

KỸ THUẬT LẬP TRÌNH

Chương 4: Chuỗi

Mục tiêu

► Sau khi học xong chương này, người học có thể:

1 Biết cách khai báo, lưu trữ và sử dụng chuỗi trong C++

2 Vận dụng các hàm xử lý chuỗi; thực hành một số thao tác trên chuỗi với C-string và string

Nội dung



Chuỗi (string)

1

Giới thiệu

2

Nhập, xuất chuỗi với C - string

3

Một số thao tác trên chuỗi với C - string

4

Hàm xử lý chuỗi với C - string

5

Kiểu dữ liệu string

-
- ▶ Chuỗi (string) là một dãy ký tự (gồm cả ký tự chữ, ký tự số, ký tự đặc biệt) và kết thúc bằng ký tự rỗng '\0'.
 - ▶ Ví dụ:
 - ▶ “Đại học Mo TpHCM”
 - ▶ “Khoa Công nghệ Thông tin”

- ▶ Cần phân biệt **C-string** với kiểu dữ liệu **string** có sẵn trong thư viện.

C-string

- ▶ Lưu trữ dưới dạng mảng (hoặc dùng con trỏ)
- ▶ Kết thúc bằng ký tự '\0'

string

- ▶ Lưu trữ trong class string có sẵn của C++

-
- ▶ Khai báo chuỗi (**C - string**) bằng 1 trong 2 cách:
dùng mảng (1D - array) hoặc dùng con trỏ (pointer).
 - ▶ Khai báo với class **string**: `#include <string>`

C-string

- | | |
|---------------------|-------------------------------|
| 1. Giới thiệu | 3. Một số thao tác trên chuỗi |
| 2. Nhập/ xuất chuỗi | 4. Hàm xử lý chuỗi |
| | 5. Kiểu dữ liệu string |

➤ **Khai báo chuỗi bằng mảng (1D - array)**

Cú pháp:

```
char stringName[numberOfCharacter];
```

Ví dụ:

```
char kytu[6];    //chuỗi kytu có độ dài tối đa là 5 ký tự
```

➤ **Khởi tạo giá trị bằng 1 trong 2 cách**

```
char kytu[6] = { 'C' , 'h' , 'a' , 'o' , '\\0' }
```

```
char kytu[6] = "Chao";
```

[0]	[1]	[2]	[3]	[4]	[5]
'C'	'h'	'a'	'o'	'\\0'	

► **Khai báo chuỗi bằng con trỏ (pointer):**

```
char *string_pointer;
```

Trong đó:

- char: kiểu dữ liệu.
- string_pointer: tên con trỏ.

Ví dụ:

```
char *a;
```

```
a = "hello";
```

Tương đương với

```
char a[] = "hello";
```

- ▶ **Nhập chuỗi: dùng toán tử >> và lệnh cin** (kết thúc khi gặp khoảng trắng, tab, newline)

Ví dụ:

```
char a[10];  
  
cout << "nhap chuoi toi da 10 ky tu: ";  
  
cin >> a;
```

- ▶ **Xuất chuỗi: dùng toán tử << và lệnh cout**

Ví dụ:

```
char b[6] = "hi ban";  
  
cout << b << endl; //toàn bộ chuỗi hi ban sẽ xuất ra màn hình cho  
đến khi gặp ký tự \0
```

▶ Nhập chuỗi bằng con trỏ

Ví dụ:

```
char arr[7] = "chao!";
```

```
char *a;
```

```
a = arr;
```

```
cout << a << endl;
```

```
cout << a[0] << endl;
```

```
cout << a[6] << endl;
```

C:\WINDOWS\system32\cmd.exe

chao!

c

Press any key to continue . . .

-
- ▶ **Nhập chuỗi bằng cách dùng hàm thành viên `get()`, `ignore()`**
 - ▶ **Nhập chuỗi bằng cách dùng hàm `cin.getline()`**

- | | |
|---------------------|-------------------------------|
| 1. Giới thiệu | 3. Một số thao tác trên chuỗi |
| 2. Nhập/ xuất chuỗi | 4. Hàm xử lý chuỗi |
| | 5. Kiểu dữ liệu string |

► **Nhập chuỗi bằng cách dùng hàm thành viên `get()`, `ignore()`**

```
cin.get (tên_biến_chuỗi, số_ký_tự_tối_đa_lưu_trữ) ;
```

```
char hoten[50];
```

```
cin.get(hoten, 50);
```

```
cout << hoten << endl;
```

```
cin.ignore(); //bỏ qua enter và 1 ký tự trong luồng nhập,  
hoặc cin.ignore(1)
```

```
cin.get(hoten, 50);
```

```
cout << hoten << endl;
```

```
//Nếu không có lệnh cin.ignore() thì lần nhập hoten thứ 2  
không được diễn ra.
```

▶ **Nhập chuỗi bằng cách dùng hàm `cin.getline()`**

```
cin.getline (tên_biến_chuỗi, số_ký_tự_tối_đa_lưu_trữ) ;
```

```
char hoten[50];
```

```
cin.getline(hoten, 50, '\n'); //nhập vào tối đa 49 ký tự  
kể cả khoảng trắng, ký tự '\n' sẽ được bỏ qua
```

```
cout << hoten << endl;
```

▶ **LƯU Ý:**

- ▶ Không dùng hàm `cin.get()` và `cin.getline()` với đối tượng chuỗi được khai báo là con trỏ tĩnh.
- ▶ Các hàm `cin.get()` hay `cin.getline()` đều dừng nhận ký tự khi gặp ký tự newline (hoặc enter).

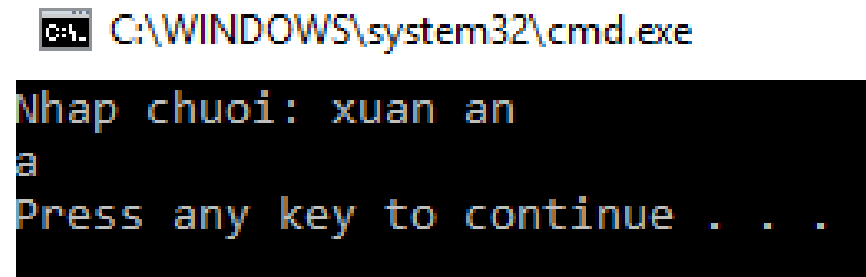
1. Giới thiệu
2. Nhập/ xuất chuỗi
3. Một số thao tác trên chuỗi
4. Hàm xử lý chuỗi
5. Kiểu dữ liệu string

-
- ▶ **Truy xuất phần tử**
 - ▶ **Xác định chiều dài**
 - ▶ **Gán chuỗi**
 - ▶ **Hàm phân loại ký tự trong header <ctype>**

▶ Truy xuất phần tử

Giống như truy xuất phần tử mảng 1 chiều

```
char a[5];  
cout << "Nhap chuoi: ";  
cin >> a;  
cout << a[2] << endl;
```



```
C:\WINDOWS\system32\cmd.exe  
Nhap chuoi: xuan an  
a  
Press any key to continue . . .
```


► Xác định chiều dài

Dùng hàm strlen() : `strlen(const char *s)` sẽ trả về kết quả số nguyên. Với s là biến chuỗi hoặc con trỏ lưu trữ chuỗi.

```
char a[10];  
cout << "Nhap chuoi: ";  
cin.get(a, 10);  
cout << strlen(a) << endl;
```

//kết quả nếu nhập a là 'chao ban' là 8

```
char a[10];  
cout << "Nhap chuoi: ";  
cin >> a;  
cout << strlen(a) << endl;
```

//kết quả nếu nhập a là 'chao ban' là 4 (vì lệnh nhập cin sẽ dừng khi gặp khoảng trắng)

1. Giới thiệu
2. Nhập/ xuất chuỗi
3. Một số thao tác trên chuỗi
4. Hàm xử lý chuỗi
5. Kiểu dữ liệu string

► Xác định chiều dài

Dùng đoạn lệnh kiểm tra:

```
char a[5];  
cout << "Nhap chuoi: ";  
cin.get(a, 5);    //lưu tối đa 4 ký tự kể cả khoảng trắng  
int demkytu = 0;  
while (a[demkytu] != '\0')  
    demkytu++;  
cout << demkytu << endl;
```

► Gán chuỗi

```
char *a = "Xuan Minh";
```

Hoặc

```
char *a;
```

```
a = "Xuan Minh";
```

Hoặc

```
char a[] = "Xuan Minh";
```

Nhưng không được

```
char a[9];
```

```
a = "Xuan Minh"; //error
```

▶ Hàm phân loại các ký tự có trong header <cctype>

- `int isalnum(int ch)`: trả về số khác 0 nếu `ch` là ký tự chữ hoặc số.
- `int isalpha(int ch)`: trả về số khác 0 nếu `ch` là chữ.
- `int isdigit(int ch)`: trả về số khác 0 nếu `ch` là số.
- `int islower(int ch)`: trả về số khác 0 nếu `ch` là ký tự thường.
- `int ispunct(int ch)`: trả về số khác 0 nếu `ch` là ký tự dấu câu.
- `int isspace(int ch)`: trả về số khác 0 nếu `ch` là ký tự khoảng trắng (spaces, tabs, newlines).
- `int isupper(int ch)`: trả về số khác 0 nếu `ch` là ký tự hoa.

▶ **Hàm phân loại các ký tự có trong header <cctype>**

Ví dụ: từ chuỗi ký tự s, hãy đếm xem chuỗi có bao nhiêu ký tự khoảng trắng?

//Nhập chuỗi s

```
char *p = s;
```

```
int dem = 0;
```

```
while (*p)
```

```
{
```

```
    if (isspace (*p))    dem++;
```

```
    p++;
```

```
}
```

```
cout << "So khoang trang la: " << dem << endl;
```

Bài tập

1. Viết chương trình nhập vào 1 chuỗi tối đa 50 ký tự. Sau đó đếm xem có bao nhiêu ký tự là ký tự chữ hoặc số? Bao nhiêu ký tự khoảng trắng?
2. Viết chương trình nhập vào 1 chuỗi tối đa 50 ký tự. Sau đó chuyển toàn bộ các ký tự đầu mỗi chữ đều in hoa, các ký tự không phải ký tự đầu sẽ chuyển sang in thường. Xuất lại chuỗi để kiểm chứng.
3. Viết chương trình nhập vào 1 chuỗi tối đa 50 ký tự. Sau đó xuất từng từ của chuỗi vừa nhập dưới dạng trên từng dòng.

- **Hàm strcat()**
- **Hàm strncat()**
- **Hàm strchr()**
- **Hàm strcmp()**
- **Hàm strncmp()**
- **Hàm strcpy()**
- **Hàm strncpy()**
- **Hàm strlen()**
- **Hàm strtok()**
- **Hàm atof()**
- **Hàm atoi()**
- **Hàm atol()**

Hàm strcat()

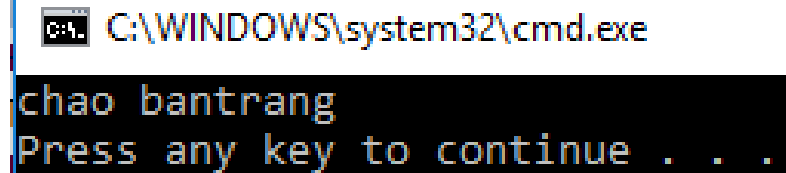
```
char *strcat(char *s1, const char *s2);
```

Công dụng: nối chuỗi s2 vào cuối chuỗi s1. Trả về chuỗi s1 sau khi nối.

Lưu ý: phải đảm bảo chuỗi s1 đủ để chứa chuỗi s2 sau khi nối.

Ví dụ:

```
char s1[20] = "chao ban";  
char s2[] = "trang";  
strcat(s1, s2);  
cout << s1 << endl;
```



C:\WINDOWS\system32\cmd.exe
chao bantrang
Press any key to continue . . .

Hàm strncat()

```
char *strncat(char *s1, const char *s2, size_t_n);
```

size_t_n: số ký tự của chuỗi s2 (tính từ trái qua) muốn nối vào s1.

Công dụng: nối n ký tự của chuỗi s2 vào cuối chuỗi s1. Trả về chuỗi s1 sau khi nối.

Lưu ý: phải đảm bảo chuỗi s1 đủ để chứa chuỗi s2 sau khi nối.

Ví dụ:

```
char s1[20] = "chao ban";  
char s2[] = "trang";  
strncat(s1, s2, 2);  
cout << s1 << endl;
```



```
C:\WINDOWS\system32\cmd.exe
```

```
chao bantr  
Press any key to continue . . .
```

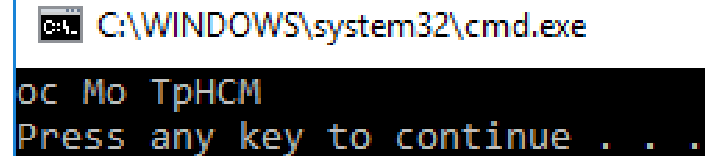
Hàm strchr():

```
char *strchr(const char *s, int c);
```

Công dụng: định vị lần xuất hiện đầu tiên của ký tự c trong chuỗi s. Nếu c được tìm thấy trong chuỗi s thì con trỏ trỏ đến c trong s sẽ được trả về. Ngược lại con trỏ NULL được trả về.

Ví dụ:

```
char *s = "Dai hoc Mo TpHCM";  
char *p;  
p = strchr(s, 'o');  
cout << p << endl;
```



C:\WINDOWS\system32\cmd.exe
oc Mo TpHCM
Press any key to continue . . .

Hàm strcmp()

```
int *strcmp(const char *s1, const char *s2);
```

Công dụng: so sánh chuỗi 1 với chuỗi 2 (so sánh theo ASCII).

Nếu $s1 == s2$ thì trả về 0

Nếu $s1 > s2$ thì trả về > 0

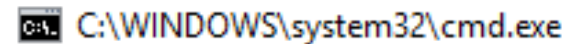
Nếu $s1 < s2$ thì trả về < 0

Hàm strcmp()

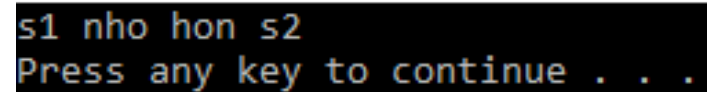
```
int *strcmp(const char *s1, const char *s2);
```

Ví dụ:

```
char *s1 = "ABC";
char *s2 = "ABCD";
if ( strcmp(s1,s2) == 0 )
    cout << "2 chuỗi bằng nhau\n";
else
    if (strcmp(s1,s2) < 0 )
        cout << "s1 nhỏ hơn s2\n";
    else
        cout << "s1 lớn hơn s2\n";
```



C:\WINDOWS\system32\cmd.exe



s1 nhỏ hơn s2
Press any key to continue . . .

Hàm strncmp()

```
int *strncmp(const char *s1, const char *s2, size_t n);
```

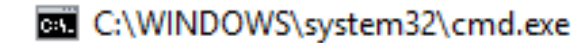
Công dụng: tương tự như strcmp() nhưng so sánh đến n ký tự.

Hàm strcmp()

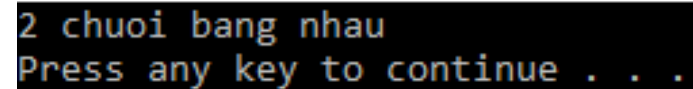
```
int *strcmp(const char *s1, const char *s2, size_t_n);
```

Ví dụ:

```
char *s1 = "ABC";  
char *s2 = "ABCD";  
if ( strcmp(s1,s2,3) == 0 )  
    cout << "2 chuỗi bằng nhau\n";  
else  
    if (strcmp(s1,s2) < 0 )  
        cout << "s1 nhỏ hơn s2\n";  
    else  
        cout << "s1 lớn hơn s2\n";
```



C:\WINDOWS\system32\cmd.exe



2 chuỗi bằng nhau
Press any key to continue . . .

Hàm strcpy()

```
int *strcpy(char *s1, const char *s2);
```

Công dụng: sao chép chuỗi s2 vào mảng ký tự s1. Trả về giá trị s1

Lưu ý: mảng s1 phải đủ kích thước chứa s2

Ví dụ:

```
char s1[30] = "Truong";  
char *s2 = "Dai hoc Mo TpHCM";  
cout << "Chuoi s1 truoac khi sao chep: " << s1 << endl;  
strcpy(s1, s2);  
cout << "Chuoi s1 sau khi sao chep: " << s1 << endl;
```

C:\WINDOWS\system32\cmd.exe

```
Chuoi s1 truoac khi sao chep: Truong  
Chuoi s1 sau khi sao chep: Dai hoc Mo TpHCM  
Press any key to continue . . .
```

Hàm strncpy()

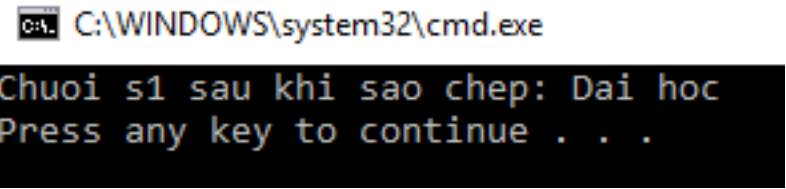
```
int *strncpy(char *s1, const char *s2, size_t_n);
```

Công dụng: Tương tự strcpy() nhưng sao chép tối đa n ký tự.

Lưu ý: mảng s1 phải đủ kích thước chứa s2

Ví dụ:

```
char s1[30] = "Truong";  
char *s2 = "Dai hoc Mo TpHCM";  
strncpy(s1, s2, 7);  
cout << "Chuoi s1 sau khi sao chép: " << s1 << endl;
```



C:\WINDOWS\system32\cmd.exe

Chuoi s1 sau khi sao chép: Dai hoc
Press any key to continue . . .

Hàm strlen()

```
strlen(const char *s)
```

Công dụng: sẽ trả về kết quả số nguyên. Với s là biến chuỗi hoặc con trỏ lưu trữ chuỗi.

Hàm strtok()

```
int *strtok(char *s1, const char *s2);
```

Công dụng: ngắt s1 thành các token (từ tố) bởi ký tự của s2.

Ví dụ:

```
char s1[] = "Dai hoc Mo TpHCM";
```

```
char s2[] = " ";
```

```
char *ptok = strtok(s1, s2);
```

```
while (ptok != NULL)
```

```
{
```

```
    cout << ptok << endl;
```

```
    ptok = strtok(NULL, s2);
```

```
}
```

```
Dai
hoc
Mo
TpHCM
Press any key to continue . . .
```

Hàm atof()

```
double atof (const char *s) ;
```

Công dụng: chuyển chuỗi s thành giá trị double.

Lưu ý: nếu không thể chuyển thì kết quả là 0

Ví dụ:

```
char *s = "209.23";  
double kq = atof(s);  
cout << kq << endl;
```

Hàm atoi()

```
int atoi (const char *s) ;
```

Công dụng: chuyển chuỗi s thành giá trị int.

Lưu ý: nếu không thể chuyển thì kết quả là 0

Hàm atol()

```
long atol (const char *s) ;
```

Công dụng: chuyển chuỗi s thành giá trị long int.

Lưu ý: nếu không thể chuyển thì kết quả là 0

Bài tập

4. Viết chương trình nhập vào 1 chuỗi họ và tên. Sau đó tiến hành tách họ, chữ lót, tên ra thành 3 dòng khác nhau.
5. Viết chương trình nhập vào số điện thoại, sau đó tách mã vùng, số tổng đài, số nội bộ. Biết rằng:
 - Mã vùng: chứa số 0 đầu tiên, có thể dài 2 đến 4 ký tự.
 - Số tổng đài: nếu ở thành phố Hà Nội hay TPHCM thì số tổng đài bao gồm 8 ký tự. Nếu ở tỉnh có 7 chữ số.
 - Số nội bộ: 3 ký tự cuối.
6. Viết chương trình nhập vào một chuỗi dạng ngày-tháng-năm? Sau đó tính xem đó là ngày thứ bao nhiêu trong năm (Tính từ ngày 01 tháng 01).

Hạn chế của C – string

- Không thể dùng phép so sánh ==
- Không thể nối chuỗi bằng +, mà phải dùng hàm thư viện
- Không thể gán 1 chuỗi cho 1 chuỗi khác (bằng dấu =)
- Không tự kiểm tra số lượng phần tử vượt quá phạm vi lưu trữ mảng

1. Giới thiệu
2. Nhập/ xuất chuỗi
3. Một số thao tác trên chuỗi
4. Hàm xử lý chuỗi
5. Kiểu dữ liệu string

string

Khai báo, khởi tạo

Nhập/xuất

Nhập với hàm getline()

Các phép toán

Các hàm cơ bản

Khai báo, khởi tạo

- Là class có sẵn, sử dụng: `#include <string>`
- Ví dụ:

```
string s;
```

```
s = "Hello world!";
```

```
cout << s << endl;
```

- | | |
|---------------------|-------------------------------|
| 1. Giới thiệu | 3. Một số thao tác trên chuỗi |
| 2. Nhập/ xuất chuỗi | 4. Hàm xử lý chuỗi |
| | 5. Kiểu dữ liệu string |

Nhập/xuất

- **Nhập**: dùng toán tử `>>` và `cin`
- **Xuất**: dùng toán tử `<<` và `cout`

Lưu ý: khi nhập bằng `>>` và `cin` thì chỉ đọc đến khi gặp khoảng trắng sẽ dừng.

Nhập với hàm getline()

Nhập kể cả khoảng trắng

```
getline(cin, s [, delimiter]);
```

- s: tên chuỗi
- delimiter: ký tự kết thúc nhập, mặc định là newline.

Ví dụ:

```
string address;  
cout << "\nNhap chuoi dia chi: ";  
getline(cin, address, '#');  
  
cout << address << endl;
```

```
Nhap chuoi dia chi: 36 nguyen cong tru#  
36 nguyen cong tru  
Press any key to continue . . .
```

1. Giới thiệu
2. Nhập/ xuất chuỗi
3. Một số thao tác trên chuỗi
4. Hàm xử lý chuỗi
5. Kiểu dữ liệu string

Các phép toán

`s1 = s2`

gán s2 cho s1

`s += x`

thêm x vào cuối chuỗi s, x là ký tự, string hoặc c-string

`s[i]`

ký tự thứ i của chuỗi

`s = s1 + s2`

chép s1 vào s, sau đó nối s2 vào s

`s1 == s2`

so sánh s1 bằng s2

`s1 != s2`

`!(s1 == s2)`

`s1 < s2`

s1 nhỏ hơn s2

`s1 <= s2`

s1 nhỏ hơn hoặc bằng s2

`s1 > s2`

s1 lớn hơn s2

`s1 >= s2`

s1 nhỏ hơn hoặc bằng s2

1. Giới thiệu
2. Nhập/ xuất chuỗi
3. Một số thao tác trên chuỗi
4. Hàm xử lý chuỗi
5. Kiểu dữ liệu string

Các hàm cơ bản

<code>s.size()</code>	số ký tự có trong chuỗi <code>s</code>
<code>s.length()</code>	số ký tự có trong chuỗi <code>s</code>
<code>s.c_str()</code>	chuyển <code>s</code> sang chuỗi c-string
<code>s.insert(pos,x)</code>	chèn <code>x</code> trước vị trí <code>pos</code> trong chuỗi <code>s</code> , <code>x</code> là ký tự, <code>string</code> hoặc c-string
<code>s.append(pos,x)</code>	chèn <code>x</code> sau vị trí <code>pos</code> trong chuỗi <code>s</code> , <code>x</code> là ký tự, <code>string</code> hoặc c-string
<code>s.erase(pos)</code>	loại bỏ ký tự tại vị trí <code>pos</code> , kích thước chuỗi <code>s</code> bị giảm 1.
<code>pos = s.find(x)</code>	tìm <code>x</code> trong chuỗi <code>s</code> , <code>x</code> là ký tự, <code>string</code> hoặc c-string, <code>pos</code> là chỉ số của ký tự đầu tiên được tìm thấy

Một số thao tác và hàm làm việc với string

- Truy xuất chỉ số từng phần tử trong chuỗi string: dùng cặp dấu [] (bắt đầu từ 0)
- **s.length()** : là hàm thuộc tính trả về 1 số nguyên là số ký tự trong chuỗi s
- **toupper(char kt)** : chuyển ký tự kt thành in hoa. Trả về 1 ký tự sau khi in hoa. Không thể tự gọi đơn thuần mà phải gán kết quả cho 1 biến cùng kiểu.
- **tolower(char kt)** : chuyển ký tự kt thành in thường. Trả về 1 ký tự sau khi in thường. Không thể tự gọi đơn thuần mà phải gán kết quả cho 1 biến cùng kiểu.

Một số thao tác và hàm làm việc với string

- `s.substr(begin_position [, size_t_n])`: lấy chuỗi con bắt đầu từ vị trí `begin_position` và số ký tự lấy là `size_t_n`. Trả về 1 chuỗi con (dạng string).
- Không thể tự gọi đơn thuần mà phải gán kết quả cho 1 biến cùng kiểu.
- Ví dụ:

```
string s = "La con gai that tuyet";  
string str = s.substr(3,3);  
cout << str << endl;
```

//kết quả: con

Một số thao tác và hàm làm việc với string

- `s.find(search_string, begin_position_search)`: tìm 1 chuỗi con `search_string` xem có ở trong `s` hay không (kể từ vị trí bắt đầu là `begin_position_search` về cuối chuỗi). Nếu không tìm thấy sẽ trả về -1, nếu có trả về vị trí đầu tiên xuất hiện.
- Không thể tự gọi đơn thuần mà phải gán kết quả cho 1 biến số nguyên.
- Ví dụ:

```
string s = "La con gái thật tuyệt";  
string search_s = "thật kinh";  
int kq = s.find(search_s, 0);  
cout << kq << endl;  
//kết quả -1
```


Một số thao tác và hàm làm việc với string

- `s.erase(begin_position [, size_t_n])`: xóa đi `size_t_n` ký tự kể từ vị trí bắt đầu xóa là `begin_position`. Nếu không ấn định `size_t_n` thì sẽ xóa kể từ vị trí bắt đầu về cuối chuỗi. Trả về chuỗi sau khi xóa.
- Chuỗi có thể xóa trực tiếp trên bản thân và gán lại cho chính nó.
- Ví dụ:

```
string s = "La con gái that tuyet";  
s.erase(0, 3);  
cout << s << endl;  
//con gái that tuyet
```

Một số thao tác và hàm làm việc với string

- **s.replace (begin_position, size_t_n, replace_string) :** thay thế chuỗi thay thế replace_string với số ký tự cần thay thế là size_t_n kể từ vị trí bắt đầu begin_position. Kết quả trả về 1 chuỗi sau khi thay thế.
- Chuỗi có thể thay thế trực tiếp trên bản thân và gán lại cho chính nó.
- Ví dụ:

```
string s = "La con gai that tuyet";  
string s1 = "kinh lam";  
s.replace(16, 4, s1);  
cout << s << endl;
```

```
//La con gai that kinh lamt
```

Một số thao tác và hàm làm việc với string

- **`s.insert(begin_position, insert_string)`**: Chèn chuỗi `insert_string` vào chuỗi `s` kể từ vị trí bắt đầu `begin_position`. Kết quả trả về chuỗi sau khi chèn.
- Chuỗi có thể chèn trực tiếp trên bản thân và gán lại cho chính nó.
- Ví dụ:

```
string s = "La con gái thật tuyệt";  
s.insert(21, " lam lam!");  
cout << s << endl;
```

```
//La con gái thật tuyệt lam lam!
```

Một số thao tác và hàm làm việc với string

- **s.assign (count, assign_char)**: Gán ký tự assign_char vào chuỗi s với count số lần. Kết quả trả về chuỗi sau khi gán.
- Chuỗi có thể chèn trực tiếp trên bản thân và gán lại cho chính nó.
- Ví dụ:

```
string s = "la con gai that tuyet";  
s.assign(5, '--');  
cout << s << endl;
```

```
//-----
```

Bài tập với string

- ▶ Viết chương trình dùng string để cho phép nhập vào một chuỗi ký tự và 1 từ. Sau đó tìm số lần xuất hiện của từ trong chuỗi đó.

Q & A