[](https://1.bp.blogspot.com/-V777Mz4Vm5g/VrSqR3IyclI/AAAAAAAADDQ/TQbAjo7cIxg/s1600/pointer+pada+pointer.jpg)

Pointer ke pointer adalah salah satu contoh pemanfaatan fungsi dari pointer, pointer merupakan sebuah variabel yang berisi dari alamat variabel lainya atau dengan kata lain variabel pointer tersebut menunjuk ke variabel lainya. Pernah penulis katakan bahwa mendirikan variabel pointer dereference berbeda seperti reference, pointer dereference membutuhkan pemesanan memori baru pada komputer untuk mendirikanya yang berarti pointer tersebut juga mempunyai alamat memori sendiri pada komputer, dengan kata lain kita juga diperbolehkan untuk menunjuk ke variabel pointer lainya, bukan hanya variabel biasa. Untuk melakukan itu kita hanya membutuhkan tanda (\*) pada tingkatan pointer.

misalnya kita mempunyai variabel biasa dan pointer dengan satu tanda ( \* ) yang menunjuk ke variabel biasa itu. cara untuk membuat pointer untuk menunjuk ke pointer tersebut adalah dengan membuat variabel pointer yang mengunakan dua tanda (\*\*).

int var1 = 1;

int \*ptr1 = &var1;

int \*\*ptr2 = &ptr1;

di atas adalah contoh bagaimana penulisan untuk membuat pointer menunjuk ke pointer. dan di bawah ini merupakan presentasi menggunakan illustrasi berupa gambar dengan mencantumkan alamat memori.

================== Gambar 5.1

Seperti halnya pointer biasanya, variabel “var1” merupakan variabel biasa yang bertipe data integer yang berada pada memori 0x00123, “ptr1” merupakan variabel pointer yang menunjuk ke variabel “var1” dengan arti pointer tersebut menyimpan alamat memori dari “var1”, untuk pointer “\*ptr1” sendiri di lokasikan pada alamat memori 0x00321. dan pointer “\*\*ptr2” lah yang sedang kita bahas, seperti halnya pointer biasa pointer tersebut juga menyimpan alamat memori dari variabel lain yaitu variabel pointer “ptr1” pada alamat 0x00321, hal yang berbeda pointer “ptr2” dengan lainya adalah pada pemberian tanda ( \* ). Pointer “ptr2” memiliki 2 tanda ( \* ) yang berarti pointer tersebut akan menunjuk ke pointer lainya.

Pointer ke pointer memiliki kelebihan yang tidak hanya bisa mengubah nilai dari variabel yang di tunjuk melalui pointer. kita juga bisa bebas merubah nilai variabel yang di tunjuk termasuk pointer yang menunjuk langsung, kita bebas merubah variabel-variabel tersebut berdasarkan tingkatan. Memang sedikit membingungkan untuk di jelaskan secara langsung.

Misalnya, dari contoh pertama kita memiliki “var1”,” ptr1” dan “ptr2”. Untuk merubah atau membaca nilai “var1” kita tidak harus melalui variabel itu sendiri atau melalui pointer “ptr1” untuk mengakses. Kita juga bisa mengakses “var1” melalui pointer “ptr2”. Pointer ke pointer seperti “ptr2” kita bebas mengakses nilai mana yang akan diubah termasuk “var1” dan “ptr1”.

Contoh Program :

#include <iostream>

using namespace std;

int main( )

{

int var1 = 20;

int var2 = 99999;

int \*ptr99;

int \*ptr1 = &var1;

int \*\*ptr2 = &ptr1;

int \*\*\*ptr3 = &ptr2;

int \*\*\*\*ptr4 = &ptr3;

int \*\*\*\*\*ptr5 = &ptr4;

//dan Seterusya

cout<<&var1<<" = "<<var1<<endl;

cout<<&ptr1<<" = "<<ptr1<<endl;

cout<<&ptr2<<" = "<<ptr2<<endl;

cout<<&ptr3<<" = "<<ptr3<<endl;

cout<<&ptr4<<" = "<<ptr4<<endl;

cout<<&ptr5<<" = "<<ptr5<<endl;

\*\*\*ptr5=&ptr99;

\*\*\*\*ptr5=&var2;

cout<<&var1<<" = "<<var1<<endl;

cout<<&ptr1<<" = "<<ptr1<<" = "<<\*ptr1<<endl;

cout<<&ptr2<<" = "<<ptr2<<" = "<<\*ptr2<<" = "<<\*\*ptr2<<endl;

cout<<&ptr3<<" = "<<ptr3<<" = "<<\*ptr3<<" = "<<\*\*ptr3<<" = "<<\*\*\*ptr3<<endl;

cout<<&ptr4<<" = "<<ptr4<<" = "<<\*ptr4<<" = "<<\*\*ptr4<<" = "<<\*\*\*ptr4<<" = "<<\*\*\*\*ptr4<<endl;

cout<<&ptr5<<" = "<<ptr5<<" = "<<\*ptr5<<" = "<<\*\*ptr5<<" = "<<\*\*\*ptr5<<" = "<<\*\*\*\*ptr5<<" = "<<\*\*\*\*\*ptr5<<endl;

return 0;

}

Di atas adalah contoh program dari penggunaan pointer ke pointer. contoh di atas tidak hanya mencontohkan pointer tingkat dua (\*\*) tetapi juga mencontohkan hingga tingkat lima ( \*\*\*\*\* ). Mereka saling menunjuk berurutan dari “ptr5” sampai “var1” seperti halnya rantai yg saling berkaitan. Masing-masing pointer mempunyai alamat memori pointer yang mempunyai satu tingkatan yang lebih rendah dari pointer tersebut. Termasuk juga pada pointer “ptr1” yang menunjuk ke satu tingkatan lebih rendah yaitu variabel biasa “var1”.

Di atas juga dicontohkan bagaimana kita mengakses variabel melalui pointer yang mempunyai tingkatan lebih tinggi dari pointer yang menunjuknya. Di contoh atas terdapat pernyataan “ \*\*\*ptr5=&ptr99; ” yang berarti akan merubah isi dari variabel pointer “ptr2” tanpa melalui satu tingkatan lebih tinggi dari “ptr2” yaitu “ptr3”, di pernyataan itu kita ubah “ptr2” melalui pointer “ptr5” untuk menunjuk ke pointer “ptr99”.

Dan baris selanjutnya terdapat pernyataan “ \*\*\*\*ptr5=&var2; ” dengan arti akan merubah isi dari “ptr99”, merubah ” ptr99” untuk menunjuk ke variabel biasa yang telah kita buat yaitu “var2” melalui ponter “ptr5”.

Ingat pointer ke pointer itu seperti tingkatan. Jika kita ingin menunjuk ke variabel biasa, variabel pointer tersebut membutuhkan satu tanda dereference ( \* ) untuk menunjuk ke variabel biasa ( yang tanpa tanda \* ), begitu pula jika kita ingin menunjuk ke pointer tingkat satu tersebut yang mempunyai satu tanda ( \* ), berarti untuk tingkatan selanjutnya yang menunjuk ke pointer tersebut dibutuhkan dua tanda dereference ( \*\* ). Dan berlaku seterusnya.

Jika kita ingin mengubah arah tunjuk dari pointer bertingkat carilah pointer yang mempunyai satu tingkatan lebih rendah. Misal kita mempunyai “ \*\*\*ptr3 ” berarti kita harus mengisinya dengan pointer satu tingkat lebih rendah yaitu yang mempunyai dua tanda dereference ( \*\* ). Selain itu tidak akan bekerja.

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 : 5 Februari 2016