

[Home](#) / [Tutorial](#) / [C](#) / C-Style String

## C-Style String

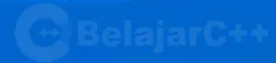
Posted on Januari 23, 2018 | Last Modified Januari 23, 2018

```
using namespace std;

int main(){
    char myString[] = "string";

    int panjangElemen = sizeof myString;
    int panjangKalimat = strlen(myString);

    cout<<"====="<<"<<myString<<endl;
    cout<<"Panjang Elemen : "<<panjangElemen<<endl;
    cout<<"Panjang Kalimat : "<<panjangKalimat<<endl;
}
```



String adalah kumpulan dari beberapa karakter. Dalam C++ terdapat dua jenis string, yaitu `std::string` dan C-Style string yang masing-masing memiliki fungsi untuk membuat penyimpanan yang dapat menyimpan text.

C-style string merupakan string yang berasal dari bahasa pemrograman C untuk menangani penyimpanan text. dan `std::string` berasal dari standar perpustakaan C++, yang merupakan evolusi c-style string yang lebih mudah digunakan. Dalam artikel ini kita akan membahas mengenai C-style string.

### Bentuk Penulisan C-style String

```
char indentitas[ukuran] = inisialisasi;
```

C-style string adalah kumpulan array dari karakter yang diakhiri dengan null terminator. Setiap kita melakukan penyimpanan pada C-style string, setiap satu karakter akan menduduki satu elemen dari array, dan banyaknya array ditentukan oleh seberapa banyak karakter dari dalam

string tersebut tapi kita juga dimungkinkan untuk memesan seberapa besar array untuk c-style string.

Setiap string dari c-style string yang telah kita dirikan akan diakhiri oleh sebuah karakter spesial yaitu /0 atau null, dalam kode ASCII memiliki kode 0. di dalam c-style string karakter tersebut diberi nama Null terminator. Dan null terminator berfungsi untuk mengindikasikan akhir dari sebuah string.

Dari pengertian di atas, C-Style String juga biasa disebut sebagai Null Terminated String.

#### Contoh Penulisan

```
char myString[] = "string";
```

C-style string pada dasarnya memang merupakan array. Semua peraturan pada array yang telah dijelaskan penulis berlaku juga pada C-style string.

Pada contoh di atas kita mendirikan sebuah string, dengan inisialisasi dan tanpa menyertakan jumlah elemen, hal itu dimungkinkan pada peraturan array. Pada pemesanan variabel string di atas, besar dari memori (atau jumlah elemen) akan otomatis ditentukan oleh CPU berdasarkan panjang dari nilai inisialisasi ditambah satu null terminator pada akhir string.

Pada variabel di atas memiliki panjang 7 elemen. 6 elemen yang berisi masing-masing karakter dari kata "string" dan satu null terminator pada akhir string. Gambar di bawah merupakan ilustrasi dari alokasi memori dari variabel C-style string.



*Elemen C-Style String*

Jika kita mendirikan variabel c-style string dengan menyertakan penentuan jumlah elemen berdasarkan seberapa yang kita butuhkan, Kita wajib menyiapkan satu elemen untuk null terminator.

#### Contoh Program :

```
#include <iostream>
#include <cstring> //untuk strlen
using namespace std;

int main(){
    char myString[] = "string";

    int panjangElemen = sizeof(myString) / sizeof(myString[0]);
    //mendapatkan jumlah elemen
    int panjangKalimat = strlen(myString); //mendapatkan jumlah
    karakter sebenarnya

    cout<<"===== "<<myString<<" : ====="<<endl;

    cout<<"Panjang Elemen : "<<panjangElemen<<endl;
    cout<<"Panjang Kalimat : "<<panjangKalimat<<endl;

    cout<<endl<<"=> ASCII code"<<endl;
    for (int i = 0; i < panjangElemen; ++i){
        cout<<myString[i]<<" : "<< static_cast<int>(myString[i])
<<endl; //Melihat Kode ASCII
    }

    return 0;
}
```

### Cara Mengakses dan Memberi nilai C-Style string

C-style string tidak memiliki banyak perbedaan dengan variable array biasa. Hanya saja saat mengakses sebuah c-style string kita bisa langsung melakukannya dengan memanggil identitas dari variable tersebut menggunakan atau tanpa menggunakan operator subscript.

Pada artikel sebelumnya, jika ketika kita mengakses variable array biasa tanpa menggunakan operator subscript, kita hanya akan mendapatkan alamat memori elemen awal dari variable array tersebut. Tapi berbeda dengan c-style string, ketika kita tidak menggunakan operator subscript apa yang akan kita dapatkan adalah nilai keseluruhan dari array yang membuatnya menjadi sebuah string, Dan ketika kita menggunakan operator subscript pada c-style string, kita akan mendapatkan satu nilai elemen dari karakter pada string tersebut.

Contoh Program :

```
#include <iostream>
using namespace std;

int main(){
    char myString[] = "string";

    cout<<myString<<endl;
    cout<<myString[1]<<endl;

    return 0;
}
```

dan peraturan lainnya yang perlu kamu tau adalah C-style string tidak bisa diberi nilai menggunakan Assignment Operator (=) secara langsung tanpa menggunakan subscript operator. Pada contoh di atas kita bisa mencetak sebuah string hanya memanggil nama dari variabel tanpa subscript operator, dan kita juga bisa inisialisasi tanpa menggunakan tanda koma untuk memisahkan masing-masing nilai elemen.

```
char myString[] = "string" //Ok
cout<<myString; //Ok
myString = "new change"; //Error
myString[2] = 'a'; //Ok
cin>>myString; //Ok
```

meskipun kita tidak bisa mengubah nilai menggunakan operator penugasan, tapi kita masih bisa mengubah nilai string menggunakan perintah seperti input cin, merubah nilai setiap elemen satu demi satu atau kita bisa menggunakan manipulasi string seperti strcpy(). Dan di bawah ini adalah contoh program salah satu cara kita mengubah nilai dari C-style string menggunakan input cin.

Contoh Program

```
#include <iostream>
using namespace std;

int main()
{
    char nama[] = "Belajar C++";
    cout<<"Sebelum : "<<nama<<endl;
    std::cout << "Ubah : ";cin>>nama;
    cout << "Sesudah : " << nama <<endl;

    return 0;
}
```

contoh di atas adalah salah satu cara kita untuk mengubah nilai dari C-Style String, dan masih ada banyak cara untuk mengubah nilai dari c-style string, hal itu akan penulis jelaskan pada artikel selanjutnya.

Setelah penjelasan di atas, ada beberapa masalah yang perlu kalian kalian tau. Contoh program di atas meminta kita untuk memasukan nilai baru pada variabel 'nama', dan nilai yang akan kita inputkan hampir tidak di batasi, dan kita tau bahwa c-style string selalu terbatas. Masalahnya adalah jika kita mengubah nilai c-style string lebih dari ukuran maka otomatis kita juga mengubah alamat memori setelah lokasi memori c-syle string, seperti yang telah dijelaskan pada artikel sebelumnya bahwa hal itu sangat tidak disarankan. Dan ada satu masalah lagi yang lebih buruk dari hal tersebut, yaitu jika Null terminator berubah maka CPU akan terus membaca hingga CPU menemukan Null terminator, tidak peduli seberapa jauh itu.

Meskipun kita mencoba memesan memori pada subscript c-syle string sebanyak apapun, hal itu masih tidak cukup untuk mengatasi masalah tersebut di masa depan. Dan juga kemungkinan kita akan membuat program yang memakan memori cukup besar pada komputer.

Dimana ada masalah pasti ada solusi. Untuk mengatasi masalah tersebut kita dapat menggunakan fungsi `cin.getline`, merupakan fungsi yang memungkinkan kita mengatur batas banyak karakter yang akan dimasukan dalam variabel.

Contoh Program :

```
#include <iostream>
#include <cstring>
using namespace std;

int main(){
    char nama[25] = "Nama Anda belum di isi";
    cout<<"Nama :";cin.getline(nama, sizeof(nama)-1);
    cout<<"Nama anda adalah : "<<nama<<endl;

    return 0;
}
```

selain masalah itu, cin.getline juga dapat memungkinkan kita memasukan karakter sepasi. pernahkah anda mencoba untuk menginputkan sepasi dengan input cin biasa atau printf ke dalam c-style string. CPU tidak memasukan sepasi (dalam ASCII adalah 32) melainkan sebagai Null terminator (0). saat pembacaan itu dapat menyebabkan string terpotong.

#### Kesimpulan :


C++ merupakan evolusi dari bahasa pemrograman C. dan dalam C++ kita masih dimungkinkan untuk menggunakan fitur dari C. kita tau bahwa kita masih dimungkinkan menggunakan C-Style string, tapi bagi penulis menyarankan untuk menggunakan string dari C++. Pembuat bahasa membuatnya agar lebih mudah untuk digunakan dan juga C++ string memiliki banyak kelebihan dari pada C-style string.

Penulis menyarankan untuk tidak menggunakan bukan berarti untuk dilupakan atau dihindari. Penulis menyarankan jika ingin menggunakan c-style string sebaiknya anda benar-benar memiliki tujuan untuk menggunakannya.

 Share

 Tweet

 Share

 Pin

 Share

---

#### Terkait

Escape Sequences  
Januari 22, 2018  
dalam "C++"

Dasar Input Output pada C  
Januari 22, 2018  
dalam "C"

Struktur dan Bagian-Bagian  
Dasar C++  
Januari 22, 2018  
dalam "C++"

Array Multidimensi

Januari 23, 2018

dalam "C++"

Tipe Data

Januari 22, 2018

dalam "C++"

Array

Januari 23, 2018

dalam "C++"