Ngôn ngữ lập trình C

B13: Xâu ký tự (String)

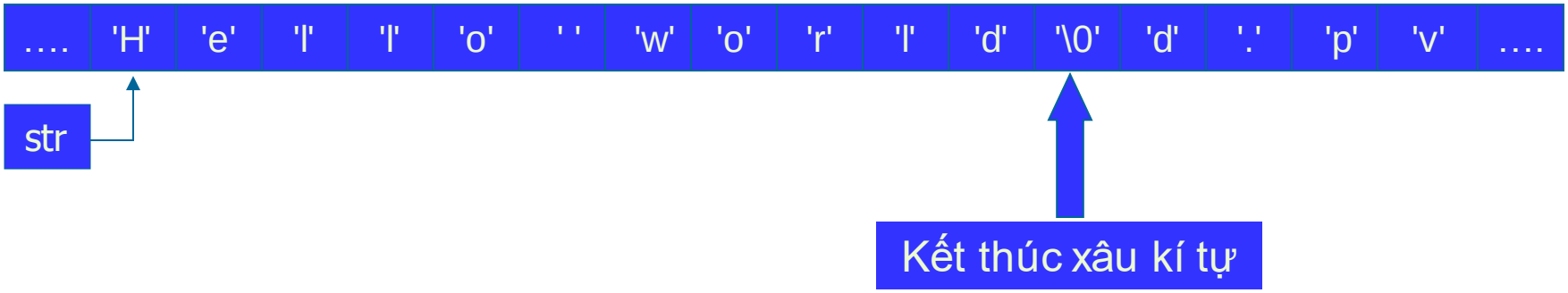
Khoa Công nghệ thông tin

Khái niệm xâu kí tự

- Là một mảng các kí tự (một dãy các kí tự viết liên tiếp nhau).
- Dùng để lưu trữ text.
- Xâu rỗng: Xâu không gồm kí tự nào cả.
- Độ dài xâu: Số kí tự có trong xâu.
- Ví dụ:
  - "Tin hoc" là một xâu kí tự gồm 7 kí tự: 'T', 'i', 'n', dấu cách (' '), 'h', 'o', và 'c'.

Kết thúc xâu kí tự

- Lưu trữ xâu kí tự: Ký tự kết thúc xâu là kí tự: `NULL` hoặc `'\0'` (mã `ascii` là 0).



- Đây là quy ước để biết đâu là nơi kết thúc một xâu kí tự.
- Nghĩa là để lưu trữ một xâu có N kí tự thì chúng ta cần một mảng có độ dài N + 1 để lưu trữ thêm kí tự kết thúc xâu.

Khái niệm xâu kí tự

- Cần phân biệt giữa kí tự và xâu bao gồm một kí tự.
- Ví dụ: 'A’ là ký tự A được mã hóa bằng 1 byte, trong khi "A“ là một xâu kí tự chứa kí tự A, xâu này được mã hóa bằng 2 bytes cho kí tự A và kí tự '\0’.
- Việc truy cập phần tử của xâu hoàn toàn tương tự cách truy cập một phần tử mảng thông thường.

Khai báo xâu kí tự

- Cú pháp:

```
char ten_xau[so_ky_tu_toi_da];
```

- Ví dụ:

```
char ho_va_ten[20];
```

- Trong C không tồn tại các phép toán so sánh, gán nội dung của xâu này cho xâu khác. Để thực hiện các công việc này C cung cấp cho người lập trình một thư viện các hàm chuẩn, được khai báo trong tệp header có tên là **string.h.**
- Để sử dụng các hàm thao tác xâu, trên đầu chương trình cần phải có dòng khai báo:

```
#include<string.h>
```

Thư viện xâu kí tự

- Thư viện `#include<string.h>`:
  - int strlen(char* tên_xâu);
    - Trả về độ dài (số kí tự có trong xâu) của xâu kí tự tên_xâu.
  - char* strcpy(char* xâu_đích, char* xâu_nguồn)
    - Sao chép nội dung xâu_nguồn và ghi lên xâu_đích.
  - char* strncpy(char* xâu_đích, char* xâu_nguồn, int n)
    - Tương tự strcpy() nhưng ngừng sao chép sau n ký tự. Trong trường hợp không có đủ số ký tự trong xâu nguồn thì hàm sẽ điền thêm các ký tự trắng vào xâu đích.

Thư viện xâu kí tự

- Thư viện `#include<string.h>`:
  - **int strcmp(char* xâu_thứ_nhất, char* xâu_thứ_hai);**
    - Trả về
      - giá trị < 0 nếu xâu_thứ_nhất nhỏ hơn xâu_thứ_hai
      - giá trị 0 nếu xâu_thứ_nhất bằng xâu_thứ_hai
      - giá trị > 0 nếu xâu_-thứ _nhất lớn hơn xâu_thứ_hai
  - **int strncmp(char* xâu_thứ_nhất, char* xâu_thứ_hai, int n);**
    - Tương tự như hàm strcmp nhưng giới hạn việc so sánh với n ký tự đầu tiên của hai xâu.
  - **int stricmp(char* xâu_thứ_nhất, char* xâu_thứ_hai);**
    - Tương tự như strcmp() nhưng không phân biệt chữ hoa hay chữ thường.
  - **int strnicmp(char* xâu_thứ_nhất, char* xâu_thứ_hai, int n);**
    - Tương tự như hàm stricmp nhưng giới hạn việc so sánh với n ký tự đầu tiên của hai xâu.

Thư viện xâu kí tự

- Thư viện `#include<string.h>`:
  - **char* strchr(char* str, int ch);**
    - Tìm kiếm vị trí của kí tự ch trong xâu str.
    - Nếu có kí tự ch trong str thì hàm strchr() trả về con trỏ trỏ tới kí tự ch đầu tiên trong str, ngược lại nó sẽ trả về con trỏ NULL.
  - **char* strrchr(char* str, int ch);**
    - Tương tự như strchr() nhưng việc tìm kiếm bắt đầu từ cuối xâu str.
  - **char* strstr(char* str1, char* str2);**
    - Tìm kiếm vị trí của xâu con str2 trong xâu str1.
    - Nếu str2 là xâu con của str1 thì hàm strstr() trả về con trỏ trỏ tới kí tự đầu tiên của xâu con str2 đầu tiên trong str1, ngược lại nó sẽ trả về con trỏ NULL.

Thư viện xâu kí tự

- Thư viện `#include<string.h>`:
  - char* strcat(char* xâu_đích, char* xâu_nguồn);
    - Ghép nối xâu_nguồn vào ngay sau xâu_đích.
    - Kết quả trả về của hàm strcat() là xâu mới ghép nối từ 2 xâu xâu_nguồn và xâu_đích.
  - char* strncat(char* xâu_đích, char* xâu_nguồn, int n);
    - Tương tự strcat nhưng chỉ giới hạn với n ký tự đầu tiên của xâu_nguồn.
  - char* strlwr(char* xâu)
    - Chuyển đổi các chữ in hoa trong xâu sang chữ thường.
  - char* strupr(char* xâu)
    - Chuyển đổi các chữ thường trong xâu sang chữ in hoa.

Thư viện xâu kí tự

- Thư viện `#include<string.h>/<stdlib.h>`:
  - void strset(char* s, char c)
    - Khởi đầu giá trị tất cả các ký tự của xâu s bằng ký tự c
  - void strnset(char* s, char c, int n)
    - Khởi đầu giá trị cho n ký tự đầu tiên của xâu s bằng ký tự c
  - int atoi(char* str)
    - Chuyển một xâu kí tự là biểu diễn của một số nguyên thành số nguyên tương ứng.
    - Nếu chuyển đổi thành công, hàm atoi() trả về giá trị số nguyên chuyển đổi được, ngược lại trả về giá trị 0.

Thư viện xâu kí tự

- Thư viện `#include<ctype.h>`: các hàm xử lý kí tự.
  - **`int toupper(int ch)`**
    - Chuyển một kí tự chữ cái thường (các kí tự 'a', 'b', ..., 'z') thành kí tự chữ cái hoa tương ứng ('A', 'B', ..., 'Z').
  - **`int tolower(int ch)`**
    - Chuyển một kí tự chữ cái hoa ('A', 'B', ..., 'Z') thành kí tự chữ cái thường tương ứng ('a', 'b', ...'z').
  - **`int isalpha(int ch)`**
    - Kiểm tra một kí tự có phải là chữ cái hay không ('a', 'b', ..., 'z', 'A', 'B', ..., 'Z').
    - Hàm trả về giá trị khác không nếu đúng là chữ cái, trả về giá trị 0 nếu ngược lại.

Thư viện xâu kí tự

- Thư viện `#include<ctype.h>`: các hàm xử lý kí tự.
  - **`int isdigit(int ch)`**
    - Kiểm tra một kí tự có phải là chữ số hay không ('0', '1', …'9').
    - Hàm trả về giá trị khác không nếu đúng, trả về giá trị 0 nếu ngược lại.
  - **`int islower(int ch)`**
    - Kiểm tra một kí tự có phải là chữ cái thường hay không ('a', 'b', …'z').
    - Hàm trả về giá trị khác không nếu đúng, trả về giá trị 0 nếu ngược lại.
  - **`int isupper(int ch)`**
    - Kiểm tra một kí tự có phải là chữ cái hoa hay không ('A', 'B', …'Z').
    - Hàm trả về giá trị khác không nếu đúng, trả về giá trị 0 nếu ngược lại.

Thư viện xâu kí tự

- Thư viện `#include<ctype.h>`: các hàm xử lý kí tự.
  - **`int iscntrl(int ch)`**
    - Kiểm tra một kí tự có phải là kí tự điều khiển hay không (là các kí tự không hiển thị được và có mã ASCII từ 0 đến 31).
    - Hàm trả về giá trị khác không nếu đúng, trả về giá trị 0 nếu ngược lại.
  - **`int isspace(int ch)`**
    - Kiểm tra một kí tự có phải là dấu cách (space, mã ASCII là 32), kí tự xuống dòng ('\n', mã ASCII là 10), kí tự về đầu dòng ('\r', mã ASCII là 13), dấu tab ngang ('\t', mã ASCII là 9) hay dấu tab dọc ('\v', mã ASCII là 11) hay không.
    - Hàm trả về giá trị khác không nếu đúng, trả về giá trị 0 nếu ngược lại.

Ví dụ 13.1

```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[ 20 ] = "Happy ";
    char s2[] = "New Year ";
    char s3[ 40 ] = "";
    printf( "s1 = %s\ns2 = %s\n", s1, s2 );
    printf( "strcat( s1, s2 ) = %s\n", strcat( s1, s2 ) );
    printf( "strncat( s3, s1, 6 ) = %s\n", strncat( s3, s1, 6 ) );
    printf( "strcat( s3, s1 ) = %s\n", strcat( s3, s1 ) );
    return 0;
}
```

```
s1 = Happy
s2 = New Year
strcat( s1, s2 ) = Happy New Year
strncat( s3, s1, 6 ) = Happy
strcat( s3, s1 ) = Happy Happy New Year
```

Con trỏ và xâu ký tự

- Như đã biết xâu kí tự là một dãy các kí tự đặt trong hai dấu nháy kép, ví dụ: `"Dai Hoc PHENIKAA"`.
- Cũng giống như mảng, xâu kí tự là một hằng địa chỉ biểu thị địa chỉ đầu của mảng kí tự chứa xâu (Mảng này chứa các kí tự của xâu cộng thêm kí tự `NULL`).
- Vì vậy, nếu chúng ta khai báo `char*pc;` là một con trỏ kiểu `char` thì phép gán `pc="Dai Hoc PHENIKAA"` là hợp lệ. Sau khi thực hiện câu lệnh này, trong con trỏ pc có địa chỉ đầu của xâu `"Dai Hoc PHENIKAA"`.

Con trỏ và xâu ký tự

- Giữa con trỏ kiểu `char` và mảng kiểu `char` có sự khác biệt. Xét ví dụ sau:

```
char ch[20];
char *pc;
pc = "Dai Hoc Bach Khoa";/*Hợp lệ*/
ch = "Dai Hoc Bach Khoa";/*Không hợp lệ*/
```

- câu lệnh gán đầu tiên là hợp lệ vì `pc` là con trỏ kiểu `char`, còn câu lệnh gán thứ hai không hợp lệ vì `ch` được khai báo là một mảng ký tự, do đó ch là một hằng, chúng ta không thể gán giá trị cho hằng được.

→ Có thể thao tác trên tất cả các kí tự của xâu thông qua một con trỏ.

Bài tập

1. Viết một chương trình nhập vào một dòng kí tự, viết một hàm để đếm số kí tự dấu cách của xâu và hiển thị ra màn hình số kí tự trắng của xâu.
2. Nhập một xâu kí tự từ bàn phím gồm các từ, ví dụ "Thu do Ha Noi". Lập chương trình để bỏ bớt các dấu trống giữa các từ sao cho các từ chỉ cách nhau ít nhất một dấu trống.
3. Viết chương trình nhập vào từ bàn phím họ và tên của một người, sau đó in phần tên ra màn hình. Ví dụ:"Le Hoang Anh" thì in ra "Anh".
4. Nhập vào một câu, kết thúc bằng dấu chấm. In ra câu đó có bao nhiêu từ.

Bài tập 13.5

- Viết một hàm:
  - Đầu vào là một xâu và 2 kí tự
  - Hàm làm nhiệm vụ tìm kiếm trong xâu đưa vào các kí tự thứ nhất và thay thế bằng kí tự thứ 2..
- Viết chương trình để kiểm tra hàm trên, cho phép người dùng nhập vào một xâu và 2 kí tự. In ra màn hình xâu đã nhập ban đầu và xâu sau khi thay thế.
- Ví dụ:
  - input: “papa”, ‘p’, ‘m’
  - output: “mama”

Bài tập 13.6 – Mã hóa

- Phương pháp mã hóa cổ điển các văn bản được thực hiện như sau:
  - Dịch ký tự sang k bước trong bảng chữ cái, và xoay vòng tròn. Ví dụ: a -> c, b->d, z ->b. Việc giải mã được thực hiện ngược lại.
  - Viết hàm mã hóa và giải mã một xâu ký tự với giá trị bước dịch chuyển tùy biến ở dạng tham số.
  - Sử dụng hàm trên để mã hóa một đoạn văn bản nhập từ bàn phím sau đó giải mã.

Bài tập 13.7 – Mã hóa

- Viết hàm mã hóa bằng việc dịch tất cả các kí tự của một từ sang một vị trí:
  - Dịch trái: « hello » → « elloh »
  - Dịch phải: « hello » → « ohell »

Bài tập 13.8

- Cho một đoạn văn bản, thống kê tần suất xuất hiện của một kí tự là chữ cái và chữ số mà người dùng nhập từ bàn phím.