**Destructor**

Destructor merupakan kebalikan dari constructor, Destuctor adalah special member function yang akan dieksekusi saat sebelum berakhirnya program atau penghapusan instantce (object). destructor biasanya digunakan untuk melakukan persiapan sebelum object tersebut dihapus.

Untuk mendirikan destructor sama seperti kita mendirikan constructor tapi dibutuhkan tanda ( ~ ) yang diletakan pada awal dari identitas destructor.

Bentuk penulisan

class nama\_class{

public :

~nama\_class(){

//Definisi Destructor

}

}

Karakteristik Destructor

Destructor memiliki identitas yang sama seperti classnya, dan memiliki tanda ( ~ ) pada awal identitas destructor

Destructor tidak memiliki return type.

Destructor harus berada dalam label public.

Destructor tidak mempunyai parameter.

Destructor tidak bisa dipanggil dari luar class dan bisa di panggil dari dalam member function class tersebut (Tapi tidak disarankan).

Destructor akan dieksekusi saat sebelum object tersebut dihapus (jika Object hasil alokasi) atau di saat compiler bertemu tanda “ } ” pada akhir fungsi utama (Jika Object biasa), Destructor memberi kemudahan untuk para programmer untuk melakukan persiapan yang dibutuhkan sebelum object itu dihapus. Biasanya digunakan untuk mengerjakan beberapa tugas seperti menyimpan data, melepaskan memori dinamis yang dipesan saat object berjalan, menutup file dan lain-lain.

Dalam mendirikan destructor, destructor tidak bisa memilki parameter, dan juga berarti destructor tidak bisa digunakan untuk overloading (Satu Class hanya memiliki satu Destructor).

Contoh Program

#include <iostream>

#include <string>

using namespace std;

class siswa{

private:

string nama;

public:

siswa(string); //Constructor

~siswa(); //Destructor

};

//Definisi ember Functon

siswa::siswa(string iNama){ //Constructor

nama = iNama;

cout<<nama<<" = Constructor Terpanggil"<<endl;

}

siswa::~siswa(){ //Destructor

cout<<nama<<" = Destructor Terpanggil"<<endl;

}

int main(){

siswa budi("Budi"); //Constructor Dipanggil

siswa \*pBudi=new siswa("pBudi");//Constructor Dipanggil

delete pBudi;//Destructor Dipanggil

return 0;

}//Destructor Dipanggil

Di atas adalah contoh program yang mencontohkan cara menggunakan destructor. Dan membuktikan sifat destructor yang akan dipanggil ketka object akan dihapus. Di baris pertama pada fungsi utama terdapat pernyataan untuk mendirikan object biasa dengan nama “budi”. Dan baris kedua terdapat deklarasi object pointer dengan menunjuk ke memori yang telah di alokasikan.

Baris selanjutnya adalah pernyataan yang akan berfungsi untuk menghapus “pBudi”, dari tindakan penghapusan tersebut compiler akan memanggil destructor dari class yang digunakan. Dan destructor dari object “budi” akan dipanggil pada saat compiler bertemu tanda “ } ” di fungsi utama (Berakhirnya program).

Jika object tersebut didirikan dengan cara biasa (buka dari alokasi memori, beserta member-member di dalamnya adalah member yang bukan hasil alokasi, sebenarnya kita tidak perlu untuk melepaskan memori dari deklarasi tersebut atau mendirikan destructor yang hanya untuk melepas member-member di dalamna, karena semua deklarasi member biasa (bukan hasil alokasi) akan dihapus otomatis oleh compiler, disaat berakhirnya program.

Tapi jika kita membuat member-member secara dinamis (Hasil alokasi), kita diwajibkan untuk melepaskan semua memori yang telah dipesan oleh kita (karena tidak otomatis dihapus oleh compiler saat proram berakhir). Di sinilah destructor dibutuhkan untuk melakukan pelepasan member-member dinamis pada object.

Dari penjelasan tersebut bukan berarti destructor digunakan hanya untuk melakukan pelepasa memori yang telah di alokasi. Tapi destructor juga bisa digunakan untuk hal lainya.

Contoh Program

#include <iostream>

using namespace std;

class angka{

private:

int \*arr;

int panjang;

public:

angka(int); //Constructor

~angka(); //Destructor

void cetakData();

void isiData();

};

//Definisi member Functon

angka::angka(int i){ //Constructor

panjang = i;

arr = new int[i];

isiData();

}

angka::~angka(){ //Destructor

cout<<endl;

cetakData();

delete[]arr;

cout<<"Alamat Array Sudah Dilepaskan"<<endl;

}

void angka::cetakData(){

for(int i=1;i<=panjang;i++){

cout<<i<<" = "<<arr[i]<<endl;

}

}

void angka::isiData(){

for(int i=1;i<=panjang;i++){

cout<<i<<" = ";cin>>arr[i];

}

cout<<endl;

}

int main(){

angka belajarcpp(3); //Constructor Dipanggil

angka \*ptrBelajarcpp = new angka(5); //Constructor Dipanggil

delete ptrBelajarcpp; //Destructor Dipanggil

return 0;

}//Destructor Dipanggil