Structures atau struct adalah salah satu fitur bahasa pemrograman C, merupakan keyword yang memungkinkan kita untuk membuat pengelompokan variabel dengan tipe data yang berbeda. struct dibuat sebagai tipe data baru untuk program anda. Mirip seperti array yang memungkinkan kita membuat kelompok dari variabel, perbedaan struct dan array adalah, array hanya memungkinkan untuk membuat kelompok variabel dengan jenis type data yang sama, dan struct memungkinkan kita untuk membuat sekelompok variabel dengan bermacam-maca tipe data. Isi dari struct bisa disebut sebagai elemen atau anggota(member).

Bentuk umum penulisan

Contoh ke Satu

struct namaTipe {

tipeMember1 namaMember1;

tipeMember2 namaMember2;

…

} namaObject;

struct namaTipe namaObject; //deklarasi object diluar definisi struct, atau

namaTipe namaObject;

Contoh ke Dua

struct {

tipeMember1 namaMember1;

tipeMember2 namaMember2;

…

] siswa ;

Keterangan =

NamaTipe = adalah nama atau identifier sebagai pengenal dari struct tersebut.

tipeMember = adalah tipe data dari deklarasi variabel, anggota sturct.

namaMember = adalah nama atau identifier dari member deklarasi variabel.

namaObject = adalah nama pengenal yang sah untuk mengakses anggota deklarasi di dalam struct.

Hal yang perlu diperhatikan =

Kita membutuhkan object untuk mengakses anggota dari struct.

Jika kita menulis nama disebelah keyword struct (contoh penulisan ke satu), nama tersebut akan berlaku sebagai pengenal struct tersebut, dan kita perlu untuk mendeklarasikan object untuk mengakses anggota struct.

Untuk deklarasi object kita bisa membuatnya di dalam struct diantara tanda “{…}” dan tanda semicolon ‘ ; ’. atau bisa didirikan diluar definisi struct.

Jika kita hanya menulis nama object, setelah “{…}” dan sebelum ‘ ; ’, dan tanpa memberikan nama disebelah keyword struct (contoh penulisan ke dua), anda akan membuat struct tanpa pengenal. Nama diantara kedua tanda tersebut akan berlaku sebagai object dan kita tidak bisa membuat object di luar definisi struct.

Contoh penulisan

struct siswa{

int noInduk ;

char nama[20] ;

float nilaiUjian ;

}budi;

struct siswa riski; //deklarasi object diluar definisi struct, atau

siswa adi;

//Contoh Ke-dua

struct{

int noInduk ;

char nama[20] ;

float nilaiUjian ;

}siswa;

Cara mengakses Struct

Sebelum kita melakukan pengaksesan kepada struct yang kita buat, terlebih dahulu kita perlu mengenal tentang Object. Object adalah deklarasi yang menggunakan tipe data dengan struct yang kita buat, dengan hal ini kita dimungkinkan memiliki semua anggota struct pada deklarasi object yang kita buat. Ada dua cara kita untuk membuat object pada struct, cara pertama kita bisa sisikan nama object setelah tanda “{ … }” dan sebelum tanda semicolon ‘ ; ’, dan cara kedua kita dirikan seperti deklarasi variabel diluar dari definisi struct tetapi masih di dalam jangkauan dari struct, contohnya kita bisa dirikan di luar function bersebelahan dengan file header itu akan membuatnya menjadi object global, atau kita bisa dirikan di dalam function utama atau function yang lain, membuatnya menjadi object local. Untuk contohnya anda bisa melihat contoh yang penulis berikan di atas.

Setelah kita berhasil membuat sebuah deklarasi object. untuk mengakses anggota struct kita membutuhkan tanda ”Member Access Operator ( . )” di antara nama object dan nama anggota variabel struct.

budi.noInduk

riski.nama

adi.nilaiUjian

Initializing Struct

Initializing atau inisialisasi adalah tugas untuk nilai awal dari variabel. dalam struct ada dua cara untuk membuat nilai awal pada anggota-anggota struct. Yang pertama adalah memberikan inisialisasi disaat mendrikan variabel sebagai member struct (non-static member initialization), maka nilai itu akan berlaku kepada semua object yang dibuat dengan struct tersebut. dan kedua adalah dengan memberikan inisialisasi disaat mendirikan object (uniform initialization), nilai tersebut hanya akan berlaku pada object itu sendiri.

Contoh penulisan

//Cara kesatu (non-static member initialization)

struct siswa{

int noInduk = 1 ;

char nama[20] = unamed ;

float nilaiUjian = 0;

};

//Cara Kedua ( uniform initialization )

struct siswa budi = {20103,Budi Kanjeng Berbudi,8.75}; //inisialisasi noInduk=20103, nama[20]=“Budi Kanjeng Berbudi”, nilaiUjian=8.75.

siswa tono {1, “Tono Ono Loro”}; //C++ 11, inisialisasi noInduk = 1, nama[20]=”Tono Ono Loro”, nilaiUjian = 0.0 (nilai basis).

(C++ 11) Ada satu hal yang harus anda ingat, ketika anda menggunakan cara ke satu, anda tidak bisa menggunakan cara kedua bersamaan, Begitu pula sebaliknya. inisialisasi adalah tentang hal memberi nilai awal. jika cara ke satu sudah dilakukan, melakukan cara kedua bukan lagi sebuah inisialisasi. Tapi pada (C++ 14) anda bisa menggunakan kedua cara secara bersamaan.

Contoh Program Sturct

#include <iostream>

#include <string.h>

using namespace std;

struct siswa{

int nomerInduk;

char nama[30];

float nilaiUjian;

} ;

int main() {

//Inisialisasi

struct siswa budi = {1, "Budi Kanjeng Berbudi", 7.85};

siswa andi { 2, "Andi Ndi Adi"};

siswa tono;

//Assigment Operator

budi.nomerInduk = 1234;

strcpy(budi.nama, "Budi Bebudi Udin");

budi.nilaiUjian = 8.25;

andi.nomerInduk = 1234;

andi.nilaiUjian = 8.50;

tono.nomerInduk = 1234;

strcpy(tono.nama, "Tono Mono Ton");

tono.nilaiUjian = 9.20;

//Output

cout<<budi.nomerInduk<<endl;

cout<<budi.nama<<endl;

cout<<budi.nilaiUjian<<endl<<endl;

cout<<andi.nomerInduk<<endl;

cout<<andi.nama<<endl;

cout<<andi.nilaiUjian<<endl<<endl;

cout<<tono.nomerInduk<<endl;

cout<<tono.nama<<endl;

cout<<tono.nilaiUjian<<endl<<endl;

return 0;

}