Akses Modifiers adalah salah satu fitur yang penting dalam OOp, yang memungkinkan kita mengatur sifat akses dari member class agar tidak sembarang perintah dapat melihat atau merubah nilai dari member class secara langsung. Akses modifier memiliki 3 akses specifiers, yaitu public, private, dan protected.

Semua baris deklarasi sebagai member class akan mengikuti sifat dari label di atasnya hingga label berikutnya atau tanda penutup class “ } ” , ketiga akses specifier mempunyai sifat mereka masing-masing. Dan sifat dasar akses atau default access adalah private. Dan dibawah ini adalah macam-macam label akses specifier dengan keterangan dan contoh program.

**Macam-macam Akses Specifier**

**Public**

Public Adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya (di bawahnya), sehingga memiliki sifat dapat di akses dari manapun. Dapat di akses dari dalam class itu sendiri, dari anak class (derived class) dan juga dari luar class.

Contoh program

#include <iostream>

using namespace std;

class siswa{

public: //Akses Specifier

int id;

};

int main (){

siswa budi;

budi.id = 10;

cout<<budi.id<<endl;

return 0;

}

Di atas adalah contoh program untuk membuktikan sifat dari akses specifier public. Member yang mengikuti label public adalah “id”.dan di main function kita membuat object “budi”, untuk mencoba mengakses member “id” dari instance “budi”. Dan membuktikan bahwa label public akan bersifat dapat diakses dari manapun, dari dalam class, anak cass dan diluar class.

**Private**

Private adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya, sehingga memiliki sifat yang tidak dapat di akses dari manapun kecuali melalui friend function dan dari dalam class itu sendiri.

Contoh program

#include <iostream>

using namespace std;

class siswa{

private:

int id;

};

class orangtua : public siswa{

public:

int getId(){

return id;

}

};

int main (){

siswa budi;

budi.id = 10;

cout<<budi.id<<endl;

return 0;

}

Di atas adalah program yang sama seperti contoh program ke satu (public), penulis hanya menambahkan anak class (derived class), dan mengganti label public dengan private.

Untuk anda yang tdak mengerti derived class (inheritance), tidak perlu bingung, penulis akan jelaskan tentang inheritance di artikel selanjutnya, dan untuk kali ini, kita hanya akan membuktikan sifat dari akses specifier.

Dari program di atas akan menghasilkan pesan Error pada compiler, yang berbunyi.

Error : ‘int siswa::id’ is private

Pesan tersebut berasal dar 3 tempat yaitu pesan pertama dari perintah yang berada di dalam derived class, kedua dan ketiga berasal dari perintah pemanggilan member “id” yang berada pada main function.

Pesan itu membuktikan bahwa member “id” mengikuti sifat label private. Yang tidak dapat diakses dari luar class dan tidak bisa diakses dari derived class “orangtua”. Private memiliki sifat hanya dapat di akses dari dalam class itu sendiri. Jadi masih ada cara untuk mengakses member id tersebut dari luar class, dengan cara membuat perantara di dalam class dengan function yang bersifat public, cara ini disebut setter dan getter.

Contoh program

#include <iostream>

using namespace std;

class siswa{

private:

int id;

public:

int getId(){

return id;

}

void setId(int i){

id = i;

}

};

int main (){

siswa budi;

budi.setId(10);

cout<<budi.getId()<<endl;

return 0;

}

Atau kita juga bisa menggunakan friend function untuk dapat mengakses member class yang bersifat private, Yang akan penulis jelaskan pada artikel berikutnya.

**Protected**

Protected adalah label yang berfungsi untuk menentukan sifat akses semua member yang mengikutinya, sehingga memiliki sifat yang tidak dapat diakses dari luar class tapi dapat di akses dari dalam class maupun anak class (derived class).

Contoh program

#include <iostream>

using namespace std;

class siswa{

protected:

int id;

};

class orangtua : public siswa{

public:

int getId(){

return id;

}

};

int main (){

siswa budi;

budi.id = 10;

cout<<budi.id<<endl;

return 0;

}

Di atas adalah contoh program yang masih sama seperti contoh program pada contoh ke satu dari penjelasan private. Penulis hanya mengubah label akses specifier dari private menjadi protected.

Dari program di atas akan menghasilkan pesan Error pada compiler, yang berbunyi.

Error : ‘int siswa::id’ is protected

Pesan tersebut hanya berasal dari perintah pemanggilan member “id” yang berada pada main function, membuktikan sifat akses dari protected. Member “id” tidak bisa diakses dari luar class tapi masih bisa diakses dari derived class.

Untuk dapat mengakses member yang bersifat protected dari luar class kita juga bisa menggunakan cara setter getter atau friend function.

Note = jika anda tidak memberikan label akses specifier, maka sifat dari member akan menjadi private akses. Private adalah sifat dasar akses dari class.