**BUỔI 19: CHUỖI TRONG C**

**FULL HOUSE**

**I. Chuỗi trong C** 

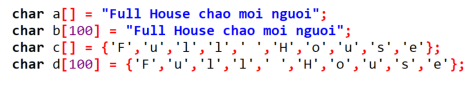
**1. Định nghĩa:**

Trong ngôn ngữ lập trình C, chuỗi là một tập hợp các ký tự (**char**) được lưu trữ trên các ô nhớ liên tiếp và luôn luôn có 1 ký tự **null** là **\0** báo hiệu kết thúc chuỗi.

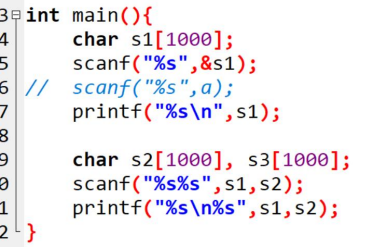


Như vậy, nếu bạn muốn khai báo chuỗi để lưu **n** ký tự, bạn cần mảng ký tự có kích thước tối đa ít nhất là **n+1**.

**2. Khởi tạo**

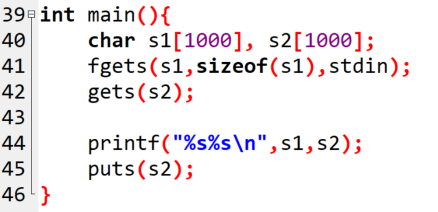
**3. Cách nhập xuất**

Chúng ta có thể sử dụng hàm **scanf** để nhập và hàm **printf** để xuất. Ở hàm **scanf** ta có thể bỏ đi dấu **&**. Chúng ta có thể nhập nhiều chuỗi một lúc trong hàm **scanf**.



**4. Cách nhập chuỗi có khoảng trắng**

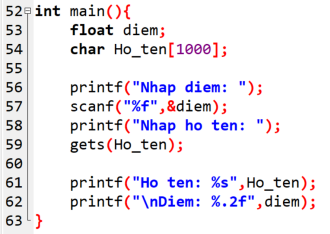
**a. Nhập bình thường**

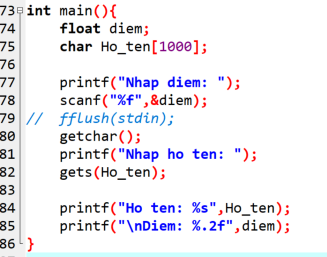
****

*# Hàm* ***gets*** *và hàm* ***fgets*** *đều là 2 hàm dùng để nhập chuỗi có chứa khoảng trắng nhưng hàm* ***fgets*** *sẽ đọc luôn kí tự* ***\n*** *còn hàm* ***gets*** *thì không.*

*# Hàm* ***printf*** *và hàm puts đều là 2 hàm dùng để in ra chuỗi nhưng hàm* ***puts*** *sẽ in ra thêm 1 dấu* ***\n*** *ở cuối và hàm này chỉ in duy nhất được 1 chuỗi còn hàm* ***printf*** *thì ngược lại.*

**b. Lỗi không cho nhập chuỗi (getchar() và fflush(stdin);** • Chương trình lỗi

• Chương trình được fix 



Khi chúng ta nhập một biến trước khi sử dụng hàm **gets** hay **fgets** thì bộ nhớ sẽ lưu lại kí tự **\n**. Hàm **fflush(stdin)** và hàm **getchar()** đều dùng để xóa bộ nhớ đệm nhưng ta nên dùng hàm **getchar().**

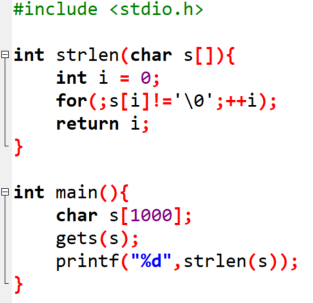
**II. Một số hàm hỗ trợ**

**1. Hàm strlen().**

• **Thư viện**: string.h

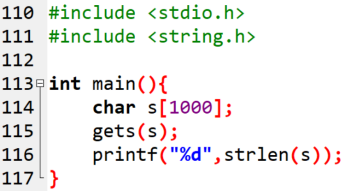
• **Mục đích**: Hàm **strlen()** trả về chiều dài của chuỗi, nó không đếm ký tự null ‘\0’.

• **Xây dựng hàm**: 



• **Cú pháp**: **int strlen**(const char**\*** s)

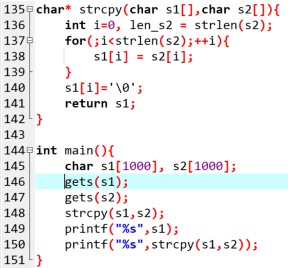
• **Ví dụ**:



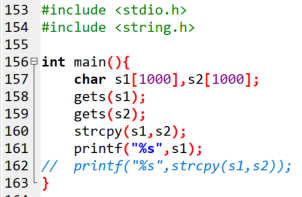
**2. Hàm strcpy();**

• **Thư viện**: string.h

• **Mục đích**: Sao chép nội dung của chuỗi source tới chuỗi destination. Đây là một hàm rất quan trọng khi xử lý chuỗi. Hàm strcpy chỉ sao chép dữ liệu từ mảng B[ ] sang mảng A[ ], nếu mảng A[ ] có kích thước nhỏ hơn mảng B[ ]thì sẽ xảy ra việc tràn dữ liệu của mảng A[ ]. • **Xây dựng hàm**: 



• **Cú pháp**: **char \*strcpy**(char **\***destination, const char **\***source) • **Ví dụ**:



**3. Hàm strcat();**

• **Thư viện**: string.h

• **Mục đích**: Dùng để nối 2 chuỗi. Kết quả được lưu vào chuỗi đầu tiên. • **Xây dựng hàm:**

****• **Cú pháp**: **char** \***strcat**(char **\***des, const char **\***source) • **Ví dụ:**

****

**4. Hàm strcmp()**

• **Thư viện**: string.h

• **Mục đích**: Dùng để so sánh hai chuỗi với nhau. Hàm sẽ so sánh hai chuỗi với nhau và trả về các giá trị tương ứng.

Ta có 2 mảng s1[] và s2[]

Nếu s1[] > s2[], kết quả trả về là 1 

Nếu s1[] == s2[], kết quả trả về là 0

Nếu s1[] < s2[], kết quả trả về là -1

• **Xây dựng hàm:**

****

• **Cú pháp**: **int strcmp**(const cha**r\***s1, const char **\***s2)

• **Ví dụ:**

****

**5. Hàm strchr()**

• **Thư viện**: string.h

• **Mục đích**: Dùng để tìm kiếm sự xuất hiện đầu tiên của ki tự c trong chuỗi s1. Kết quả trả về của hàm là một con trỏ chỉ đến phần tử đầu tiên của chuỗi s1 có chứa kí tự c hoặc giá trị NULL nếu kí tự c không có trong chuỗi s1. 

• **Xây dựng hàm**:

• **Cú pháp**: **char \*strchr**(const char **\***s1, char s2)

• **Ví dụ**:



**6. Hàm strstr()**

• **Thư viện**: string.h

• **Mục đích**: Dùng để tìm kiếm sự xuất hiện đầu tiên của chuỗi s2 trong chuỗi s1. Kết quả trả về của hàm là một con trỏ chỉ đến phần tử đầu tiên của chuỗi s1 có chứa chuỗi s2 hoặc giá trị NULL nếu chuỗi s2 không có trong chuỗi s1. 

• **Xây dựng hàm:**

****• **Cú pháp**: **char \*strstr**(const char **\***s1, const char **\***s2)

• **Ví dụ**:



**7. Hàm strupr()**

• **Thư viện**: string.h

• **Mục đích**: Dùng để chuyển đổi chuỗi chữ thường thành chuỗi chữ hoa, kết quả trả về của hàm là một con trỏ chỉ đến địa chỉ chuỗi được chuyển đổi.

• **Xây dựng hàm:** 

****

• **Cú pháp**: **char \*strupr**(char **\***s)

• **Ví dụ**:



**8. Hàm strlwr()**

• **Thư viện**: string.h

• **Mục đích**: Muốn chuyển đổi chuỗi chữ hoa thành chuỗi toàn chữ thường, ta sử dụng hàm **strlwr**(), các tham số của hàm tương tự như hàm strupr(). • **Xây dựng hàm:**

****

• **Cú pháp: char \*strlwr**(char **\***s)

• **Ví dụ:**

****

**9. Hàm strrev()**

• **Thư viện:** string.h

• **Mục đích:** Hàm **strrev**(string) trả về một chuỗi được đảo ngược • **Xây dựng hàm:**

****• **Cú pháp: char \*strrev**(char **\***s) 

• **Ví dụ:**

****

10. **Hàm tolower() và hàm isupper();**

|  | **toupper()** | **islower()** |
| --- | --- | --- |
| **Thư viện** | **ctype.h** | **ctype.h** |
| **Mục đích** | Chuyển đổi các chữ cái hoa thành chữ cái thường | Kiểm tra chữ cái truyền vào có phải chữ cái hoa không |
| **Cú pháp** | **int tolower**(int c)**;** | **int isupper**(int c)**;** |

• **Xây dựng hàm:** 

****

• **Ví dụ:**

****

**11.Hàm toupper() và hàm islower();**

|  | **toupper()** | **islower()** |
| --- | --- | --- |
| **Thư viện** | **ctype.h** | **ctype.h** |
| **Mục đích** | Chuyển đổi các chữ cái hoa thành chữ cái thường | Kiểm tra chữ cái truyền vào có phải chữ cái hoa không |
| **Cú pháp** | **int toupper**(int c)**;** | **int islower**(int c)**;** |

• **Xây dựng hàm:** 

****• **Ví dụ:**

****