[](https://4.bp.blogspot.com/-De-NjR-n26Q/VrTTBzoP9FI/AAAAAAAADEU/AjCWwUx2RxE/s1600/pointer+pada+string.jpg)

Pointer sebagai string adalah pemanfaatan penggunaan urutan alamat memori yang berdekatan dari pointer sebagai uturan atau kumpulan dari karakter yang menjadikaya sebagai string. Seperti halnya array, tipe data char dengan pointer juga bisa berlaku seperti halya char dengan array. String merupakan kumpulan dari beberapa karakter dan char ( Character ) adalah salah satu tipe data dari bahasa pemrograman berfungsi untuk menyimpan data dalam bentuk karakter seperti a-z dan tanda karakter lainya di dalam memori komputer. Di bawah merupakan contoh program dari perbandingan antara char[] dan char\*. String Array vs Pointer :

#include <iostream>

using namespace std;

int main( )

{

char arr[]="IniBelajarCPP";

char \*ptr=arr;

cout<<"Eksekusi Array : "<<endl;

cout<<arr<<endl;

cout<<arr[3]<<endl; //Subscript

cout<<arr+3<<endl; //Aritmatika

cout<<\*(arr+3)<<endl; //Aritmatika

cout<<"Eksekusi Pointer : "<<endl;

cout<<ptr<<endl;

cout<<ptr[3]<<endl; //Subscript

cout<<ptr+3<<endl; //Aritmatika

cout<<\*(ptr+3)<<endl; //Aritmatika

return 0;

}

Sesuai apa yang baru saja anda lihat, bahwa array dan pointer dalam string hamper tidak ada perbedaan. Bisa dibilang mereka adalah sama. Pada artikel ini penulis akan membahas tentas mengenai fungsi, penggunan, dan semua penjelasan yang berhubungan mengenai char/string array dan pointer.

String dengan Array

Merupakan kumpulan dari karakter yang dilokasikan pada alamat memori yang berdekatan dan menjadikanya sebagai string.

Bentuk Umum Penulisan :

char identifier[index];

Contoh Penulisam :

char arrString[15] = “belajarcpp.com”;

char arrString[] = “belajarcpp.com”;

pernyataan itu menyatakan bahwa memesan memori sebesar char (1 Byte) dengan nama variabel arrString. Sebanyak 15 digit/lokasi memori. Ingat, saat anda mendirikan variabel array anda bisa mengosongi jumlah index “ [] ” jika anda langsung memberi nilai awal (inisialisasi) pada variabel tersebut. Jika tidakanda wajib untuk mengisi jumlah index untuk memberi jumlah elemen yang akan di pesan. Dan ingat bahwa mendirkan variabel string otomatis akan dibutuhkan karakter “ null ” atau “ \o ” pada akhir elemen.

----------Gambar 6.1

| B | e | l | a | j | a | r | c | p | p | . | c | o | m | \0 |

Dan saat kita mendirikan variabel array dan mengisi index, memesan berapa jumlah elemen. Maka akan terjadi seperti ini

int arrString[20] = “Belajarcpp.com”;

| B | e | l | a | j | a | r | c | p | p | . | c | o | m | \0 | | | | | | |

Cara Mengakses String Array

Fungsiya dan pengaksesan sama seperti array biasa, kita dapat mengakses setiap karakter dalam array menggunakan notasi subscript, seperti

cout<<arrString[5] ;

pernyataan di atas akan mencetak karakter ‘ j ’. dan penggunaan array dan pointer pada tipe data char akan berlaku special pada pengaksesan, memanggil identifier tanpa menggunakan notasi subscript. Pada artikel “Pointer dengan Array” dijelaskan bahwa jika kita memanggil sebuah array tanpa notasi subscript selain tipe data char akan menghasilkan alamat memori oleh elemen pertama dari array. Berbeda dengan array/pointer, mereka akan mencetak semua karakter yang ada pada semua elemen. Dan pernyataan di bawa ini akan mencetak “belajarcpp.com”.

cout<<arrString;

mereka akan mencetak semua karakter yang ada pada semua elemen. Dan pernyataan di bawa ini akan mencetak “belajarcpp.com”.

kekurangan string array adalah bahwa anda diwajibkan untuk menyatakan berapa jumlah karakter yang akan dipesan/digunakan pada index untukjumlah elemen. Memang kita bisa mengosonginya tetapi setelah itu kita harus memberinya nilai awal ( inisialisasi ), bukan berarti jika kita mengosonginya kita mendapatkan jumlah karakter yang tidak terbatas. Kita akan mendapatkan jumlah elemen sebanyak jumlah karakter pada nilai awal ( inisialisasi ).

char arrString[20] = “belajarcpp.com”; //15 karakter dan memiliki 5 elemen kosong.

char arrString[] = “belajarcpp.com”; //15 karakter dengan null pada akhir.

Pernyataan pada baris pertama didirikan variabel string dengan jumlah index 20, diberikan nilai awal sebanyak 14 karakter ditambah 1 karakter null, dengan arti masih mempunyai 5 elemen yang masih kosong. Dan baris kedua didirikan variabel yang sama namun kita tidak memberi jumlah index dengan arti akan menyesuaikan otomatis dengan nilai awal yang berjumlah 14+1. Dengan demikian kita diwajibkan untuk meentukan jumlah maksimum karakter yang anda butuhkan untuk menyimpan dalam array. Jenis alokasi array dimana ukuran array ditentukan pada saat kompilasi disebut Static Allocation. Dengan itu kita akan direpotkan jika kita ingin memberi nilai ulang, lebih dari jumlah krakter yang sudah dipesan.

String dengan Pointer

Sudah dijelaskan bahwa pointer sangat mirip dengan array terutama pada string. Membuat beberapa memori yang saling berdekatan. Tapi, pointer hanya memegang alamat, mereka tidak dapat menampung semua karakter dalam array karakter. Ini berarti bahwa ketika kita menggunakan char \* untuk melacak string array karakter yang mengandung string harus sudah ada yang secara static atau dinamis,

Bentuk Umum Penulisan :

char \*identifier;

Contoh Penulisan :

char \*ptrString = arrString;

pada contoh diatas didirikan string pointer dengan nilai / menunjuk ke string yang sudah ada sebelumnya pada string array. Dan di bawah ini adalah gambar illustrasi apa yang sebenarnya terjadi.

---------------------------Gambar 6.2

Ingat, jika pemberian tugas atau inisialisasi pada pointer dengan array kita tidak menggunakan operator Address-of ( & ). Bisa kita buktikan dengan mencetak variabel array selain tipe data char tanpa menggunakan notasi subscript / index untuk penentu elemen, apa yang terjadi adalah program akan mencetak alamat memori dari elemen pertama pada array tersebut begitupula dalam operasi penugasan. Ini akan berlangsung berbeda jika kita menggunakan tipe data char, apa yang terjadi akan mencetak semua isi elemen danmenjadikanya sebuah string. Walaupun seperti itu, ini akan berlaku sama dalam operasi penugasan. Variabel pointer hanya akan mendapatkan alamat memori dari elemen pertama array.

Dan setelah pemberian penugasan, pengaksesan ptrString akan bersifat sama seperti pengaksesan array. Untuk itu bisa kita lihat pada contoh program yang penulis berikan pada paragraph pertama. Itu karena array dan pointer itu sama.

jika salah satu dari kalian sudah pernah mendirikan variabel pointer string dan langsung memberi nilai awal tanpa melacak/menunjuk alamat string yang sudah ada.

char \*ptrString2 = “belajarcpp.com”;

itu bisa saja dilakukan, dengan cara kerja, string “belajarcpp.com” akan otomatis di lokasikan pada alamat memori komputer dan setelah itu variabel pointer akan menunjuk karakter pertama pada string tersebut. Bukan berarti bahwa pointer tersebut mempunyai niai string. tetapi dari pernyataan tersebut di beberapa compiler seperti codeblock akan menghasilkan pesan, bukan error namum peringatan, mungkin pesan-pesan yang akan anda dapat akan berbeda-beda pada masing-masing compiler yan anda gunakan, dan di bawah ini merupakan pesan peringatan yang penulis dapat pada compiler codeblock.

Warning: deprecated conversion from string constant to ‘ char\* ’ [-Wwrite-strings]

Yang berarti “belajarcpp.com” merupakan konstanta dan string tersebut disimpan dalam lokasi tertentu pada alamat memori yang biasanya hanya bisa dibaca ( read-only ). kesimpulanya kita tidak bisa mengubah string tersebut.

ptrString2 = “BelajarC++”;

di atas merupakan pernyataan operasi penugasan pada pointer yang telah didirikan di atas. Seperti yang telah di jelaskan, meskipun pernyataan penugasan itu berhasil di jalankan bukan berarti anda mengubah string “belajarcpp.com” dengan “BelajarC++”. String pertama akan selalu ada dan tidak digantikan dengan string yang baru, pointer hanya mengganti arah tunjuk pointer ke string yang kita berikan.

Dari penjalasan di atas bukan berarti kita tidak dibolehkan menggunakan pernyataan variabel string dengan pointer seperti pernyataan di atas. Saran penulis terutama pada assignment / operator penugasan kita harus berhati-hati dengan alamat memori terutama ponter. Setiap memori yang sudah dilokasikan dan tidak akan digunakan kembali, Disarankan untuk di hapus terlebih dahulu agar tidak terjadi kebocoran memori / memory leak. Dan akan lebih baik lagi jika sebelum pointer digunakan sebaiknya memori yang akan digunaan dialokasikan dahulu, kemudian gunakan fungsi strcpy untuk mengarahkan pointer ke string. Setelah memori tersebut sudah tidak digunakan lagi sebaiknya dihapus. Untuk penjelasan alokasi dan penghapusan memori akan di jelaskan pada artikel berikutnya.

Kesimpulan Array vs Pointer

Seperti yang telah dijelaskan, meskipun sama tetapi fungsi utama dari mereka berbeda. Singkatnya setiap arra akan di decay menjadi pointer setelah terkompilasi tapi tidak sebaliknya.

Mungkin kalian juga tau bahwa pemrograman C++ juga menyediakan tipe data string untuk mempermudah kita dalam hal ini.

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