02204171 Structured Programming

Chapter 9 : C String

Outline

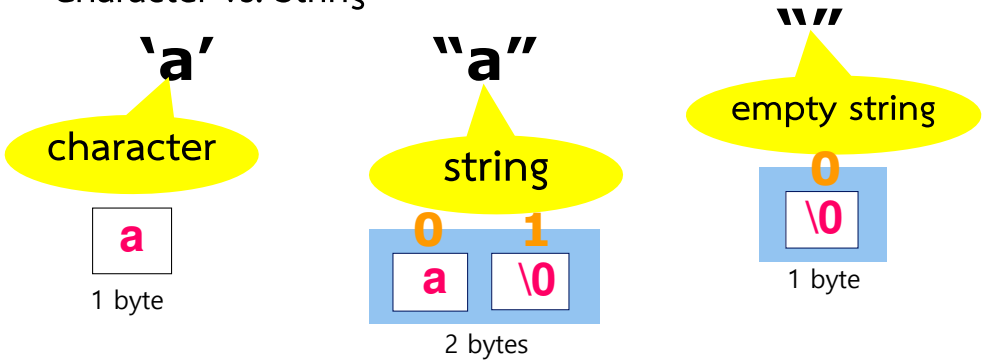
1. What is a String?

- String คือ ชุดของตัวอักษร (สายอักขระ) เรียงต่อกัน ซึ่งเราเรียกแทนว่า ข้อความ
- ในภาษา C ไม่มีข้อมูลชนิด String โดยตรง แต่จะมองเป็น array ของตัวอักษร
  - String ต่างกับ array ของตัวอักษรทั่วไป ตรงที่

ตัวอักขระตัวสุดท้ายจะต้องเป็นค่า Null (‘\0’)

1. What is a String?

- ค่าคงที่ของ String จะอยู่ภายใต้เครื่องหมาย “ ” (double quotes) เช่น “B+”, “CPE”, “Math”, “204171”
- Character vs. String

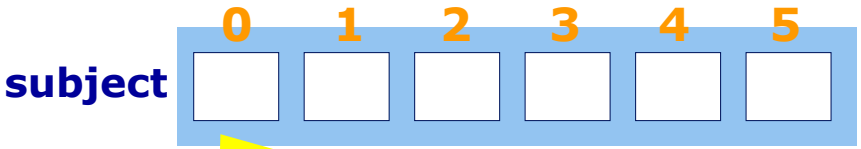


2. String Declaration and Initialization

- รูปแบบ(Syntax) การประกาศตัวแปรชนิด string:

```
char    ชื่อตัวแปร[จำนวนสมาชิก];
```

- ตัวอย่าง: char subject[6];



ประกาศ array ของ char เฉยๆ
ยังไม่รู้ว่าเป็น string หรือไม่

2. String Declaration and Initialization

- เราจะทราบว่าข้อมูลที่เก็บเป็น string ก็ต่อเมื่อกำหนดค่าเริ่มต้น: $n_{elements} \geq n_{chars\ in\ string} + 1$

```
char ชื่อตัวแปร[จำนวนสมาชิก]={ค่าที่1,ค่าที่2, ... , '\0'};
```

หรือ

```
char ชื่อตัวแปร[]="ข้อความที่ต้องการ";
```


1. What is a String?

- ตัวอย่างการประกาศ string

```
char arr1[]={'H','e','l','l','o','\0'};
char arr2[]="Hello";
```

Null เป็นตัวปิดท้าย
ข้อความ

Quick check

- จงวาดโครงสร้างและค่าที่เก็บในตัวแปรชนิด String ต่อไปนี้

```
char str[11]="Good Day";

char subject[]={'M','a','t','h','\0'};
```

3. Accessing element of String

- การเข้าถึงตัวอักขระใน String ทำได้โดย**ระบุหมายเลขของตัวอักขระ ซึ่งอักขระตัวแรกเริ่มต้นที่หมายเลข 0** (เสมือนการเข้าถึงสมาชิกอาเรย์ ของตัวอักขระ)

- **ตัวอย่าง 1**

```
char s[]="Hello";
printf("%c",s[0]);
printf("%c",s[1]);

s[3] = 'p';
s[4] = '!';

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

'H'	'e'	'l'	'l'	'o'	'\0'

3. Accessing element of String

- **ตัวอย่าง 2**

```
char ch;
char title[] = "Titanic";

ch = title[1];
title[3] = ch;

printf ("%s\n", title);

printf ("%c\n", ch);
```


3. Accessing element of String

- **ตัวอย่าง 3**

H	e	l	l	o		w	o	r	l	d	\0

```
int i;
char str1[] = "Hello world";
char str2[5];
str2[0]='E';
str2[1]='n';
str2[2]='g';
str2[3]='2';
str2[4]='\0';

printf("%s\n",str1);

for(i=0; i<5; i++)
   printf("str2[%d]==%c\n",i,str2[i]);
```


4. Strings I/O in C Programming

- Read & write Strings in C using **printf()** and **scanf()**

- Read & write Strings in C using **puts()** and **gets()**

4. Strings I/O: printf() and scanf()

- การรับค่า string โดยใช้ scanf()

scanf("%s", ชื่อตัวแปร);

ไม่มีเครื่องหมาย & เพราะชนิดตัวแปรอ้างถึงหน่วยความจำอยู่แล้ว

```
char str[12];
printf("Enter string:");
scanf("%s", str);
printf("%s", str);
```

```
Enter string:Hello World
Hello
```

\0 จะถูกเพิ่มเข้าไปอัตโนมัติ

str:	H	e	l	l	o	\0	?	?	?	?	?	?

scanf จะอ่านค่าของสตริงที่ป้อนจากคีย์บอร์ดจนกว่าจะป้อนช่องว่าง หรือเครื่องหมายขึ้นบรรทัดใหม่ ('\n') ทำให้ไม่สามารถรับข้อความที่มีช่องว่างอยู่ด้วยได้

4. Strings I/O: printf() and scanf()

- การรับค่า string โดยใช้ scanf()

สำหรับการรับ String ทั้งบรรทัด (ที่รวมช่องว่าง) โดยถือว่าข้อความที่ป้อนทั้งบรรทัดเป็นข้อความเดียวกัน ให้ใช้ format specifier คู่กันกับ "%s"

scanf("%[^\n]s", ชื่อตัวแปร);

กำหนดให้อ่านตัวอักษรทั้งบรรทัด ยกเว้น \n(newline)

scanf จะรับค่า string ที่ป้อนจนกว่าจะพบ '\n' จึงหยุดอ่าน

4. Strings I/O: printf() and scanf()

- ตัวอย่าง 4

J	o	h	n		S	n	o	w	\0

```
char name[10];
printf("Enter name: ");
scanf("%[^\n]s", name);
printf("Hello, %s.\n", name);
printf("|%30s|\n", name);
printf("|%-30s|", name);
```

```
Enter name:John Snow
Hello, John Snow.
```


4. Strings I/O: puts() and gets()

- gets() สำหรับรับ String ทั้งบรรทัด (รวมช่องว่าง) จนกว่าจะป้อน '\n'

gets(ชื่อตัวแปร);

- puts() สำหรับแสดง String และ '\n'(การขึ้นบรรทัดใหม่) ทางจอภาพ

puts(ชื่อตัวแปร/"ข้อความ");

4. Strings I/O: puts() and gets()

- ตัวอย่าง 5

J	o	h	n		S	n	o	w	\0

```
char name[10];
puts("Enter name: ");
gets (name);
puts("Hello, ");
puts (name);
```

```
Enter name:
John Snow
Hello,
John Snow
```

*** ข้อควรระวัง: ไม่ว่าจะ scanf() หรือ gets() หากความยาว String ที่ป้อนจากคีย์บอร์ดยาวกว่า*ขนาดของตัวแปรอาเรย์* ที่จะจัดเก็บอาจทำให้เกิดปัญหา buffer overflow(ความยาวข้อความที่รับมาเกินกว่าขนาดอาเรย์ที่ประกาศ)

4. Strings I/O in C Programming

- การป้องกันไม่ให้รับ String ที่มีความยาวเกินกว่าขนาดของอาเรย์ที่ประกาศ ทำได้โดยการ**ระบุขนาดของตัวอักษรที่จะรับ** ใน format specifier ของ **scanf()**

A	e	g	o	n		T	a	r	\0

```
char name[10];
printf("Enter name: ");
scanf("%9[^\n]s", name);
printf("Hello, %s.", name);
```

ขนาดที่ระบุ คือจำนวนสมาชิก -1

```
Enter name:Aegon Targaryen
Hello, Aegon Tar
```


5. String handling functions

- **ฟังก์ชั่นสำหรับจัดการ String ในภาษา C**

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

- **strlen()** แสดงขนาดความยาว String ไม่นับรวม '\0'
- **strcmp(), strncmp()** การเปรียบเทียบ String 2 ชุด (string compare)
- **strcat(), strncat()** การนำ String ชุดที่สองไปต่อท้าย String ชุดที่หนึ่ง (string concatenation)
- **strcpy(), strncpy()** การคัดลอก String (string copy)
- **strstr()** การค้นหา String ย่อยใน String หลัก

5. String handling functions

- การหาจำนวนตัวอักษรของ string ให้ใช้ฟังก์ชั่น **strlen()**

—ผลลัพธ์เป็นข้อมูลชนิด int **strlen(S1)**

(S1 คือตัวแปร หรือค่าคงที่ String)

- **ตัวอย่าง 6**

```
char str[11]="Good Day";
printf("length:%d", strlen(str));
```

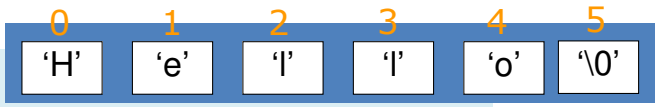
G	o	o	d		D	a	y	\0	\0	\0

```
length: 8
```

ตัดการนับที่ \0

5. String handling functions

- ตัวอย่าง 7




```
int i;
char s[]="Hello";
printf("%d\n",strlen(s));
printf("%c", s[0]);
for (i = 1; i < strlen(s); i = i+1)
{
    printf("-%c", s[i]);
}
```


5. String handling functions

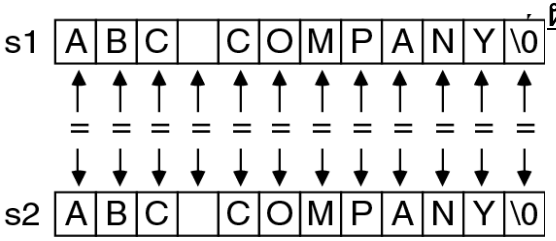
- การเปรียบเทียบ String 2 ชุด

ความยาวของ String ที่ต้องการเปรียบเทียบ

strcmp(S1,S2) **strncmp(S1,S2,size)**

(S1และ S2 คือตัวแปร หรือค่าคงที่ String ที่ต้องการเปรียบเทียบ)

การเปรียบเทียบ String ในภาษา C จะเปรียบเทียบทีละอักขระของ String ทั้งสองชุด ตามค่า ASCII Code



ผลลัพธ์ของการเปรียบเทียบ

S1 < S2	ค่าน้อยกว่า 0 (-1)
S1 > S2	ค่ามากกว่า 0 (1)
S1 == S2	ค่าเท่ากับ 0

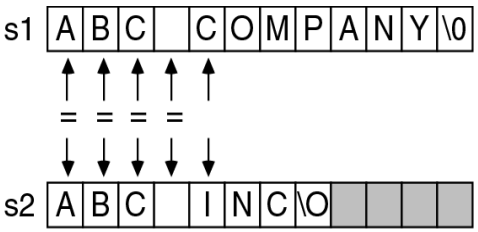
5. String handling functions

- การเปรียบเทียบ String 2 ชุด

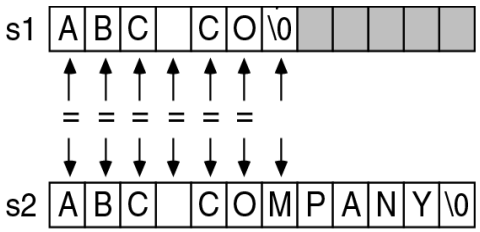
ทั้งสองฟังก์ชั่น เปรียบเทียบอักขระของ String สองชุด ทีละตัวตามค่า ASCII Code จนกว่าจะพบ**อักขระที่แตกต่างกัน** หรือ

พบ '\0'(สิ้นสุด String)/ครบจำนวนอักขระ(size)ที่ต้องการเปรียบเทียบ

strcmp(S1,S2)



strncmp(S1,S2,size)



5. String handling functions

- ตัวอย่าง การเปรียบเทียบ String 2 ชุด

S1	S2	ผลลัพธ์ strcmp(S1,S2)	ผลลัพธ์ strncmp(S1,S2,2)
"ABAA"	"ABCD"	**<0**	**=0**
"B123"	"A089"	**>0**	**>0**
"4178"	"459"	**<0**	**<0**
"abc888"	"abc888"	**=0**	**=0**
"AbC"	"Abcde"	**<0**	**=0**

5. String handling functions

- **ตัวอย่าง 8: การเขียน *strcmp()* เพื่อเปรียบเทียบ**

```
char passwd[9];
printf("Enter password: ");
scanf("%8s", passwd);
if(!strcmp("1q2w3e4r",passwd))
   puts("Login complete ...");
else
   puts("Incorrect password!");
```

กรณีที่ String 2 ชุด เหมือนกันผลการเปรียบเทียบ มีค่าเท่ากับ 0 (ซึ่งมีค่าเป็น false)

5. String handling functions

- **การต่อ String:** นำ String ชุดที่สองไปต่อท้าย String ชุดที่หนึ่ง โดยตัวอักษรตัวแรกของสตริงชุดที่สอง จะทับ null character

strcat(S1,S2) **strncat(S1,S2,size)**

(S1 คือตัวแปร, S2 คือตัวแปร หรือค่าคงที่ String)

ความยาวของ String ชุดที่สอง ที่ต้องการต่อ

C	o	n	\0	?	?	?	?	?	?	?	?	?	?

c	a	t	e	n	a	t	i	o	n	\0	?	?	?

strcat(S1,S2)

C	o	n	c	a	t	e	n	a	t	i	o	n	\0

strncat(S1,S2,3)

C	o	n	c	a	t	\0	?	?	?	?	?	?	?

Quick Check2

- แสดงผลการทำงานของโปรแกรมต่อไปนี้

```
char  s1[14]="",s2[14]="HaHaHa";

while(strcmp(s1,s2)!=0)
      strcat(s1,"Ha");

puts(s1);
```


5. String handling functions

- **การคัดลอก String:** คัดลอก String ชุดที่สองไปเก็บไว้ใน String ชุดแรก ไม่สามารถกำหนดค่า String โดยใช้ assignment statement เช่น

```
char  str1[] = "Test String";
char  str2[12];
str2 = str1;
str2 = "Test String";
```

✗

```
strcpy(str2,str1);
strcpy(str2,"Test String");
```

✓

strcpy(S1,S2) **strncpy(S1,S2,size)**

(S1 คือตัวแปร, S2 คือตัวแปร หรือค่าคงที่ String)

ความยาวของ String ชุดที่สอง ที่ต้องการคัดลอก

จะคัดลอกตัวอักขระใน string ตั้งแต่ตัวแรกไปเรื่อยๆ จนกว่า

จะพบ '/0' หรือ **ครบความยาว string ที่ต้องการคัดลอก** จึงหยุด

5. String handling functions

- **การคัดลอก String:** คัดลอก String ชุดที่สองไปเก็บไว้ใน String ชุดแรก

```
char name1[16], name2[16];
strcpy(name1,"Spider Man");
```

name1:

S	p	i	d	e	r		M	a	n	\0	\0	\0	\0	\0	\0

```
strcpy(name2,"999999999999999");
```

name2:

9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	\0

```
strncpy(name2,name1,7);
```

name2:

S	p	i	d	e	r		9	9	9	9	9	9	9	9	\0

5. String handling functions

- **การค้นหา String ย่อยใน String หลัก**

strstr(S1,S2)

(S1 คือตัวแปร,
S2 คือตัวแปร หรือค่าคงที่ String)

ผลลัพธ์ของฟังก์ชั่น หากหาเจอ คือ ตำแหน่งแรก (Memory address) ที่พบ String *S2* ซึ่งเรียกว่า *Pointer* หากหาไม่เจอจะมีค่าเป็น *NULL*

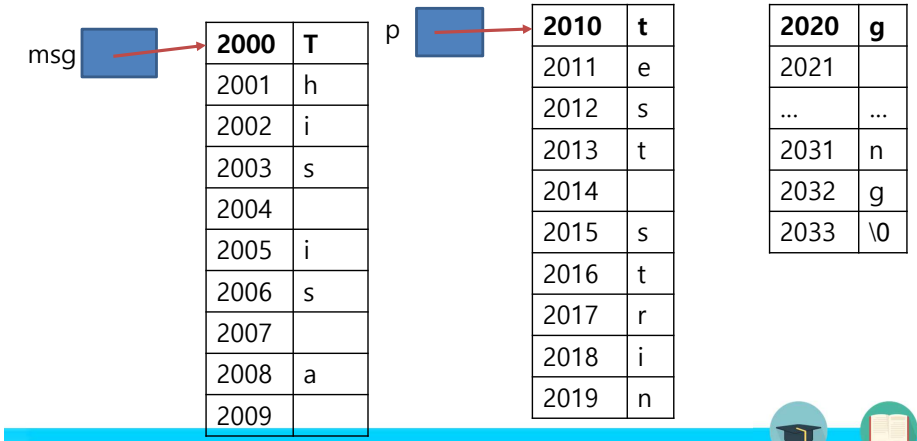
```
char msg[] ="This is a test string for testing";
char *p;
p = strstr (msg,"test");
if(p)  {
 printf("string found\n" );
 printf ("First occurrence of string \"test\" is \"%s\"", p);
}
else printf("string not found\n" );
```

```
string found
First occurrence of string "test" is "test string for testing"
```


5. String handling functions

- Memory Management Explanation

```
char msg[] ="This is a test string for testing";
char *p;
p = strstr (msg,"test");
```



2000	T
2001	h
2002	i
2003	s
2004	
2005	i
2006	s
2007	
2008	a
2009	

2010	t
2011	e
2012	s
2013	t
2014	
2015	s
2016	t
2017	r
2018	i
2019	n

2020	g
2021	
...	...
2031	n
2032	g
2033	\0

5. String handling functions

- Summary

String manipulation	Function	Parameter Type	Number of Parameter	Return type
การหาความยาวข้อความ	**strlen()**	string 1 ชุด	1	จำนวนเต็ม
การเปรียบเทียบข้อความ	**strcmp()** **strncmp()**	string 2 ชุด string 2 ชุด จำนวนเต็ม 1 ตัว	2 3	จำนวนเต็ม
การต่อข้อความ	**strcat()** **strncat()**	string 2 ชุด string 2 ชุด จำนวนเต็ม 1 ตัว	2 3	string
การคัดลอก	**strcpy()** **strncpy()**	string 2 ชุด string 2 ชุด จำนวนเต็ม 1 ตัว	2 3	string
การค้นหา	**strstr()**	string 1 ชุด	1	Pointer

5. String handling functions

ยังมี ฟังก์ชั่นในการดำเนินการกับ string อีกมาก นิสิตสามารถศึกษาได้จาก

- http://www.c4learn.com/index/string-operations-or-character-array-in/
- https://www.tutorialspoint.com/c_standard_library/string_h.htm