Pernyataan CONTINUE adalah pernyataan yang memungkinkan kita mengatur aliran eksekusi CPU terhadap program buatan kita. Pernyataan CONTINUE adalah keyword yang berfungsi untuk menyelesaikan perulangan saat itu dan melanjutkan ke perulangan selanjutnya dengan mengabaikan sisa pernaytaan pada badan perulangan tersebut. Pernyataan CONTINUE hanya dapat digunakan di dalam pernyataan pengulangan.

Bentuk Penulisan

continue;

Pernyataan CONTINUE sangat berguna jika kita ingin mengakhiri perulangan pada saat itu dan melanjutkan ke perulangan selanjutnya. Pada saat CPU bertemu dengan pernyataan CONTINUE, CPU akan meloncat mengabaikan pernyataan-pernyataan yang ada di bawah pernyataan CONTINUE menuju ke akhir baris dari pernyataan perulangan, dengan hal itu perulangan selanjutnya akan segera dimulai..

Contoh Program

#include <iostream>

using namespace std;

int main ()

{

int jumlah=0;

cout<<"Jumlah : ";cin>>jumlah;

for (int i = 1; i <= jumlah; i++){//Perulangan dimulai

if ((i % 2)==0) continue; //jika tereksekusi CPU akan meloncat

cout<<i<<endl; //pernyataan ini akan diabaikan

//CPU akan mendarat di sini

}

return 0;

}

Contoh program di atas adalah contoh penggunaan dari pernyataan CONTINUE. Program di atas akan meminta anda untuk memasukan dan akan menampilkan hitungan angka ganjil sebanyak angka yang pengguna masukan.

Dari program di atas dapat menampilkan hitungan ganjil karena jika pernyataan CONTINUE tereksekusi oleh CPU maka CPU akan meloncat ke baris akhir dari pernyataan pengulangan dan jika kondisi perulangan terpenuhi maka perulangan akan melanjutkan perulangan selanjutnya. Loncatan tersebut akan mengabaikan sisa dari badan pernyataan pengulangan, sehingga pernyataan keluaran di bawah pernyataan CONTINUE tidak akan dieksekusi oleh CPU.

Setelah penjelasan di atas memang CONTINUE adalah pernyataan yang sangat berguna, tapi hati-hati dalam menggunakanya, Perhatikan aliran program yang sedang anda buat, jika tidak anda bisa membuat suatu program yang sulit untuk dipahami dan anda juga bisa membuat program yang gagal berfungsi.

Salah satu kasus adalah anda bisa membuat sebuah program dengan pengulangan tak terhinga seperti di bawah ini.

#include <iostream>

using namespace std;

int main ()

{

int jumlah=0;

cout<<"Jumlah : ";cin>>jumlah;

int i = 1;

while(i <= jumlah){

if ( i == 5) continue;

cout<<i<<endl;

i++;

}

return 0;

}

Di atas adalah program contoh dari kesalahan penggunaan pernyataan CONTINUE, program di atas mempunyai maksud untuk menampilkan penghitungan 1 ke angka yang dimasukan oleh pengguna, dengan melewati angka 5.

Tapi hasil dari program di atas akan memiliki perulangan yang tak terhinga, program di atas jika dijalankan terlihat berhenti dan tidak berakhir. Hal itu karena pernyataan CONTINUE diletakan sebelum penaikan nilai variabel yang digunakan untuk conditional expression oleh pernyataan pengulangan. Hal tersebut akan menyebabkan pernyataan penaikan variabel akan diabaikan.

Ketika variabel “i” bernilai 5 CPU akan mengeksekusi pernyataan CONTINUE, melompat dan mengabaikan penaikan variabel tersebut. di saat evaluasi pernyataan pengulangan, variabel “i” akan bernilai 5 karena pernyataan penaikan variabel tersebut di abaikan. Karena hal itu CPU akan kembali lagi untuk mengeksekusi pernyataan CONTINUE. Dan hal itu akan terjadi berulang-ulang tanpa henti.

Di bawah ini adalah contoh program untuk menghindari hal tersebut.

#include <iostream>

using namespace std;

int main ()

{

int jumlah=0;

cout<<"Jumlah : ";cin>>jumlah;

int i = 1;

while(i <= jumlah){

if ( i == 5) {

i++;

continue;

}

cout<<i<<endl;

i++;

}

return 0;

}

Atau

#include <iostream>

using namespace std;

int main ()

{

int jumlah=0;

cout<<"Jumlah : ";cin>>jumlah;

int i = 1;

do{

if ( i == 5) continue;

cout<<i<<endl;

}while(++i <= jumlah);

return 0;

}

saat anda menggunakan pernyataan lompatan anda harus berhati-hati dalam penggunaan pernyataan tersebut, dan cermat dalam menganalisis bagaimana aliran program anda.

Pengertian dan Contoh Pernyataan Continue C++ - Belajar C++

Pernyataan lompatan Continue berfungsi untuk

Pernyataan lompatan Continue berfungsi untuk melewati/melompoti suatu atau beberpa angka pengulangan. untuk melewatkan sisa loop pada iterasi saat ini, seolah-olah akhir blok pernyataan telah dicapai, menyebabkan ia melompat ke awal iterasi berikut. mengembalikan proses yang sedang dilaksanakan ke-awal loop lagi, tanpa menjalankan sisa perintah dalam loop tersebut.

Bentuk umum penulisan :

continue;

Contoh Program :

// Perulngan FOR

#include <iostream>

using namespace std;

int main( )

{

int nomer = 1 ;

for ( nomer = 1 ; nomer <= 10 ; ++nomer )

{

if( nomer == 5 )

continue ;

cout << nomer << endl ;

}

return 0 ;

}

//Perulangan WHILE

#include <iostream>

using namespace std;

int main( )

{

int nomer = 0 ;

while(nomer<=10){

nomer++;

if( nomer == 5 )

continue ;

cout << nomer << endl ;

}

return 0 ;

}

//Pengulangan DO-WHILE

#include <iostream>

using namespace std;

int main( )

{

int nomer = 0 ;

do{

nomer++;

if( nomer == 5 )

continue ;

cout << nomer << endl ;

}while(nomer<=10);

return 0 ;

}

Contoh Program 2 :

#include <iostream>

using namespace std ;

int main ( )

{

int a = 3, b = 2, c = 1, bil ;

cout << "Bil-A | Bil-B | Bil-C\n" << endl ;

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

for (bil = 1; bil <= 10 ; ++bil )

{

a += b; b += c; c += 2 ;

if ( c == 13 ) continue ;

cout << a << "\t |"<<b<<"\t |" << c << endl ;

}

return 0 ;

}

Cukup sekian dari belajarcpp.com semoga artikel ini bisa bermanfaat untuk kita semua. Mohon maaf atas kekuranganya dan Terima Kasih atas dukungan dan kunjungan anda ke BelajarCPP. Have a nice day.

Last update : 4 Februari 2016