Operator logika berfungsi untuk membandingkan dua operand bertipe Boolean untuk memecahkan masalah berdasarkan operator yang diunakan, dan hasl dari operasi operator ini akan menghasilkan nilai bertipe Boolean 1 (true) atau 0 (false), tetapi pernyataan “membandingkan dua operand” tersebut bukan merupakan pengertian untuk operator dengan bentuk tanda ‘!’, karena operator ‘!’ hanya membutuhkan satu operand bertipe Boolean.

Tabel Operator Logika :

|  |  |
| --- | --- |
| Operator | Keterangan |
| && | Operasi Logika AND |
| || | Operasi Logika OR |
| ! | Operasi Logika NOT |

Jika operand yang dinyatakan tidak bertipe Boolean (atau tidak nilai 1 atau 0) misalnya operand memiliki nilai 76, maka operator akan memperlakukan operand tersebut sebagai Boolean. Jika nilai operand adalah 0 maka akan di anggap sebagai 0 (false) dan jika nilai bukan 0 maka akan dianggap 1 (true) meskipun nilai tersebut minus.

Macam-macam Operator Logika

( && ) Operator Logika AND

Operator logika AND digunakan untuk mengevaluasi dua ekspresi relasi sebagai operand untuk mendapatkan rasional tunggal. Operasi akan menghasilkan nilai 1 (true) jika kedua operand bernilai 1 (true) tapi jika salah satu atau keduanya bernilai 0 (false) maka akan menghasilkan nilai 0 (false). Dan di bawah ini adalah table dari perbandingan Operator logika AND.

Table Logika AND :

|  |  |  |
| --- | --- | --- |
| X | Y | X && Y |
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | False |

Contoh program

#include <iostream>

using namespace std;

int main ()

{

bool a = 1, b = 0;

cout<< (a && b) <<endl;

return 0;

}

Contoh Kasus :

Ekspresi Relasi 1 : 4 < 10

Ekspresi Relasi 2 : 7==6

Ekspresi Relasi 3 : 9 >= 4

Penggabungan ketiga ekspresi relasi diatas menjadi :

4 < 10 && 7==6 && 9 >= 4

Maka ketiga ekspresi tersebut mempunyai nilai :

• Ekspresi Relasi-1 : 4 < 10 (BENAR = 1)

• Ekspresi Relasi-2 : 7 == 6 (SALAH = 0)

• Ekspresi Relasi-3 : 9 >= 4 (BENAR = 1)

Dari ekspresi relasi tersebut mempunyai 2 nilai BENAR dan 1 nilai SALAH, maka akan menghasilkan :

4 < 10 && 7==6 && 9 >= 4 (SALAH = 0)

Karena seperti yang telah dijelaskan bahwa Operator Logika AND akan bernilai Benar hanya jika semuanya bernilai benar dan false jika salah satu atau lebih ekspresi bernilai false.

Contoh Program :

#include <iostream>

using namespace std;

int main( ){

int a, b, c, hasil1, hasil2, hasil3, hasilAkhir;

cout<<"Masukan Nilai A = ";cin>>a;

cout<<"Masukan Nilai B = ";cin>>b;

cout<<"Masukan Nilai C = ";cin>>c;

hasil1 = a + 4 < 10;

hasil2 = b > a + 5;

hasil3 = c - 3 >= 4;

hasilAkhir = hasil1 && hasil2 && hasil3;

cout<<"= Program Ekspresi AND ="<<endl;

cout<<"=========================="<<endl;

cout<<"Hasil dari Hasil1 = a + 4 < 10 adalah "<<hasil1<<endl;

cout<<"Hasil dari Hasil2 = b > a + 5 adalah "<<hasil2<<endl;

cout<<"Hasil dari Hasil3 = c - 3 >= 4 adalah "<<hasil3<<endl;

cout<<"Hasil dari Hasil Akhir = d && e && f adalah "<<hasilAkhir<<endl;

return 0;

}

Operator Logika OR

Operator logika OR digunakan untuk menghubungkan dua atau lebih ekspresi relasi untuk mendapatkan hasil rasional tunggal, Operator || dibaca sebagai OR, akan dianggap 1 (true), bila salah satu ekspresi relasi yang dibandingkan bernilai 1 (true) dan bila semua ekspresi relasi yang dibandingkan bernilai 0 (false), maka akan bernilai 0 (false).

Tabel Perbandingan Operator Logika OR

|  |  |  |
| --- | --- | --- |
| X | Y | A || B |
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

Contoh Program :

#include <iostream>

using namespace std;

int main ()

{

bool a = 1, b = 0;

cout<< (a || b) <<endl;

return 0;

}

Contoh Kasus :

Ekspresi Relasi 1 : 4 < 10

Ekspresi Relasi 2 : 7==6

Ekspresi Relasi 3 : 9 >= 4

Penggabungan ketiga ekspresi relasi diatas menjadi :

4 < 10 && 7==6 && 9 >= 4

Maka ketiga ekspresi tersebut mempunyai nilai :

• Ekspresi Relasi-1 : 4 < 10 (BENAR = 1)

• Ekspresi Relasi-2 : 7 == 6 (SALAH = 0)

• Ekspresi Relasi-3 : 9 >= 4 (BENAR = 1)

Dari ekspresi relasi tersebut mempunyai 2 nilai BENAR dan 1 nilai SALAH, maka akan menghasilkan :

4 < 10 || 7==6 || 9 >= 4 (BENAR/1)

Karena seperti yang telah dijelaskan bahwa Operator Logika OR akan bernilai Benar jika salah satu ekspresi relasi atau semua ekspresi relasi bernilai benar dan false jika semua ekspresi bernilai false.

Contoh Program :

#include <iostream>

using namespace std;

int main( ){

int a, b, c, hasil1, hasil2, hasil3, hasilAkhir;

cout<<"Masukan Nilai A = ";cin>>a;

cout<<"Masukan Nilai B = ";cin>>b;

cout<<"Masukan Nilai C = ";cin>>c;

hasil1 = a + 4 < 10;

hasil2 = b > a + 5;

hasil3 = c - 3 >= 4;

hasilAkhir = hasil1 || hasil2 || hasil3;

cout<<"= Program Ekspresi OR ="<<endl;

cout<<"=========================="<<endl;

cout<<"Hasil dari Hasil1 = a + 4 < 10 adalah "<<hasil1<<endl;

cout<<"Hasil dari Hasil2 = b > a + 5 adalah "<<hasil2<<endl;

cout<<"Hasil dari Hasil3 = c - 3 >= 4 adalah "<<hasil3<<endl;

cout<<"Hasil dari Hasil Akhir = d || e || f adalah "<<hasilAkhir<<endl;

return 0;

}

Cara Kerja operator AND dan OR pada Compiler :

Berdasarkan pernyataan-pernyataan di atas, C++ dalam mengevaluasi hal ini akan menggunakan cara Arus pendek (Short-circuit) yang berarti hanya akan mengevaluasi apa yang diperlukan dari kiri ke kanan. Mendapatkan tujuanya dan mengabaikan sisanya. contoh (5==6 && 4>3). C++ akan memproses dan mengevaluasi dari kiri ke kanan apakah 5==6 adalah 0 (false) atau 1 (true) dan jika mempunyai nilai 1 (true) maka akan melanjutkan pemeriksaan ke ekspresi selanjutnya tapi kenyataanya adalah 0 (false) maka pemeriksaan akan berhenti di situ dan mengabaikan sisanya. karena sifat operator AND && akan, untuk mendapatkan nilai 1 (true) dibutuhkan semua operand bernilai 1 (true). Hal itu tidak jauh beda dengan operator OR ||.

&& : Jika mendapatkan nilai 0 (false) maka langsung menghasikan nilai 0 (false).

|| : Jika mendapatkan nilai 1 (true) maka langsung menghasikan nilai 1 (True).

Contoh Program :

#include <iostream>

using namespace std;

int main( ){

int a, b, c, hasil1, hasil2, hasil3, hasilAkhir;

cout<<"Masukan Nilai A = ";cin>>a;

cout<<"Masukan Nilai B = ";cin>>b;

cout<<"Masukan Nilai C = ";cin>>c;

hasil1 = a + 4 < 10;

hasil2 = b > a + 5;

hasil3 = c - 3 >= 4;

hasilAkhir = hasil1 && hasil2 || hasil3;

cout<<"= Program Ekspresi AND OR ="<<endl;

cout<<"==========================="<<endl;

cout<<"Hasil dari Hasil1 = a + 4 < 10 adalah "<<hasil1<<endl;

cout<<"Hasil dari Hasil2 = b > a + 5 adalah "<<hasil2<<endl;

cout<<"Hasil dari Hasil3 = c - 3 >= 4 adalah "<<hasil3<<endl;

cout<<"Hasil dari Hasil Akhir = d && e || f adalah "<<hasilAkhir<<endl;

return 0;

}

Operator Logika NOT

Operator logika NOT berfungsi membalikan nilai sebenarnya. Jika nilai yang disebutkan bernilai 1 (true) maka akan menghasilkan nilai 0 (false), begitu pula sebaliknya.

Contoh :

!true // Pernyataan bernilai 1 (true) dibalikan dengan operator NOT ‘!’ dan menjadi 0 (false)

!false // Pernyataan bernilai 0 (false) dibalikan dengan operator NOT ‘!’ dan menjadi 1 (true)

!(7==7) // Pernyataan bernilai 1 (true) dibalikan dengan operator NOT ‘!’ dan menjadi 0 (false)

!(9<=2) // Pernyataan bernilai 0 (false) dibalikan dengan operator NOT ‘!’ dan menjadi 1 (true)

Contoh kasus :

Ekspresi Relasi : 4 < 10

Penggunaan umum Operator Logika NOT diatas menjadi;

!( 4 < 10) = 0 (false)

ekspresi diatas mempunyai nilai 1 (true) dan jika mengunakan operator logika NOT, maka ekspresi tersebut akan bernilai 0 (false)

Contoh Program :

#include <iostream>

using namespace std;

int main(){

int A, hasilSebenarnya, hasilNOT;

cout<<"Masukan Nilai A = "; cin>>a;

hasilSebenarnya = (a + 5 < 15);

hasilNOT = !(a + 5 < 15);

cout<<"= Program Ekspresi NOT ="<<endl;

cout<<"========================"<<endl;

cout<<"Nilai A = "<<a<<endl

cout<<"Nilai hasilSebenarnya = (a + 4 < 10) = "<<hasilSebenarnya<<endl;

cout<<"Nilai hasilNOT = !(hasilSebenarnya) atau !(a + 5 < 15) = "<<hasilNOT;

return 0;

}