

# Bài 37: MẢNG KÝ TỰ TRONG C++ (C-STYLE STRINGS)

Xem bài học trên website để ủng hộ Kteam: [Mảng ký tự trong C++ \(C-style strings\)](#)

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage [How Kteam](#) nhé!

## Dẫn nhập

Ở bài học trước, mình đã chia sẻ cho các bạn về [CÁC THAO TÁC TRÊN MẢNG 2 CHIỀU](#) trong C++.

Hôm nay, mình sẽ giới thiệu cho các bạn về **Mảng ký tự trong C++ (C-style strings)**.

---

## Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về:

- [KIỂU KÝ TỰ TRONG C++ \(Character\)](#)
- [MẢNG 1 CHIỀU \(Arrays\)](#)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Mảng ký tự (C-style strings) là gì?
- Khai báo và khởi tạo mảng ký tự (C-style strings)
- Xuất mảng ký tự (C-style strings) với `std::cout`
- Nhập mảng ký tự (C-style strings) với `std::cin`

# Mảng ký tự (C-style strings) là gì?

Trong bài học [CƠ BẢN VỀ CHUỖI KÝ TỰ TRONG C++ \(An introduction to std::string\)](#), bạn đã biết được **chuỗi ký tự** là tập hợp các ký tự **tuần tự**, được đặt trong dấu ngoặc kép. Dùng để biểu diễn những thông báo, văn bản, ... trong chương trình.

**Ví dụ:** "Hello, HowKteam.com!" chính là một chuỗi ký tự.

Ngôn ngữ C++ có 2 loại chuỗi ký tự khác nhau:

- Chuỗi ký tự **std::string** được cài đặt trong một lớp của **thư viện chuẩn STL**.
- Chuỗi ký tự **C-style** nguyên bản từ ngôn ngữ C.

Kiểu chuỗi ký tự **std::string** được xây dựng từ **chuỗi ký tự C-style**. Vì vậy, chuỗi ký tự **std::string** thường được sử dụng trong C++ vì tính **đơn giản**, và **dễ sử dụng** của nó.

Trong bài học này, chúng ta sẽ tìm hiểu bản chất và cách sử dụng của kiểu **C-style strings**.

**Chuỗi ký tự C-style** bản chất là **mảng 1 chiều các ký tự**, kết thúc bằng ký tự **'\0' (null)**. Hay còn gọi là **null-terminated string**.

**Ví dụ:**

```
char sz[] = "Kteam";
```

Hình bên dưới mô tả 1 C-style string tên là **sz** có kiểu **char** gồm **6 phần tử** (5 ký tự thường, và 1 ký tự null '\0') nằm trong vùng nhớ RAM:

```
char sz[] = "Kteam";
```

```
sz[0] sz[1] sz[2] sz[3] sz[4] sz[5]
```

'K'	't'	'e'	'a'	'm'	'\0'
-----	-----	-----	-----	-----	------

Chuỗi ký tự **sz** được khởi tạo 5 phần tử, nhưng chuỗi ký tự C-style luôn mặc định kết thúc bằng ký tự **null '\0'**, nên chuỗi sz sẽ có 6 phần tử.

## Khai báo và khởi tạo mảng ký tự (C-style strings)

### Khai báo mảng ký tự (C-style strings)

Cú pháp khai báo tương tự như cách khai báo mảng 1 chiều.

#### Ví dụ:

```
char szFullName[30]; // Dài 29 ký tự  
char szDayOfBirth[9]; // Dài 8 ký tự
```

**Chú ý:** Mảng ký tự kết thúc bằng ký tự '\0' (null) => **độ dài chuỗi = kích thước mảng - 1**

## Khởi tạo giá trị cho mảng ký tự (C-style strings)

**Cách 1:** Khởi tạo với độ dài cụ thể.

```
char sz[10] = { 'K', 't', 'e', 'a', 'm', '\0' };  
char sz[10] = "Kteam"; // Tự động thêm '\0' vào cuối chuỗi
```

```
char sz[10] = { 'K', 't', 'e', 'a', 'm', '\0' };  
char sz[10] = "Kteam";
```

0	1	2	3	4	5	6	7	8	9
'K'	't'	'e'	'a'	'm'	'\0'	'\0'	'\0'	'\0'	'\0'




**Cách 2:** Khởi tạo tự động xác định độ dài.

```
char sz[] = { 'K', 't', 'e', 'a', 'm', '\0' };
char sz[] = "Kteam"; // Tự động thêm '\0' vào cuối chuỗi
```

```
char sz[] = { 'K', 't', 'e', 'a', 'm', '\0' };
char sz[] = "Kteam";
```

0	1	2	3	4	5
'K'	't'	'e'	'a'	'm'	'\0'



Bản chất của mảng ký tự cũng là mảng 1 chiều, nghĩa là bạn có khởi tạo chuỗi ký tự, nhưng bạn không thể gán chuỗi ký tự trực tiếp cho nó.

**Ví dụ** các phép gán hợp lệ:

```
char sz[] = "Kteam"; // Tự động thêm '\0' vào cuối chuỗi
sz[0] = 'F';
sz[1] = 'r';
sz[2] = 'e';
sz[3] = 'e';
sz[4] = 'E';
```

**Ví dụ** các phép gán **KHÔNG** hợp lệ:

```
char sz[] = "Kteam"; // Tự động thêm '\0' vào cuối chuỗi
sz = "FreeE"; // Lỗi, không được gán trực tiếp
```

## Xuất mảng ký tự (C-style strings) với `std::cout`

Không giống như mảng 1 chiều thông thường cần sử dụng vòng lặp để xuất từng phần tử, mảng ký tự cho phép sử dụng đối tượng `std::cout` để in toàn bộ ký tự ra màn hình.

Khi in **mảng ký tự (C-style strings)**, đối tượng **std::cout** sẽ in tất cả ký tự cho đến khi gặp ký tự **'\0' (null)**.

### Ví dụ:

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

int main()
{
    // 0 1 2 3 4 5
    // K t e a m \0
    char szKteam[] = "Kteam";
    cout << szKteam << endl; // "Kteam"

    // 0 1 2 3 4 5 6 7 8 9
    // F r e e E d u \0 \0 \0
    char szFreeEdu[10] = "FreeEdu";
    cout << szFreeEdu << endl; // "FreeEdu"

    // ký tự szFreeEdu[7] là '\0' (null), nên chỉ in chuỗi đến vị trí 6
    // 0 1 2 3 4 5 6 7 8 9
    // F r e e E d u \0 \0 a
    szFreeEdu[9] = 'a';
    cout << szFreeEdu << endl; // "FreeEdu"

    return 0;
}
```

### Output:



# Nhập mảng ký tự (C-style strings) với std::cin

**Mảng ký tự (C-style strings)** trong C++ cho phép sử dụng đối tượng **std::cin** để nhập dữ liệu từ bàn phím. Nhưng việc sử dụng trực tiếp đối tượng **std::cin** sẽ gặp **2 vấn đề**.

## Vấn đề 1: không nhập được chuỗi kí tự có chứa khoảng trắng

Khi đọc thông tin từ bàn phím, đối tượng **std::cin** sẽ đọc các ký tự cho đến khi gặp ký tự **khoảng trắng ' '**, hoặc ký tự **enter '\n'**.

Vì vậy, nếu chuỗi của bạn chứa các khoảng trắng, đối tượng **std::cin** chỉ đọc được **từ đầu tiên**.

### Ví dụ:

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

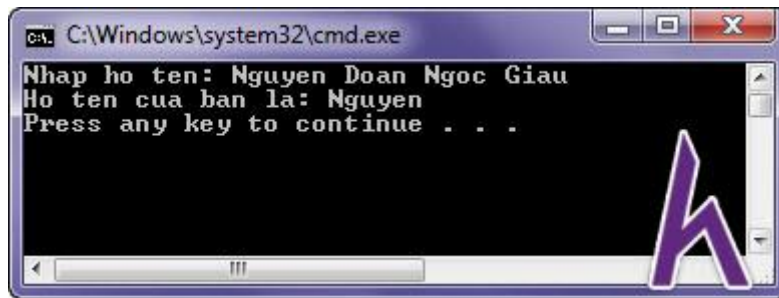
int main()
{
    char szFullName[50];

    cout << "Nhap ho ten: ";
    cin >> szFullName;

    cout << "Ho ten cua ban la: " << szFullName << endl;

    return 0;
}
```

### Output:



## Vấn đề 2: tràn mảng khi nhập quá số lượng ký tự so với khai báo

Bạn không thể kiểm soát được các thao tác của người dùng. Trường hợp bạn khai báo 1 mảng ký tự gồm 10 phần tử, nhưng người dùng cố tình (hoặc vô ý) nhập 1 chuỗi nhiều hơn 10 phần tử. Lúc này, vấn đề **tràn mảng** xảy ra, và chương trình của bạn sẽ gặp lỗi.

### Ví dụ:

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

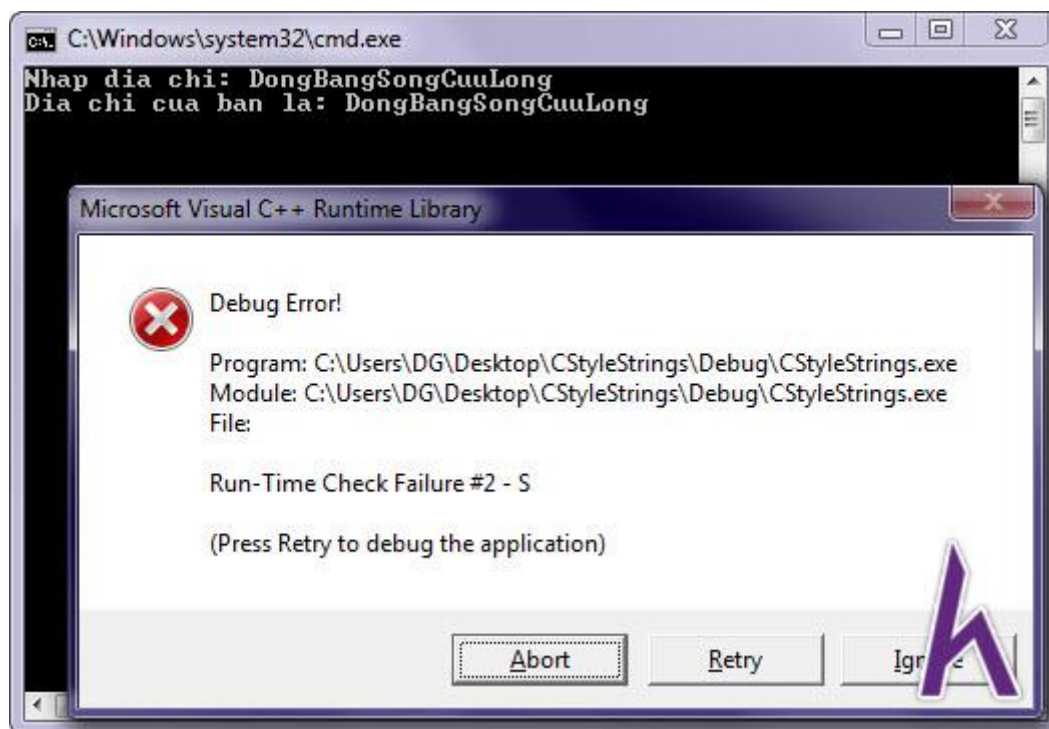
int main()
{
    char szAddress[10];

    cout << "Nhap dia chi: ";
    cin >> szAddress;

    cout << "Dia chi cua ban la: " << szAddress << endl;

    return 0;
}
```

### Output:



## Giải quyết 2 vấn đề nhập chuỗi bằng phương thức `std::cin.getline()`

Hàm **`std::cin.getline()`** sẽ đọc tất cả các ký tự từ bàn phím (**bao gồm khoảng trắng ' '**) cho đến khi gặp **ký tự enter 'n'** (mặc định). Nếu số lượng ký tự nhập vào lớn hơn **độ dài truyền vào hàm**, mọi **ký tự dư thừa sẽ bị loại bỏ**.

### Ví dụ:

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

int main()
{
    char szAddress[20];

    cout << "Nhập địa chỉ: ";
    cin.getline(szAddress, 20);

    cout << "Địa chỉ của bạn là: " << szAddress << endl;
```



```
    return 0;  
}
```

### Output:



Với cách này, bạn sẽ giải quyết được vấn đề nhập khoảng trắng và đảm bảo không bị tràn mảng.

## Kết luận

Qua bài học này, bạn đã biết được bản chất và cách sử dụng [Mảng ký tự trong C++ \(C-style strings\)](#).

Trong bài tiếp theo, mình sẽ giới thiệu cho các bạn về [CÁC THAO TÁC TRÊN MẢNG KÝ TỰ \(C-style strings\)](#) trong C++.

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên **"Luyện tập – Thử thách – Không ngại khó"**.