernilai benar. |
| || | **OR** - Jika salah satu operand bernilai benar (TRUE) maka kondisi bernilai benar. |
| ! | **NOT** - Digunakan untuk membalik kondisi. Jika kondisi benar (TRUE) maka akan berubah menjadi salah (FALSE), begitu pula sebaliknya. |

Berikut adalah contoh penggunaanya:

( (5 == 5) && (3 > 6) ) // Hasil = false, karena ( true && false )

( (5 == 5) || (3 > 6) ) // Hasil = true, karena ( true || false )

**#7 Conditional Ternary Operator**

Pada dasarnya, operator kondisi merupakan penyederhanaan dari bentuk if..else yang setiap blok dari if dan else hanya terdiri dari satu statement/perintah.

Struktur penulisan operator kondisi:

(ekspresi) ? (jika benar) : (jika salah);

Berikut contoh dalam program c++:

#include <iostream>

using namespace std;

int main ()

{

int a,b,c;

a=2;

b=7;

c = (a>b) ? a : b;

cout << c;

}

**#8 Comma Operator**

Operator koma (,) memiliki beberapa fungsi, yaitu:

* Untuk memisahkan deretan deklarasi variabel
* Untuk memisahkan argumen fungsi
* menyatukan dua ekspresi menjadi sebuah pernyataan, dan
* memungkinkan pemberian lebih dari satu ekspresi pada inisialisasi nilai awal.

Berikut contoh program dengan operator koma:

#include <iostream>

using namespace std;

int main ()

{

int a,b;

a = (b=3, b+2);

cout << a;

}

Agak rumit namun penjelasanya seperti ini: Pertama variabel b akan diberi nilai 3, kemudian nilai b (3) akan dijumlahkan dengan 2 sehingga variabel a bernilai 5.

**#9 Bitwise Operators**

Operator bitwise adalah operator yang digunakan untuk memanipulasi bit-bit nilai data yang ada di memori. Operator ini hanya dapat digunakan untuk tipe data:

* char
* int
* long int

Untuk operator bitwise dalam bentuk bit, dilakukan dalam bit per bit (binary).

& AND Bitwise AND

| OR Bitwise inclusive OR

^ NOT Unary complement (bit inversion)

<< SHL Shift bits left

>> SHR Shift bits right

Tabel kebenanaran:

| **p** | **q** | **p & q** | **p | q** | **p ^ q** |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 |

Jika kita asumsikan memiliki nilai A = 63 dan B = 13, mari kita ubah menjadi biner:

A = 0011 1100

B = 0000 1101

Sehingga akan menjadi:

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001

~A = 1100 0011

**#10 Operator Lainnya**

Berikut adalah operator lainnya yang didukung oleh bahasa C++:

| **Operator** | **Keterangan** |
| --- | --- |
| sizeof | Untuk mengembalikan nilai dari ukuran variabel. Contoh sizeof(a), dimana ‘a’ adalah integer maka akan mengembalikan nilai 4 |
| . (dot) and -> (arrow) | Operator member yang digunakan untuk mereferensikan member class, structures, dll. |
| Cast | Untuk mengkonversi suatu data ke tipe data lain. Contohnya int(2.2000) akan menjadi 2 |
| \* | Digunakan sebagai operator pointer |

Dan berikut adalah daftar operator berdasarkan kategorinya:

| **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 |

**Menarik Kesimpulan**

Tenang saja ada tidak perlu menghafal semua operator tersebut, namun anda perlu mengetahuinya. Karena saya yakin nantinya anda akan membutuhkannya.