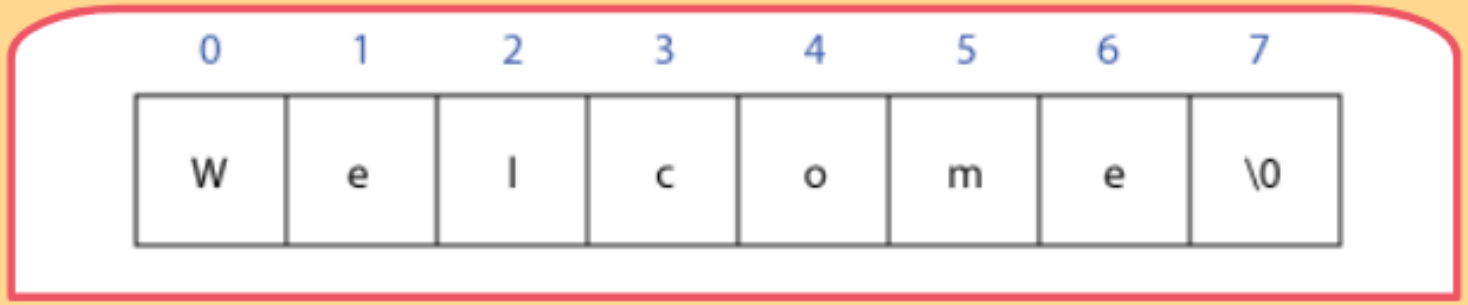
CHUỖI TRONG C



Định nghĩa

- Là một tập hợp các ký tự (**char**)
- Được lưu trữ trên các ô nhớ **liên tiếp**
- Luôn luôn có 1 ký tự **null** là **\0** cuối chuỗi.

```
char s[] = "xin chao";
```

x	i	n		c	h	a	o	\0

*# 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**.*


Khởi tạo

Khởi tạo

```
char a[] = "Full House chao moi nguoi";
char b[100] = "Full House chao moi nguoi";
char c[] = {'F', 'u', 'l', 'l', ' ','H', 'o', 'u', 's', 'e'};
char d[100] = {'F','u', 'l', 'l', ' ', 'H', 'o', 'u', 's', 'e'};
```

Nhập xuất cơ bản

Nhập xuất cơ bản

```
    char s1[1000];
    scanf("%s",&s1);
//  scanf("%s",s1);
    printf("%s\n",s1);

    char s2[1000], s3[1000];
    scanf("%s%s",s1,s2);
    printf("%s\n%s",s1,s2);
```

Nhập xuất Nâng cao

Nhập xuất Nâng cao

! Hãy nhận xét CÁC kí tự '\n' ?

```
char s1[1000], s2[1000];
fgets(s1,sizeof(s1),stdin);
gets(s2);

printf("%s%s\n",s1,s2);
```

Nhập:

```
Full House
Chao cac ban
```

Xuất:

```
Full House
Chao cac ban
Full House
Chao cac ban

--------------------------------
Process exited after 26.62 seconds
Press any key to continue . . .
```

Nhập xuất Nâng cao

```
char s1[1000], s2[1000];
fgets(s1,sizeof(s1),stdin);
gets(s2);

printf("%s%s\n",s1,s2);
puts(s2);
```

Nhập xuất Nâng cao

! Hãy nhận xét CÁC kí tự '\n' ?

```
char s1[1000], s2[1000];
fgets(s1,sizeof(s1),stdin);
gets(s2);

printf("%s%s\n",s1,s2);
puts(s2);
```

Nhập:

```
Full House
Chao cac ban
```

Xuất:

```
Full House
Chao cac ban
Full House
Chao cac ban
Chao cac ban

--------------------------------
Process exited after 17
```

Cho chương trình

Cho chương trình

Lỗi gì xảy ra? Tại sao?

```
float diem;
char Ho_ten[1000];

printf("Nhap diem: ");
scanf("%f",&diem);
printf("Nhap ho ten: ");
gets(Ho_ten);

printf("Ho ten: %s",Ho_ten);
printf("\nDiem: %.2f",diem);
```

Cho chương trình

```
float diem;
char Ho_ten[1000];

printf("Nhap diem: ");
scanf("%f",&diem);
printf("Nhap ho ten: ");
gets(Ho_ten);

printf("Ho ten: %s",Ho_ten);
printf("\nDiem: %.2f",diem);
```

```
int main(){
    float diem;
    char Ho_ten[1000];

    printf("Nhap diem: ");
    scanf("%f",&diem);
//  fflush(stdin);
    getchar();
    printf("Nhap ho ten: ");
    gets(Ho_ten);

    printf("Ho ten: %s",Ho_ten);
    printf("\nDiem: %.2f",diem);
}
```

Một số hàm hỗ trợ

strlen

strlen()	
Thư viện	string.h
Mục đích	
Xây dựng hàm	
Cú pháp	
Áp dụng	

strlen

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	
Áp dụng	

```c
#include <stdio.h>

int strlen(char s[]){
    int i = 0;
    for(;s[i]!='\0';++i);
    return i;
}

int main(){
    char s[1000];
    gets(s);
    printf("%d",strlen(s));
}
```

strlen

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)
Áp dụng	

```c
#include <stdio.h>

int strlen(char s[]){
	int i = 0;
	for(;s[i]!='\0';++i);
	return i;
}

int main(){
	char s[1000];
	gets(s);
	printf("%d",strlen(s));
}
```

strlen

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)
Áp dụng	

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

int main(){
    char s[1000];
    gets(s);
    printf("%d",strlen(s));
}
```

strcpy

strcpy()	
Thư viện	string.h
Mục đích	Sao chép nội dung của chu ỗi B tới chuỗi A
Xây dựng hàm	
Cú pháp	char *strcpy(char *A, const char *B)
Áp dụng	

```
char* strcpy(char s1[],char s2[]){
	int i, len_s2 = strlen(s2);
	for(i=0;i<strlen(s2);++i){
		s1[i] = s2[i];
	}
	s1[i]='\0';
	return s1;
}

int main(){
	char s1[1000], s2[1000];
	gets(s1);
	gets(s2);
	strcpy(s1,s2);
	printf("%s",s1);
	// printf("%s",strcpy(s1,s2));
}
```

strcpy

strcpy()	
Thư viện	string.h
Mục đích	Sao chép nội dung của chu ỗi B tới chuỗi A
Xây dựng hàm	
Cú pháp	char *strcpy(char *A, const char *B)
Áp dụng	

```
153 #include <stdio.h>
154 #include <string.h>
155
156 int main(){
157     char s1[1000],s2[1000];
158     gets(s1);
159     gets(s2);
160     strcpy(s1,s2);
161     printf("%s",s1);
162 //  printf("%s",strcpy(s1,s2));
163 }
```

strcat

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 *A, const char *B)
Áp dụng	

```
182 char* strcat(char s1[],char s2[]){
183     int i= 0, len_s1 = strlen(s1);
184     for(;i<strlen(s2);++i){
185         s1[len_s1+i] = s2[i];
186     }
187     s1[i+len_s1]='\0';
188     return s1;
189 }
190
191 int main(){
192     char s1[1000], s2[1000];
193     gets(s1);
194     gets(s2);
195     strcat(s1,s2);
196     printf("%s",s1);
197 //  printf("%s",strcat(s1,s2));
198 }
```

strcat

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 *A, const char *B)
Áp dụng	

```
200 #include <stdio.h>
201 #include <string.h>
202
203 int main(){
204     char s1[1000],s2[1000];
205     gets(s1);
206     gets(s2);
207     strcat(s1,s2);
208     printf("%s",s1);
209 //  printf("%s",strcat(s1,s2));
210 }
211
```

strcmp

strcmp()	
Thư viện	string.h
Mục đích	Dùng để so sánh hai chuỗi với nhau. (-1 0 1)
Xây dựng hàm	
Cú pháp	char *strcat(char *A, const char *B)
Áp dụng	

```
226 int strcmp(char s1[],char s2[]){
227     int len_s1 = strlen(s1);
228     int len_s2 = strlen(s2);
229     int out;
230     for(int i=0;i<len_s1 && i<len_s2;++i){
231         out = s1[i]-s2[i];
232         if(out) {
233             if(out<0) return -1;
234             else if(out>0) return 1;
235         }
236     }
237     if(len_s1<len_s2) return -1;
238     else if(len_s1>len_s2) return 1;
239     return 0;
240 }
241
242 int main(){
243     char s1[1000], s2[1000];
244     gets(s1);
245     gets(s2);
246     printf("%d",strcmp(s1,s2));
```

strcmp

strcmp()	
Thư viện	string.h
Mục đích	Dùng để so sánh hai chuỗi với nhau. (-1 0 1)
Xây dựng hàm	
Cú pháp	char *strcat(char *A, const char *B)
Áp dụng	

```
249 #include <stdio.h>
250 #include <string.h>
251
252 int main(){
253     char s1[1000],s2[1000];
254     gets(s1);
255     gets(s2);
256     printf("%d",strcmp(s1,s2));
257 }
258
```

strchr

strchr()	
Thư viện	string.h
Mục đích	Dùng để tìm kiếm sự xuất hiện đầu tiên của ki tự c trong chuỗi s1
Xây dựng hàm	
Cú pháp	char *strchr(const char *s1, char s2)
Áp dụng	

```c
char* strchr(char s1[],char c){
    int len_s1 = strlen(s1);
    for(int i=0;i<len_s1;++i){
        if(s1[i]==c){
            return s1+i;
        }
    }
    return NULL;
}

int main(){
    char s1[1000], c;
    gets(s1);
    scanf("%c",&c);
    if(strchr(s1,c)==NULL)
        printf("Khong tim thay ki tu trong s1");
    else printf("%s",strchr(s1,c));
}
```

strchr

strchr()	
Thư viện	string.h
Mục đích	Dùng để tìm kiếm sự xuất hiện đầu tiên của ki tự c trong chuỗi s1
Xây dựng hàm	
Cú pháp	char *strchr(const char *s1, char s2)
Áp dụng	

```
290 #include <stdio.h>
291 #include <string.h>
292
293 int main(){
294     char s1[1000],c;
295     gets(s1);
296     scanf("%c",&c);
297     if(strchr(s1,c)==NULL)
298         printf("Khong tim thay chuoi s2 trong s1");
299     else printf("%s",strchr(s1,c));
300 }
```

strstr

strstr()	
Thư viện	string.h
Mục đích	Dùng để tìm kiếm sự xuất hiện đầu tiên của chuỗi s2 trong chuỗi s1
Xây dựng hàm	
Cú pháp	char *strstr(const char *s1, const char *s2)
Áp dụng	

```
279 char* strstr(char s1[],char s2[]){
280     int len_s1 = strlen(s1);
281     int len_s2 = strlen(s2);
282     for(int i=0;i<=len_s1-len_s2;++i){
283         if(s1[i]==s2[0]){
284             int j=1, tmp = 1;
285             while(j<len_s2){
286                 if(s1[i+j]!=s2[j]){
287                     tmp =0;
288                     break;
289                 }
290                 ++j;
291             }
292             if(tmp) return s1+i;
293         }
294     }
295     return NULL;
296 }
297
298 int main(){
299     char s1[1000], s2[1000];
300     gets(s1);
301     gets(s2);
302     if(strstr(s1,s2)==NULL)
303         printf("Khong tim thay chuoi s2 trong s1");
304     else printf("%s",strstr(s1,s2));
305 }
```

strstr

strstr()	
Thư viện	string.h
Mục đích	Dùng để tìm kiếm sự xuất hiện đầu tiên của chuỗi s2 trong chuỗi s1
Xây dựng hàm	
Cú pháp	char *strstr(const char *s1, const char *s2)
Áp dụng	

```
307 #include <stdio.h>
308 #include <string.h>
309
310 int main(){
311     char s1[1000],s2[1000];
312     gets(s1);
313     gets(s2);
314     if(strstr(s1,s2)==NULL)
315         printf("Khong tim thay chuoi s2 trong s1");
316     else printf("%s",strstr(s1,s2));
317 }
318
```

strupr

strupr()	
Thư viện	string.h
Mục đích	Dùng để chuyển đổi chuỗi chữ th ường thành chuỗi chữ hoa
Xây dựng hàm	
Cú pháp	char *strupr(char *s)
Áp dụng	

```
372 char* strupr(char s1[]){
373     int len_s1 = strlen(s1);
374     for(int i=0;i<len_s1;++i){
375         if(s1[i]>='a'&&s1[i]<='z'){
376             s1[i] -=32;
377         }
378     }
379     return s1;
380 }
381
382 int main(){
383     char s1[1000];
384     gets(s1);
385     printf("%s",strupr(s1));
386 }
```

strupr

strupr()	
Thư viện	string.h
Mục đích	Dùng để chuyển đổi chuỗi chữ th ường thành chuỗi chữ hoa
Xây dựng hàm	
Cú pháp	char *strupr(char *s)
Áp dụng	

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

int main(){
    char s1[1000];
    gets(s1);
    printf("%s",strupr(s1));
}
```

strlwr

strlwr()	
Thư viện	string.h
Mục đích	Dùng để chuyển đổi chuỗi chữ hoa thành chuỗi chữ thường
Xây dựng hàm	
Cú pháp	char *strlwr(char *s)
Áp dụng	

```
407 char* strlwr(char s1[]){
408     int len_s1 = strlen(s1);
409     for(int i=0;i<len_s1;++i){
410         if(s1[i]>='A'&&s1[i]<='Z'){
411             s1[i] +=32;
412         }
413     }
414     return s1;
415 }
416
417 int main(){
418     char s1[1000];
419     gets(s1);
420     printf("%s",strlwr(s1));
421 }
```

strlwr

strlwr()	
Thư viện	string.h
Mục đích	Dùng để chuyển đổi chuỗi chữ hoa thành chuỗi chữ thường
Xây dựng hàm	
Cú pháp	char *strlwr(char *s)
Áp dụng	

```
23 #include <stdio.h>
24 #include <string.h>
25
26 int main(){
27     char s1[1000];
28     gets(s1);
29     printf("%s",strlwr(s1));
30 }
```

strrev

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)
Áp dụng	

```
445 char* strrev(char s1[]){
446     int len_s1 = strlen(s1);
447     for(int i=0;i<len_s1/2;++i){
448         char tmp = s1[i];
449         s1[i] = s1[len_s1-i-1];
450         s1[len_s1-i-1] = tmp;
451     }
452     return s1;
453 }
454
455 int main(){
456     char s1[1000];
457     gets(s1);
458     printf("%s",strrev(s1));
459 }
```

strrev

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)
Áp dụng	

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

int main(){
    char s1[1000];
    gets(s1);
    printf("%s",strrev(s1));
}
```

tolower và isupper

	tolower()	isupper()
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);

tolower và isupper

Xây dựng hàm:

```
int main(){
    char c;
    scanf("%c",&c);
    if(c>='A'&&c<='Z'){
        c += 32;
    }
    printf("%c",c);
}
```

Áp dụng:

```
#include <ctype.h>

int main(){
    char c;
    scanf("%c",&c);
    if(isupper(c)){
        c = tolower(c);
    }
    printf("%c",c);
}
```

toupper và islower

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

toupper và islower

Xây dựng hàm:

```
505 int main(){
506     char c;
507     scanf("%c",&c);
508     if(c>='a'&&c<='z'){
509         c -= 32;
510     }
511     printf("%c",c);
512 }
```

Áp dụng:

```
516 #include <ctype.h>
517
518 int main(){
519     char c;
520     scanf("%c",&c);
521     if(islower(c)){
522         c = toupper(c);
523     }
524     printf("%c",c);
525 }
```