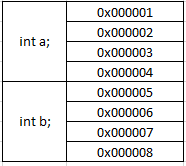
# **Pemrograman C++**

#### **Pengertian Pointer**

Pointer adalah penunjuk suatu variabel. Karena menunjuk suatu variabel, maka pointer wajib memiliki alamat dari variabel yang ditunjuknya. Kadangkala dalam program yang besar, penghematan memori wajib untuk dilakukan. Dengan mekanisme copy dan paste nilai variabel satu kedalam variabel lain, akan sangat memboroskan memori. Dengan mekanisme pointer, suatu variabel dalam suatu fungsi dapat diakses oleh fungsi yang lain.

#### **Alamat Dari Variabel**

Misalkan kamu memiliki variabel x dan terletak di memori 0x000001. Jika kamu ingin memasukkan nilai 100 kedalam variabel x, maka processor harus membawa nilai 100 tersebut kedalam variabel x yang terletak di alamat memori 0x000001. Hal yang perlu kamu ketahui adalah, setiap variabel ternyata memiliki ukuran byte yang berbeda-beda dalam memori. Sebagai contoh suatu variabel bertipe int memiliki ukuran 4 byte dalam memori. Maka variabel tersebut akan menempati 4 kapling lokasi dalam memori, misalkan 0x000001, 0x000002, 0x000003, dan 0x000004. Jika terdapat dua buah variabel bertipe int yang bersebelahan, maka alamat variabel pertama terletak di 0x000001 dan variabel kedua terletak dialamat 0x000005.



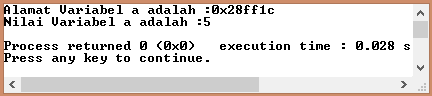
Memori menggunakan bilangan heksadesimal yang ditandai dengan awalan ‘0x’, sehingga jika suatu variabel menempati blok kesepuluh dalam memori, maka alamatnya adalah 0x00000a

#### **Deference (&)**

Deference (&) merupakan suatu operator yang berfungsi untuk menanyakan alamat dari suatu variabel. Apabila kamu memberikan simbol & pada awal variabel dan mencetak hasilnya pada jendela CLI, maka yang akan tercetak adalah alamat dari variabel tersebut bukan nilai yang ditampung oleh variabel tersebut. Berikut contoh programnya:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #include <iostream>  using namespace std;    int main(){  int a = 5;  cout<<"Alamat Variabel a adalah :"<<&a<<endl;  cout<<"Nilai Variabel a adalah :"<<a<<endl;  return 0;  } |

Hasil output programnya adalah:



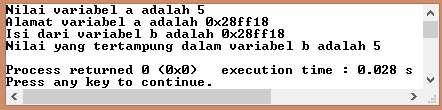
Alamat variabel ‘a’ pada setiap komputer akan berbeda-beda tergantung kompiler dalam mengalokasikan memori untuk suatu variabel.

#### **Reference (\*)**

Reference (\*) merupakan suatu operator yang berfungsi menyatakan suatu variabel adalah variabel pointer. Sama halnya dengan operator deference, peletakan simbol operator reference diletakan diawal variabel. Operator reference ini akan membuat suatu variabel pointer untuk menampung alamat. Nah untuk lebih mudahnya perhatikan contoh berikut:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | #include <iostream>  using namespace std;    int main(){  int a=5; //Memberikan nilai 5 pada variabel a  int \*b; //Mendeklarasikan variabel b sebagai pointer  b = &a; //Mengkopikan alamat variabel a kedalam variabel pointer b  cout<<"Nilai variabel a adalah "<<a<<endl;  cout<<"Alamat variabel a adalah "<<&a<<endl;  cout<<"Isi dari variabel b adalah "<<b<<endl;  cout<<"Nilai yang tertampung dalam variabel b adalah "<<\*b<<endl;    return 0;  } |

Hasil output programnya adalah:



Dari hasil output program di atas dapat dipahami bahwa, nilai variabel b sama dengan variabel a, walaupun variabel b tidak diberikan nilai yang sama dengan variabel a. Hal ini terjadi karena variabel pointer b menunjuk alamat variabel a dan variabel pointer b dapat mengakses nilai yang terkandung dalam alamat yang ditunjuknya.

#### **Pointer dan Array**

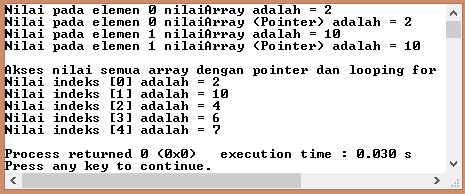
Tahukah kamu bahwa jika kamu menuliskan sebuah array tanpa tanda kurung kotak ([]) maupun indeksnya, maka array tersebut menunjuk atau bersi alamat elem pertama dari array tersebut. Misalkan kamu membuat sebuah pointer bertipe int dengan nama pointerArray dan sebuah array dengan tipe yang sama yaitu int dengan nama nilaiArray, maka pemberian nilai berikut sah dan dapat dilakukan:

|  |  |
| --- | --- |
| 1  2  3 | int \*pointerArray;  int nilaiArray[5];  pointerArray = nilaiArray; //memberikan alamat elemen pertama array pada variabel pointer |

Dari sintaks di atas, variabel pointer dengan nama pointrArray sekarang menunjuk pada alamat memori elemen pertama array dengan nama nilaiArray. Nah secara tidak langsung kita dapat mengakses nilai elemen array dengan menggunakan variabel pointer. sintaks pemrogramannya:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <iostream>  using namespace std;    int main(){  int nilaiArray[5] = {2,5,4,6,7};  int \*pointerArray;  pointerArray = nilaiArray;  cout<<"Nilai pada elemen 0 nilaiArray adalah = "<<nilaiArray[0]<<endl;  cout<<"Nilai pada elemen 0 nilaiArray (Pointer) adalah = "<<\*pointerArray<<endl;  pointerArray++; // naikan alamat yang ditunjuk pointer sejauh 1 integer  \*pointerArray = 10; //Ubah nilai elemen 1 array (nilaiArray) menjadi 10  cout<<"Nilai pada elemen 1 nilaiArray adalah = "<<nilaiArray[1]<<endl;  cout<<"Nilai pada elemen 1 nilaiArray (Pointer) adalah = "<<\*pointerArray<<endl;  pointerArray = nilaiArray; //kembalikan pointer pada elemen pertama  cout<<endl;  cout<<"Akses nilai semua array dengan pointer dan looping for"<<endl;  for (int i=0; i<5; i++){   cout<<"Nilai indeks ["<<i<<"] adalah = "<<\*(pointerArray+i)<<endl;  }  return 0;  } |

Hasil output programnya adalah sebagai berikut:



Dari hasil di atas, dapat kamu lihat bagaimana cara mengakses nilai array dengan menggunakan pointer.

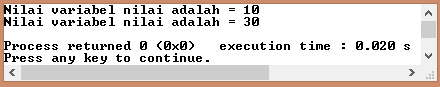
Suatu pointer bertipe int hanya dapat digunakan untuk menunjuk pada variabel int. Sebab, tipe data tersebut akan menentukan sifat pointer ketika alamat pointer tersebut dilakukan increment. Begitu juga untuk tipe data yang lain.

#### **Pointer Sebagai Parameter Suatu Fungsi**

Seperti halnya dengan array, pointer dapat digunakan sebagai parameter suatu fungsi. Karena sifat pointer yang hanya sebagai penunjuk, maka setiap perubahan yang terjadi pada parameter, sebenarnya terjadi pada variabel yang ditunjuk bukan pada variabel pointer. Berikut Mimin contohkan program sederhananya:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | #include <iostream>  using namespace std;    void tambah(int \*angka){  \*angka +=20;  }    int main(){  int nilai = 10;  cout<<"Nilai variabel nilai adalah = "<<nilai<<endl;  tambah(&nilai); //Memasukkan alamat variabel nilai pada fungsi tambah  cout<<"Nilai variabel nilai adalah = "<<nilai<<endl;  return 0;  } |

Berikut adalah output programnya:



Pada program diatas, Mimin membuat fungsi tambah, dimana ketika alama suatu variabel diberikan sebagai parameter fungsi, nilai 20 akan ditambahkan pada alamat tersebut. Perbedaan parameter berupa pointer dan non-pointer adalah, pada variabel non-pointer, ketika kamu ingin memasukkan nilai pada sebuah variabel kedalam parameter, kamu harus memasukkan nilai dari variabel tersebut. Untuk parameter berupa pointer, kamu cukup memasukkan alamat variabel yang menampung nilai dengan operator (&) kedalam parameter fungsi

Yang menarik adalah pada fungsi tambah di atas, walaupun fungsi tersebut merupakan fungsi void tanpa nilai kembalian alias return, namun nilai variabel parameter berubah.