Pernyataan GOTO adalah pernyataan yang memungkinkan kita untuk mengatur arahnya aliran pengeksekusian CPU terhadap program kita. GOTO berfungsi untuk memerintahkan CPU melompat ke baris manapun berdasarkan label yang telah dibuat.

Bentuk Penulisan Goto

goto nama\_label;

Bentuk penulisan Label Goto

nama\_label:

untuk membuat pernayataan goto bekerja, dibutuhkan dua pernyataan yaitu GOTO dan LABEL. Pernyataan GOTO menggunakan keyword “goto” diikuti nama label berfungsi untuk memberi tahu CPU agar melompat ke baris yang miliki label dengan nama tersebut.

Pernyataan LABEL di isi dengan nama label diakhiri dengan tanda titik dua, berfungsi untuk menandai suatu baris program, tempat dimana loncatan CPU karena GOTO mendarat.

Contoh Penulisan Goto

goto labelku;

Contoh Penulisan Label Goto

labelku:

Ketika CPU bertemu dengan pernyataan GOTO maka CPU pada saat itu juga akan melompat ke label dengan nama yang tertera pada pernyataan GOTO. Dan untuk penempatan GOTO beserta label GOTO kita bebas meletakanya dimana saja tetapi harus berada pada satu function scope.

Contoh Program :

#include <iostream>

using namespace std;

int main ()

{

int angka=0;

cobaLagi:

cout<<"Masukan Angka : ";cin>>angka;

if (angka!=5)goto cobaLagi;

return 0;

}

Contoh program di atas adalah contoh penggunaan pernyataan GOTO. Program di atas akan meminta pengguna untuk memasukan angka apapun, jika pengguna memasukan angka 5 maka pernaytaan GOTO akan dieksekusi, CPU akan meloncat ke label “cobaLagi” yang diletakan di atas.

Peletakan Label dari GOTO bisa diletakan di baris sebelumnya, kita juga bisa meletakan label tersebut di baris selanjutnya.

Contoh Program

#include <iostream>

using namespace std;

int main ()

{

char loncat=0;

cout<<"Masukan Angka [y/n] : ";cin>>loncat;

cout <<"1"<<endl;

if (loncat=='y')goto loncatan;

cout <<"2"<<endl;

loncatan:

cout <<"3"<<endl;

return 0;

}

Contoh program di atas akan meminta persetujuan anda untuk melompat, jika anda menginputkan ‘n’ maka aliran akan berjalan normal dan menghitung 1 sampai 3, jika anda memilih y maka akan tampil perhitungan 1 dan 3.

Setelah penjelasan mengenai GOTO dan cara penggunaan GOTO, penulis belajarcpp.com sarankan untuk tidak menggunakan pernyataan GOTO jika tidak dibutuhkan. Sebenarnya bukan hanya penulis yang menyarankan, tapi hamper semua programmer menghindari penggunaan pernyataan GOTO.

Pernyataan GOTO dapat menurunkan kualitas program, karena bisa membuat program mempunyai aliran yang sangat tidak jelas. Kadang hal itu bisa menyebabkan kegagalan pada program anda.

Di bahasa pemrograman lainnya, salah satunya seperti bahasa pemrograman BASIC, Goto sangat sering digunakan, hal itu normal di bahasa pemrograman BASIC tapi berbeda pada Bahasa pemrograman C++, semua programmer yang menggunakan bahasa pemrograman C++ sama sekali tidak menggunakan pernyataan GOTO, mereka mengganti pernyataan GOTO dengan memanfaatkan pernyataan lain yang lebih canggih dan aman seperti pernyataan pengulangan, penyeleksian dan lain-lain.

Dan di bawah ini adalah contoh program mengapa GOTO tidak disarankan untuk digunakan.

#include <iostream>

using namespace std;

int main ()

{

goto loncat;

int var = 2;

loncat:

cout<<var;

return 0;

}

Program di atas tidak akan bisa dijalankan karena CPU akan dilempar langsung ke baris pada label loncat dengan arti deklarasi var tidak pernah di eksekusi, dan CPU tidak bisa menemukan variabel var saat pernaytaan keluaran.

Bayangkan jika anda membuat program dengan kode yang cukup banyak, dan anda menggunakan begitu pernataan GOTO. Kemungkinan yang akan terjadi adalah anda akan bingung bagaimana CPU akan mengalir untuk mengeksekusi program anda, anda sulit untuk menyunting program anda dan kemungkinan program akan mendapatkan kesalahan yang cukup fatal karena aliran program yang tidak jelas.

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

Pernyataan Pengulagan GOTO berfungsi untuk mengendalikan arah jalannya proses pengeksekusian. memungkinkan untuk melompati atau mengembalikan proses pengeksekusian, kita bisa mengembalikan ke atas tempat dimana label yang sudah didirikan pada sebelumnya.

Pernyataan goto merupakan instruksi untuk mengarahkan eksekusi program ke-pernyataan yang diawali dengan suatu label. Label merupakan suatu pengenal (Identifier) yang diikuti dengan tanda titik dua ( : ).

Bentuk Umum Penulisan :

// pendirian label

namaLabel:

// GOTO pemanggilan label

goto namaLabel;

Ingat, mendirikan label menggunakan titik dua (colon) pada akhir baris bukan titik koma (semicolon).

Contoh :

#include <iostream>

using namespace std ;

int main ( )

{

atas : // deklarasi label

cout << "Cetak" << endl ;

goto atas ; //penggunaan label

return 0 ;

}

Program di atas adalah contoh penggunaan pernyataan lompatan GOTO. Pada baris pertama telah didirikan identifier label bernama “atas”, berfungsi untuk menandai awal mulai proses pengeksekusian saat pemanggilan GOTO. Dan program di atas tidak akan berhenti untuk mencetak kata “Cetak” dikarenakan proses pengeksekusian selalu diarahkan kembali ke atas. Dan kemungkinan tidak ada yang menghentikan program tersebut meskipun kita coba untuk menggunakan pernyataan lompatan break. Karena pernyataan goto bukanlah pernyataan pengulangan meskipun program diatas seperti perulangan tak terhingga.

Contoh Program :

#include <iostream>

#include <conio.h>

#include <windows.h>

using namespace std;

int main ( )

{

int angka, hasil;

char lagi;

atas: // deklarasi label

system("cls");

cout<<"Masukan Bilangan = "; cin>>angka;

hasil=angka%2;

cout<<"Nilai "<<angka<<" dibagi 2 adalah "<<hasil;

cout<<"\n\nRestart Program ? [Y/N]";

lagi = getche();

if(lagi=='Y' || lagi=='y') goto atas; //penggunaan label

else return 0;

}

Dan di atas adalah contoh program penggunaan pernyataan lompatan GOTO ke dua. Penggunaan “GOTO” pada program kedua tdak sama seperti contoh program yang pertama, di program kedua pernyataan goto dapat kita kendalikan agar tidak mendapatkan pengulangan tak terhingga. Dengan menggunakan pernyataan penyeleksian IF-ELSE, jika pegguna memasukan karakter ‘Y’ maka pemanggilan GOTO akan dilaksanakan, jika pemanggilan yang lain maka akan memanggil pernyataan “return 0” yang berarti keluar program.

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