Delete adalah salah satu keyword untuk melakukan penghapusan atau pembebasa memori, yang tersedia pada bahasa pemrograman C++. Delete adalah salah satu operator yang tersedia pada header <new>, tetapi kita juga masih bisa menggunakan keyword delete tanpa menggunakan header tersebut.

Mirip seperti fungsi alokasi pada bahasa pemrograman C, yaitu free(). Operasi delete memiliki kelebihan dan disarankan untuk programmer C++. Operator delete berfungsi untuk menghapus atau membebaskan memori yang telah dipesan atau dialokasi secara dinamis oleh allocator seperti new, malloc, calloc, realloc dan sebagainya. Hal itu dilakukan jika memori tersebut sudah tidak dibutuhkan lagi oleh pengguna, dibebaskan dan dikembalikan kepada heap dan membuat memori tersebut akan tersedia untuk pemesanan atau alokasi lainya.

Penggunaan operasi deallocate seperti delete sangat disarankan untuk dilakukan, setelah memori dari pemesanan atau alokasi memori jika memang sudah tidak dibutuhkan lagi, hal ini berguna untuk mencegah adanya kebocoran memori (Memory Leak).

Bentuk Umum Penulisan / Syntax :

delete pointer;

delete []pointer;

Keterangan :

pointer = pointer berisi alamat memori yang akan dibebaskan atau dihapus.

Delete memiliki dua macam penulisan dengan fungsi yang berbeda, yang pertama operasi delete “delete pointer;” dikhususkan untuk membebaskan atau menghapus memori tunggal dan dan yang kedua “delete []pointer;” dikhususkan untuk membebaskan atau menghapus memori berelemen (majemuk) dengan kata lain adalah array.

Jika pointer bukan menunjuk kepada alamat memori yang berasal dari operasi fungsi allocator memori dan jika pointer tersebut memiliki nilai NULL, maka fungsi deallocate seperti delete tidak melakukan apapun.

Dalam pembebasan atau penghapusan memori memang tidak membuat pointer tersebut bernilai null atau berhenti untuk menunjuk memori tersebut. Pada saat melakukan alokasi atau pemesanan, C++ meminta kepada Opertating System untuk mencarikan memori yang dapat digunakan, setelah mendapatkanya memori tersebut akan diserahkan ke pointer dan ditandai sebagai memori yang sudah dialokasi atau memori yang sedang dipakai, dan saat pembebasan atau penghapusan memori tersebut akan dikembalikan sebagai memori yang dapat di pakai untuk alokasi memori selanjutnya, itu alasan kuat untuk mewajibkan anda membebaskan setiap memori yang sudah dipesan jika sudah tidak dperlukan kembali. Dan juga setelah dibebaskan pointer tidak secara otomatis menjadi null. Pointer masih menunjuk ke memori tersebut, kita masih dapat mengaksesnya bahkan mengubah nilai memori tersebut tetapi ingat bahwa memori tersebut sudah bukan milik anda. Kemungkinan anda bisa mengubah nilai memori milik program lain jika memori itu sudah mendapatkan pemiliknya yang baru. Untuk mencegah hal tersebut kita bisa membuat pointer tersebut menjadi null langsung setelah pembebasan memori.

Contoh Program :

#include <iostream>

#include <new> //beberapa fungsi Masih bekerja jika tidak menggunakan header ini

#define MAX\_ARR 5

using namespace std;

int main () {

int \*ptr = nullptr, \*ptrArr = nullptr; //NULL

cout <<"ptr = "<< ptr <<endl<<"ptrArr = "<< ptrArr <<endl<<endl; //NULL

ptr = new int; //Tunggal

ptrArr = new int[MAX\_ARR]; //Majemuk / Array

cout<<"ptr = "<<ptr<<" = "<<\*ptr<<endl<<endl;

for(int i=0;i<MAX\_ARR;i++){

cout<<"ptrArr ["<<i<<"] = "<<&ptrArr[i]<<" = "<<ptrArr[i]<<endl;

}

delete ptr; //Hapus Memori Tunggal

delete []ptrArr; //Hapus Memori Array

cout<<"Deleted !"<<endl;

return 0;

}

Diatas adalah contoh program penggunaan operasi delete. Diatas sudah cukup menjelaskan contoh dan cara penulisan operasi delete, pertama kita melakukan alokasikan dengan menggunakan operasi new untuk memori tunggal dan untuk memori dengan beberapa elemen (majemuk) atau dengan kata lain adalah array. dicontohkan satu demi satu hingga pada akhir baris yaitu penghapusan atau pembebasan memori sesuai bentuk dari memori tersebut.