**PHẦN 1**

**TỔNG QUAN VỀ CHUỖI KÝ TỰ (STRING)**

Bạn đã được tiếp xúc với chuỗi ký tự (string) ngay từ bài học C++ đầu tiên qua chương trình kinh điển bên dưới:

#include <iostream>

**using** **namespace** **std**;

**int** main()

{

 **cout** << "Hello, HowKteam.com!" << endl;

 **return** 0;

}

**Chuỗi ký tự** là tập hợp các ký tự được đặt trong dấu ngoặc kép. Dùng để biểu diễn những thông báo, văn bản, … trong chương trình. Trong chương trình trên, "Hello, HowKteam.com!" chính là một chuỗi ký tự.

Trong C++, kiểu **chuỗi ký tự không được xây dựng sẵn** (không phải là “built-in string”) mà được cài đặt trong một lớp của **thư viện chuẩn STL**(**C++ Standard Template Library**).

**1. Khai báo, khởi tạo và gán giá trị một chuỗi ký tự**

Để sử dụng chuỗi trong C ++, bạn cần phải **#include** thư viện **string** thuộc **namespace std**.

#include <string>

// …

**std**::**string** strMyName;

Bạn có thể sử dụng **namespace std** để có thể sử dụng string một cách ngắn gọn hơn:

#include <string>

**using** **namespace** **std**;

// …

**string** strMyName;

Tương tự như các biến thông thường, bạn có thể khởi tạo hoặc gán giá trị cho biến string theo nhiều cách:

/\* Khai báo một chuỗi ký tự \*/

**string** strString;

/\* Khởi tạo một chuỗi theo nhiều cách \*/

**string** strString0("");                 // Khởi tạo chuỗi strString0 rỗng

**string** strString1("Kteam");            // Khởi tạo chuỗi "Kteam" cho strString1

**string** strString2{ "Free Education!" }; // Khởi tạo chuỗi "Kteam" cho strString2

**string** strString3 = "HowKteam.com";    // Khởi tạo chuỗi "HowKteam.com" cho strString3

**string** strString4 = strString1;         // Gán giá trị chuỗi strString1 cho strString4

/\* Khởi tạo một chuỗi số \*/

**string** strNumber = "88";         // Khởi tạo chuỗi "88" cho strNumber, không phải số

**Chú ý:**Khi khởi tạo giá trị là số cho một chuỗi, chuỗi đó không được coi là một số, và không có những thao tác như một biến số học ( cộng, trừ, nhân, chia …). **C++ không tự động chuyển một chuỗi số về giá trị số nguyên (integer) hoặc số chấm động (floating point)**.

**2. Xuất một chuỗi ký tự (string output):**

Bạn đã biết cách nhập xuất dữ liệu trong C++. Đối với chuỗi ký tự, thao tác xuất cũng tương tự:

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

     /\* Khởi tạo một chuỗi HowKteam.com cho biến strString \*/

     **string** strString("HowKteam.com");

     /\* Xuất chuỗi HowKteam.com lên console \*/

     **cout** << strString << endl;

     **return** 0;

}

Kết quả khi chạy chương trình là

****

**3. Nhập một chuỗi ký tự (string input)**

Đối với các kiểu dữ liệu cơ bản, bạn sử dụng đối tượng**std::cin** để **đọc một thông tin nào đó từ thiết bị nhập chuẩn** (mặc định là bàn phím), **sau đó lưu thông tin đó vào một biến.**

Cùng thử với kiểu dữ liệu std::string:

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **cout** << "Enter your full name: ";

       **string** strName;

       **cin** >> strName;

       **cout** << "Enter your age: ";

       **string** strAge;

       **cin** >> strAge;

       **cout** << "Your name is " << strName << endl;

       **cout** << "Your age is " << strAge << endl;

       **return** 0;

}

Kết quả khi chạy chương trình là



Trong chương trình trên, khi nhập chuỗi “Hello HowKteam.com! Free Education!” và nhấn enter, chương trình đã bỏ qua lần nhập tiếp theo và xuất ra kết quả.

**Chú ý:**Toán tử **>> (extraction operator)**được dùng chung với **std::cin** chỉ cho phép đọc một dãy gồm các ký tự liền nhau (không có khoảng trắng), nên nó sẽ trả về các ký tự đến khoảng trắng đầu tiên trong chuỗi vừa nhập.

Vì vậy, chương trình có 2 lần yêu cầu nhập chuỗi, nhưng khi bạn nhập một chuỗi có 3 khoảng trắng như trên, nó chỉ cho phép bạn nhập vào 1 lần.

**4. Nhập văn bản bằng getline()**

Để đọc đầy đủ chuỗi có khoảng trắng từ đối tượng nhập của lớp **iostream**(ví dụ cin), bạn nên sử dụng hàm getline() (**trong namespace std**)

**Ví dụ:**

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **cout** << "Enter your full name: ";

       **string** strName;

       getline(**cin**, strName);

       **cout** << "Enter your age: ";

       **string** strAge;

       getline(**cin**, strAge);

       **cout** << "Your name is " << strName << endl;

       **cout** << "Your age is " << strAge << endl;

       **return** 0;

}

Kết quả khi chạy chương trình là



Ở chương trình trên, bạn đã nhập được chuỗi ký tự bao gồm cả khoảng trắng với hàm **getline()**.

**5. Kết hợp giữ cin và getline() sẽ gây ra kết quả không mong muốn**

**Ví dụ:**

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **cout** << "Enter your age: ";

       **int** nAge;

       **cin** >> nAge;

       **cout** << "Enter your name: ";

       **string** strName;

       getline(**cin**, strName);

       **cout** << "Hello, " << strName << endl;

       **cout** << "Your age " << nAge << endl;

              **return** 0;

}

Kết quả khi chạy chương trình là



Khi bạn nhập một thông tin bất kỳ, mọi ký tự bạn gõ vào bàn phím (kể cả ký tự Enter **\n**) đều được đẩy vào bộ nhớ đệm trước khi được gán vào biến.

Trong chương trình trên, bạn nhập số bằng **cin**, chúng chỉ nhận số chứ không nhận được ký tự **Enter** (**‘\n’**), và ký tự **Enter** vẫn còn trong bộ nhớ đệm. Đến khi nhập chuỗi, hàm getline() nhận được ký tự **Enter** từ bộ nhớ đệm thì kết thúc nhập và chương trình vẫn chạy tiếp. Điều này khiến kết quả bị sai.

Bạn có thể xóa ký tự **Enter ‘\n’**sau khi sử dụng cin bằng cách sử dụng phương thức ***cin*.*ignore*()** thuộc **namespace std**:

// Xóa khỏi bộ nhớ đệm 32767 ký tự, hoặc đến khi gặp ký tự '\n'

**cin**.ignore(32767, '\n');

**Ví dụ:**

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **cout** << "Enter your age: ";

       **int** nAge;

       **cin** >> nAge;

       // Xóa khỏi bộ nhớ đệm 32767 ký tự, hoặc đến khi gặp ký tự '\n'

       **std**::**cin**.ignore(32767, '\n');

       **cout** << "Enter your name: ";

       **string** strName;

       getline(**cin**, strName);

       **cout** << "Hello, " << strName << endl;

       **cout** << "Your age " << nAge << endl;

       **return** 0;

}

Kết quả khi chạy chương trình là

****

**Chú ý:**Nên xóa ký tự **Enter ‘\n’** trong bộ nhớ đệm trước khi nhập chuỗi bất kỳ.

**6. Nhập một văn bản bao gồm ký tự xuống dòng**

Mặc định, hàm **getline() sử dụng ký tự ‘\n’** khi nhấn phím Enter là ký tự **báo hiệu kết thúc việc nhập chuỗi**.

Nếu muốn nhập nhiều dòng văn bản vào một biến string, bạn có thể thay đổi nó, ví dụ:

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **cout** << "Enter your text: ";

       **string** strText;

       getline(**cin**, strText, '\_');

       **cout** << "Your text: " << endl;

       **cout** << strText << endl;

       **return** 0;

}

Kết quả khi chạy chương trình là



Trong chương trình trên, bạn có thể nhập chuỗi ký tự cho đến khi chương trình nhận vào ký tự gạch dưới ‘\_’. Kết quả nhận được là một chuỗi ký tự gồm nhiều dòng.

**7. Một số thao tác cơ bản với chuỗi ký tự**

**a. Nối chuỗi (Appending strings)**

Bạn có thể sử dụng **toán tử** **+** để nối hai chuỗi với nhau, hoặc **toán tử** **+=**để nối thêm một chuỗi khác.

**Ví dụ:**

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **string** a("69");

       **string** b("96");

       **string** c = a + b;    // a and b will be appended, not added

       **cout** << c << endl;

       a += " string";

       **cout** << a << endl;

       **return** 0;

}

Kết quả khi chạy chương trình là

****

**Chú ý:**Khi nối chuỗi, lưu ý là 2 chuỗi số được nối lại với nhau, không phải cộng 2 giá trị số.

**b. Độ dài chuỗi ký tự (String length)**

Lớp string định nghĩa cho chúng ta 2 phương thức để thực hiện việc lấy ra độ dài của chuỗi kí tự.

**Ví dụ:**

#include <iostream>

#include <string>

**using** **namespace** **std**;

**int** main()

{

       **string** myName("Hello HowKteam.com!");

       **cout** << myName << " has " << myName.length() << " characters\n";

       **cout** << myName << " has " << myName.size() << " characters\n";

       **return** 0;

}

Kết quả khi chạy chương trình là

****

**PHẦN 2**

**MỘT SỐ THAO TÁC VỚI MẢNG KÝ TỰ (C-STYLE STRINGS)**

Ngôn ngữ C++ cung cấp nhiều hàm để thao tác với **mảng ký tự (C-style strings)**, những hàm này được định nghĩa bên trong thư viện **<cstring>**.

Trong bài học này, mình sẽ giới thiệu một số hàm thường được sử dụng nhất trong C++.

**1. Xem độ dài mảng ký tự (C-style strings)**

Để biết được **độ dài**mảng ký tự **(không bao gồm ký tự null ‘\0’)**, bạn có thể sử dụng hàm **strlen()**.

Ví dụ

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szTeam[20] = "Kteam"; // mảng có 20 phần tử (5 ký tự thường, 15 ký tự '\0')

 **cout** << "Team cua tui: " << szTeam << endl;

 **cout** << szTeam << " co " << **strlen**(szTeam) << " ky tu." << endl;

 **cout** << szTeam << " co " << **sizeof**(szTeam) << " phan tu trong mang." << endl;

 **return** 0;

}

**Output:**



**Lưu ý:** Hàm**strlen()** in ra số ký tự trước ký tự**‘\0’ null**, trong khi **sizeof()** trả về kích thước của **toàn bộ mảng**.

**2. Chuyển mảng ký tự (C-style strings) sang chữ hoa và chữ thường**

Để chuyển 1 chuỗi từ chữ thường sang chữ in hoa và ngược lại, bạn có thể sử dụng 2 hàm:

* **strlwr()**: chuyển chuỗi s thành **chuỗi thường**(‘A’ thành ‘a’, ‘B’ thành ‘b’, …, ‘Z’ thành ‘z’).
* **strupr()**: chuyển chuỗi s thành **chuỗi IN hoa** (‘a thành ‘A’, ‘b’ thành ‘B’, …, ‘z’ thành ‘Z’).

**Ví dụ:**

#define \_CRT\_NONSTDC\_NO\_DEPRECATE

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szString1[] = "Hello Howkteam.com!";

 **char** szString2[] = "Hello Howkteam.com!";

 **cout** << "s1: " << szString1 << endl;

 **cout** << "s2: " << szString2 << endl;

 strlwr(szString1);

 strupr(szString2);

 **cout** << "s1: " << szString1 << endl;

 **cout** << "s2: " << szString2 << endl;

 **return** 0;

}

**Output:**



Một số compiler hiện đại thường cảnh báo về việc sử dụng hàm **strlwr()** và **strupr()**, và yêu cầu lập trình viên thêm dòng lệnh **#define** **\_CRT\_NONSTDC\_NO\_DEPRECATE** vào đầu chương trình để có thể sử dụng hàm **strlwr()** và **strupr()**.

Trong **C++ 11**, bạn có thể sử dụng 2 hàm **\_strlwr\_s()** và **\_strupr\_s()** để thay thế.

**3. Sao chép mảng ký tự (C-style strings)**

Để **sao chép**1 chuỗi ký tự sang 1 chuỗi ký tự khác, bạn có thể sử dụng hàm **strcpy()**.

Ví dụ:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szSource[] = "Kteam";

 **char** szDest[20];

 // sao chép chuỗi szSource sang chuỗi szDest

 **strcpy**(szDest, szSource);

 **cout** << "Source: " << szSource << endl;

 **cout** << "Dest: " << szDest << endl;

 **return** 0;

}

**Output:**



**Chú ý:** Khi sử dụng hàm này, chuỗi đích phải**đủ lớn** **để chứa được chuỗi nguồn**. Nếu không, vấn đề **tràn mảng**sẽ xảy ra.

Một số compiler hiện đại thường cảnh báo về việc sử dụng hàm **strcpy() là không an toàn**, và yêu cầu lập trình viên thêm dòng lệnh **#define** ***\_CRT\_SECURE\_NO\_WARNINGS*** vào đầu chương trình để có thể sử dụng hàm **strcpy()**.

Trong **C++ 11**, hàm **strcpy\_s()** **được thay thế** cho hàm **strcpy()**, hàm này có thêm 1 tham số cho phép xác định độ dài của chuỗi đích. Nếu chuỗi đích không đủ lớn để chứa chuỗi nguồn, **compiler**sẽ ném ra 1 assert trong **debug mode**, và kết thúc chương trình.

Ví dụ

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szSource[] = "Howkteam.com";

 **char** szDest[5];

 // sao chép chuỗi szSource sang chuỗi szDest

 strcpy\_s(szDest, 5, szSource); // 1 assert được ném ra trong debug mode

 **cout** << "Source: " << szSource << endl;

 **cout** << "Dest: " << szDest << endl;

 **return** 0;

}

**4. Nối 2 mảng ký tự (C-style strings)**

Để **nối**1 chuỗi vào sau chuỗi khác, bạn có thể sử dụng hàm **strcat()**.

Ví dụ:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szSource[] = "Howkteam.com!";

 **char** szDest[100] = "Hello";

 // nối chuỗi

 **strcat**(szDest, " "); // "Hello "

 **strcat**(szDest, szSource); // "Hello Howkteam.com!"

 **cout** << "Dest: " << szDest << endl;

 **return** 0;

}

**Output:**



**Chú ý:** Khi sử dụng hàm **strcat(), chuỗi đích phải đủ lớn**để chứa được thêm chuỗi mới được nối nào. Nếu không, vấn đề **tràn mảng**sẽ xảy ra.

Trong **C++ 11**, hàm **strcat\_s()** **được thay thế** cho hàm **strcat()**, hàm này có thêm 1 tham số cho phép xác định độ dài của chuỗi đích. Nếu chuỗi đích không đủ lớn để chứa thêm chuỗi nguồn, **compiler**sẽ ném ra 1 assert trong **debug mode**, và kết thúc chương trình.

Ví dụ

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szSource[] = "Howkteam.com!";

 **char** szDest[10] = "Hello";

 // nối chuỗi

 strcat\_s(szDest, 10, " ");

 strcat\_s(szDest, 10, szSource); // 1 assert được ném ra trong debug mode

 **cout** << "Dest: " << szDest << endl;

 **return** 0;

}

**5. So sánh 2 mảng ký tự (C-style strings)**

Để **so sánh** hai chuỗi ký tự **s1**và **s2 (phân biệt hoa thường)**, bạn có thể sử dụng hàm **strcmp()**.

* Giá trị trả về **nhỏ hơn 0** nếu: chuỗi **s1 <** chuỗi **s2**
* Giá trị trả về **bằng 0**nếu: chuỗi **s1 ==** chuỗi **s2**
* Giá trị trả về**lớn hơn 0** nếu: chuỗi **s1 >** chuỗi **s2**

**Ví dụ:**

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szString1[] = "howkteam.com!";

 **char** szString2[] = "Howkteam.com!";

 **cout** << "s1: " << szString1 << endl;

 **cout** << "s2: " << szString2 << endl;

 // so sánh 2 chuỗi

 **int** result = **strcmp**(szString1, szString2);

 **if** (result < 0)

 **cout** << "s1 < s2" << endl;

 **else** **if** (result > 0)

 **cout** << "s1 > s2" << endl;

 **else**

 **cout** << "s1 == s2" << endl;

 **return** 0;

}

**Output:**



**Tìm kiếm chuỗi trong chuỗi**

**Để tìm vị trí xuất hiện đầu tiên của một chuỗi (s2) trong một chuỗi khác (s1), bạn có thể sử dụng hàm strstr().**

* Nếu **tìm thấy**: trả về **con trỏ đến vị trí xuất hiện đầu tiên** của chuỗi s2 trong chuỗi s1.
* Nếu**không tìm thấy**: trả về **NULL**.

Khái niệm **con trỏ** sẽ được nhắc tới trong bài CON TRỎ CƠ BẢN TRONG C++(Pointers).

**Ví dụ:**

#include <iostream>

#include <cstring>

**using** **namespace** **std**;

**int** main()

{

 **char** szString1[] = "Hello Howkteam.com!";

 **char** szString2[] = "kteam";

 **cout** << "s1: " << szString1 << endl;

 **cout** << "s2: " << szString2 << endl;

 **if** (**strstr**(szString1, szString2) != NULL)

 **cout** << "Tim thay " << szString2 << " trong " << szString1 << endl;

 **else**

 **cout** << "Khong tim thay!" << endl;

 **return** 0;

}

**Output:**

