

# String Function

Assembled for 204111  
by Kittipitch Kuptavanich

w13-Lec

## String Input and Output (2)

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

- **scanf() - %c**  
**getchar()** อ่าน characters เข้ามาหนึ่งตัวอักษร (อ่านทีละตัวอักษร)  
int c;  
c = getchar(); หรือ scanf("%c",&c);
- **scanf() - %s** อ่าน characters เข้ามาจนเจอ **blank space** หรือ newline (บรรทัดใหม่) – (อ่านทีละคำ)  
scanf("%s", message); //ไม่ต้องใส่ &
- **gets()** อ่าน characters เข้ามาจนเจอ newline (บรรทัดใหม่) – (อ่านทีละบรรทัด)
- **printf()** และ **puts()** สามารถใช้แทนกันได้  
printf("%s\n", message); ≡ puts(message);

## String Input and Output

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

- การอ่าน **input** string:
  - gets()
  - scanf()
  - getchar()
- การแสดงผล **output** string
  - puts()
  - printf()
  - putchar()

## String Input and Output (3)

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



Program 9.1

```
1 #include <stdio.h>
2 int main()
3 {
4     #define MSIZE 81
5     char message[MSIZE]; /* enough storage for 80 characters plus '\0' */
6
7     printf("Enter a string:\n");
8     gets(message);
9     printf("The string just entered is:\n");
10    puts(message);
11
12    return 0;
13 }
```

### Output

```
Enter a string:
This is a test input of a string of characters.
The string just entered is:
This is a test input of a string of characters.
```

## String Encoding

String array	Stored codes	Expression	Value
t	116	string2[0]	116
h	104	string2[1]	104
i	105	string2[2]	105
s	115		
	32		
i	105		
s	115		
	32		
a	97		
	32		
s	115		
t	116		
r	114		
i	105		
n	110		
g	103	string2[15]	103
\0	0	string2[16]	0

- สังเกตว่า ตัว null character '\0' มีค่าเป็น 0

Figure 9.4 The ASCII codes used to store this is a string A First Book of ANSI C, 4th Edition

5

## String Formatting

- char happy[] = "Have a Happy Day"
- printf("|%25s|", happy);
  - | Have a Happy Day |
- printf("|%-25s|", happy);
  - |Have a Happy Day |
- printf("|%25.12s|", happy);
  - | Have a Happy |
- printf("|%.12s|", happy);
  - |Have a Happy|

7

## String Input & Output [4]

- ตัวอย่างการอ่าน string ทีละตัวอักษร (getchar)

```
#define LSIZE 81
char message[LSIZE]; /* enough storage for 80 characters plus '\0' */
char c;
int i;

printf("Enter a string:\n");
i = 0;
while(i < (LSIZE-1) && (c = getchar()) != '\n')
{
    message[i] = c; /* store the character entered */
    i++;
}
message[i] = '\0'; /* terminate the string */
printf("The string just entered is:\n");
puts(message);

return 0;
```

สังเกตว่า ต้องอ่านค่ามาเก็บไว้ที่ c ก่อนจึงนำมาเปรียบเทียบกับเป็น '\n' หรือไม่ (วงเล็บ)

A First Book of ANSI C, 4th Edition

6

## String Library Function

- มี 3 กลุ่มใหญ่
  - str\_() ทำการเปลี่ยนแปลงทั้ง string
    - ทำทุกอักขรจนถึง NULL
  - strn\_() ทำการเปลี่ยนแปลงเฉพาะบางส่วน
    - ทำตัวอักษร n ตัวตามที่ระบุมา ที่ไม่ใช่ NULL
  - mem\_()
    - ทำตามกลุ่มอักขรที่กำหนดมา ไม่พิจารณาว่ามี NULL หรือไม่
    - ใช้น้อยกว่าสองกลุ่มแรก

8

#include &lt;string.h&gt;

## strlen()

- Return ความยาว (**length**) ของ string

- strlen(char \*)

```
/* strlen.c */
```

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

```
int main()
{
    char *t = "XXX";
    printf( "Length of <%s> is %d.\n", t, strlen( t ) );
}
```

```
output:
Length of <XXX> is 3.
```

- Note:** ความยาวของ string ไม่จำเป็นต้องเท่ากับขนาดของ array

A First Book of ANSI C, 4th Edition

9

#include &lt;string.h&gt;

## strcat()[2]

- strcat(char \*s1, char\* s2)
  - Function จะไม่ทำการเช็ค ว่า s1 มีขนาดพอหรือไม่ที่จะนำ s2 ไปต่อท้าย
    - หาก array s1 มีความยาวไม่พอก็จะเกิด error
- strncat(char \*s1, char\* s2, int n)
  - นำ ตัวอักษร n ตัว จาก s2 ไปต่อท้าย s1

A First Book of ANSI C, 4th Edition

11

#include &lt;string.h&gt;

## strcat()

- นำ string s2 มาต่อท้าย (**concatenate**) string s1
  - strcat(char \*s1, char\* s2)

```
/* strcat.c */
```

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

```
int main()
{
    char s1[50],
          s2[50];
    strcpy( s1, "Tweedledee " );
    strcpy( s2, "Tweedledum" );
    strcat( s1, s2 );
    puts( s1 );
}
```

```
output:
Tweedledee Tweedledum
```

A First Book of ANSI C, 4th Edition

10

#include &lt;string.h&gt;

## strcmp()

- เปรียบเทียบระหว่าง (**compare**) s1 และ s2
  - strcmp(char \*s1, char\* s2)

```
#define ANSWER "blue"
```

```
int main()
{
    char t[100];
    puts( "What is the secret color?" );
    gets( t );
    while ( strcmp( t, ANSWER ) != 0 )
    {
        puts( "Wrong, try again." );
        gets( t );
    }
    puts( "Right!" );
}
```

A First Book of ANSI C, 4th Edition

12

## strcmp()[2]

- strcmp(char \*s1, char\* s2)
  - ถ้า s1 มาก่อน s2 (ตามตัวอักษร)
    - strcmp("bat", "cat")
    - return ค่าลบ (< 0)
  - ถ้า s1 เหมือนกับ s2
    - strcmp("cat", "cat")
    - return 0
  - ถ้า s1 มาหลัง s2 (ตามตัวอักษร)
    - strcmp("cat", "bat")
    - return ค่าบวก (> 0)

## strcpy()

- คัดลอก (copy) ข้อมูลจาก s2 ไปยัง s1
  - strcpy(char \*s1, char\* s2)

```
int main()
{
    char s1[100],
        s2[100];
    strcpy( s1, "xxxxxx 1" );
    strcpy( s2, "zzzzzz 2" );

    puts( "Original strings: " );
    puts( "" );
    puts( s1 );
    puts( s2 );
    puts( "" );

    strcpy( s1, s2 );

    puts( "New strings: " );
    puts( "" );
    puts( s1 );
    puts( s2 );
}
```

```
output:
Original strings:
xxxxxx 1
zzzzzz 2

New strings:
zzzzzz 2
zzzzzz 2
```

## strcmp()[3]

- stricmp(char \*s1, char\* s2)
  - เปรียบเทียบโดยไม่พิจารณา case
    - case insensitive ('a' มีค่าเท่ากับ 'A')
- strncmp(char \*s1, char\* s2, int n)
  - เปรียบเทียบระหว่าง n อักษรแรกจาก s1 และ s2
    - strncmp("caterpillar", "category", 3)
      - จะได้ค่า 0
- strnicmp() – n อักษรแรก + case insensitive

## strcpy()[2]

- strcpy(char \*s1, char\* s2)
  - Function จะไม่ทำการเช็ค ว่า s1 มีขนาดพอหรือไม่ที่จะ copy s2 ไปใส่
    - หาก array s1 มีขนาดไม่พอก็จะเกิด error
- strncpy(char \*s1, char\* s2, int n)
  - คัดลอกตัวอักษรแรก n ตัว จาก s2 ไปยัง s1

## strchr()

- หาตำแหน่งแรกของอักษร (character) c ใน s1
  - char \*strchr(char \*s1, char c)

```
int main()
{
    char *t = "MEAS:VOLT:DC?";
    char *p;
    p = t;
    puts( p );
    while(( p = strchr( p, ':' ) ) != NULL )
    {
        puts( ++p );
    }
}
```

```
output:
MEAS:VOLT:DC?
VOLT:DC?
DC?
```

- ค่าที่ return จะเป็น pointer ไปยังตำแหน่งของ c

## strtok()

- ใช้ delim แบ่ง string s1 ออกเป็นส่วน ๆ (token)
- char \*strtok(char \*s1, constant char\* delim)

```
int main()
{
    const char str[80] = "hormones:the:series";
    const char* s = ":";
    char *token;

    /* get the first token */
    token = strtok(str, s);

    /* walk through other tokens */
    while( token != NULL )
    {
        printf( "%s\n", token );
        token = strtok( NULL, s );
    }

    return(0);
}
```

```
output:
hormones
the
series
```

ให้ค้นหาต่อจากตำแหน่งก่อนหน้า

## strchr() [2]

- strchr(char \*s1, char c)
  - หาตำแหน่งแรกจากทางขวา (right) ของอักษร c ใน string s1
- strstr(char \*s1, char\* s2)
  - หาตำแหน่งแรกของ string s2 ใน string s1

## Lesser Used Function

- strupr(char \*s1)
  - เปลี่ยนตัวอักษรใน s1 ให้เป็นตัวพิมพ์ใหญ่ (upper case) ทั้งหมด
- strlwr(char \*s1)
  - เปลี่ยนตัวอักษรใน s1 ให้เป็นตัวพิมพ์เล็ก (lower case) ทั้งหมด
- strrev(char \*s1)
  - กลับลำดับตัวอักษร (reverse) ใน s1 (จาก "bat" เป็น "tab")

## Conversion

- `int atoi(char *s1)`
  - เปลี่ยน ASCII string เป็น integer
  - `atoi("1234")`
- `double atof(char *s1)`
  - เปลี่ยน ASCII string เป็น เลขทศนิยมชนิด double (double-precision floating-point)
  - `atof("12.34")`
- `char [] itoa(int x, char*,int base)`
  - เปลี่ยน integer เป็น ASCII string
  - `itoa(1234,s,10)`

## Character Classification [2]

- `isblank(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นวรรค (blank space หรือ tab) หรือไม่ (return 0 กรณีไม่ใช่)
- `isspace(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นอักขระว่าง (space, `\t`, `\n`, `\f`, `\r`) หรือไม่ (return 0 กรณีไม่ใช่)
- `ispunct(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นเครื่องหมายวรรคตอน (punctuation mark) หรือไม่ (return 0 กรณีไม่ใช่)
  - เครื่องหมายวรรคตอนได้แก่
    - `!"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~`

## Character Classification

- การแยกประเภทอักขระ (`char`)
- `isalpha(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นตัวอักษร (alphabet) หรือไม่ (return 0 กรณีไม่ใช่)
- `isdigit(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นตัวเลข (digit) หรือไม่ (return 0 กรณีไม่ใช่)
- `isalnum(char c)`
  - ตรวจสอบดูว่าอักขระ `c` เป็นตัวอักษร (alphabet) หรือตัวเลข (number) หรือไม่ (return 0 กรณีไม่ใช่)

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

```
#define MAX_FORMAT_LEN 10
//ไม่เกิน 100 ตำแหน่ง
#define PLACE_STRING_WIDTH 3
```

```
int main()
{
    double i = 1.2345678912;
    int place;
    printf("Enter n: ");
    scanf("%d",&place);
    char place_s[PLACE_STRING_WIDTH];
    char formatH[MAX_FORMAT_LEN];

    strcpy(formatH,"%."); //formatH = "%. "

    itoa(place,place_s,10); //เปลี่ยนจากตัวเลขเป็น char array เช่น 13 เป็น "13"
    strcat(formatH,place_s); //formatH = "%.XX"

    strcat(formatH,"f"); //formatH = "%.XXf"
    printf(formatH,i);

    return 0;
}
```

## Example:

## Dynamic Decimal Places

```
#include <stdio.h>
#include <stdlib.h>
```

## Example:

### Dynamic Decimal Places [2]

```
int main()
{
    double i = 1.2345678912;
    int place;
    printf("Enter n: ");
    scanf("%d",&place);

    char formatH[] = "%.000f";

    formatH[2] = '0' + (place / 100);
    formatH[3] = '0' + (place / 10 % 10);
    formatH[4] = '0' + (place % 10);

    printf(formatH,i);

    return 0;
}
```

Without string.h