ไลบรารี่พื้นฐาน

Standard C Library

เอกสารประกอบการสอนวิชา CP111

มหาวิทยาลัยศรีนครินทรวิโรฒ

เนื้อหา

Agenda

- การจัดการกับสตริง
- การจัดการกับไฟล์

การจัดการกับสตริง

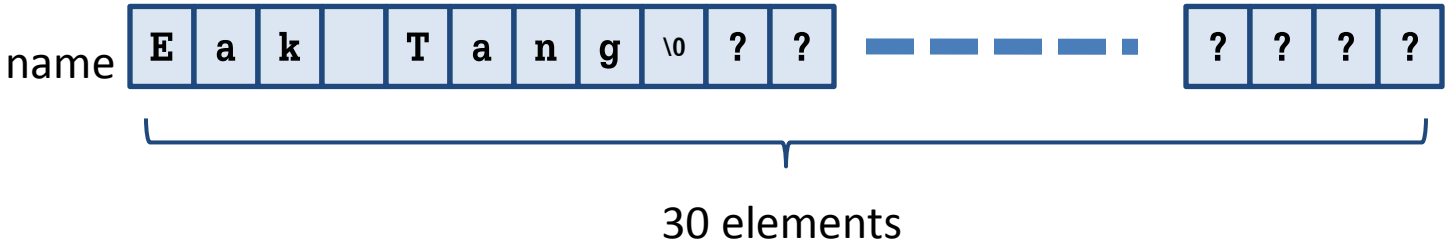
String Manipulation

ทบทวนสตริง

String Revision

- เป็นอาเรย์ของตัวแปรประเภท char
- ตัวอักษรตัวสุดท้ายเป็น '\0' (null)

```
char name[30] = "Ake Tang";
```



ทบทวนสตริง

String Revision

❑ บรรทัดที่ขีดเส้นใต้เป็นวิธีการเขียนที่ผิด

- char a[30];

  a = "Ake"; (การกำหนดค่า)

- char a[30] = "Ake";

  char b[30];

  b = a; (การกำหนดค่า)

- char a[30] = "Ake";

  char b[30] = "Tang";

  char c[30] = a + b; (การต่อสตริง)

ทบทวนสตริง

String Revision

- บรรทัดที่ขีดเส้นใต้เป็นวิธีการเขียนที่ผิด

```
char a[30] = "Ake";
char b[30];
scanf("%s", &b);
if (a == b)                    (การเปรียบเทียบ)
    printf("Equals");
```

ฟังก์ชั่นสำหรับจัดการกับสตริง

String Manipulating Functions

- การคัดลอกสตริง (String Copy)

  strcpy, strncpy

- การต่อสตริง (String Concatenation)

  strcat, strncat

- การเปรียบเทียบสตริง (String Comparison)

  strcmp, strncmp

- อื่นๆ (Others)

  strlen, strnlen, gets, sprintf

การคัดลอกสตริง

String Copy

- char *strcpy(char **strDestination*, const char **strSource*);

  strDest Destination string.

  strSource Source string.

  The function returns *strDest*. No return value is reserved to indicate an error.

ตัวอย่างการคัดลอกสตริง

String Copy Example

ex08_01.c

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

void main() {
   char a[30] = "Ake";
   char b[30];
   char c[30];

   strcpy(b, a);
   strcpy(c, "Tang");

   printf("%s\n%s\n%s\n", a, b, c);
}
```

การต่อสตริง

String Concatenation

- char *strcat(char ***strDestination***, const char ***strSource***);

strDest	Null-terminated destination string.
strSource	Null-terminated source string.

The function returns a pointer to the destination string.

No return value is reserved to indicate an error.

ตัวอย่างการต่อสตริง

String Concatenation Example

ex08_02.c

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

void main() {
   char a[30] = "Ake";
   char b[30] = "Tang";
   char name[30];
   strcpy(name, a);
   strcat(name, " ");
   strcat(name, b);

    printf("%s\n%s\n%s\n", a, b, name);
}
```

การเปรียบเทียบสตริง

String Comparison

- **int strcmp(const char *_string1_, const char *_string2_);**

  ***string1*, *string2* Null-terminated strings to compare.**

  The return value for each of the function indicates the lexicographic relation of *string1* to *string2*.

  < 0 ***string1* less than *string2***

  0 ***string1* identical to *string2***

  > 0 ***string1* greater than *string2***

ตัวอย่างการเปรียบเทียบสตริง

String Comparison Example

ex08_03.c

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

void main() {
   char a[30], b[30];
   int cmp;

   printf("a = "); gets(a);
   printf("b = "); gets(b);
   cmp = strcmp(a, b);

   if (cmp == 0)
      printf("a equals b\n");
   else if (cmp < 0)
      printf("a comes before b\n");
   else
      printf("b comes before a\n");
}
```

ฟังก์ชั่นอื่นๆ สำหรับสตริง

Other String Manipulation Functions

- **size_t strlen(const char *_string_);**

  string Null-terminated string.

  The function returns the number of characters in _string_, excluding the terminal NULL. No return value is reserved to indicate an error.

ฟังก์ชั่นอื่นๆ สำหรับสตริง

Other String Manipulation Functions

- int sprintf(char **buffer*, const char **format* [, *argument*] ...);

buffer	Storage location for output.
format	Format-control string.
argument	Optional arguments.

sprintf returns the number of bytes stored in *buffer*, not counting the terminating null character.

ตัวอย่าง

ex08_04.c

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

void main() {
   int a, b;
   char buf[30];

   printf("a = "); scanf("%d", &a);
   printf("b = "); scanf("%d", &b);
   sprintf(buf, "%d + %d = %d", a, b, a+b);

   printf("buf = %s\n", buf);
   printf("strlen(buf) = %d\n", strlen(buf));
}
```

การจัดการกับไฟล์

File Manipulation

ไฟล์

File

- เป็นรูปแบบของการจัดเก็บข้อมูลลงในหน่วยความจำถาวร
- ปกติที่ผ่านมาเราเก็บค่าต่างๆ ไว้ในหน่วยความจำชั่วคราวผ่านทางตัวแปร ซึ่งถ้าออกโปรแกรมหรือปิดเครื่องข้อมูลก็จะหายไป

- ตัวอย่างหน่วยความจำถาวร
  - Harddisk, Floppy disk, CD-ROM
- ตัวอย่างหน่วยความจำชั่วคราว
  - Ram, Cache

รูปแบบของไฟล์

File Type

- **Text File (ASCII File)**
  - หมายถึงไฟล์ที่จัดเก็บข้อมูลประเภทตัวอักษร
  - ปัจจุบันมีการแตกประเภท Text File เป็น Unicode Text File และ Non-Unicode Text File ด้วย
- **Binary File**
  - หมายถึงไฟล์ทั่วๆ ไป
  - Text file ถือว่าเป็น binary file ด้วย

การจัดการกับไฟล์

File Manipulation

- การเปิดและปิดไฟล์

  fopen, fclose

- การอ่านและเขียนไฟล์ประเภท **ASCII**

  fscanf, fprintf

- การอ่านและเขียนไฟล์ประเภท **binary**

  fread, fwrite

การเปิดไฟล์

- **FILE *fopen(const char* *filename*, const char* *mode*);**

  filename **Filename.**

  mode **Type of access permitted.**

  The function returns a pointer to the open file. A null pointer value indicates an error.

การเปิดไฟล์

- *mode* Type of access permitted.
  - "r" Opens for reading. If the file does not exist or cannot be found, the fopen call fails.
  - "w" Opens an empty file for writing. If the file exists, its contents are destroyed.
  - "a" Opens for writing at the end of the file (appending) without removing the EOF marker before writing new data to the file; creates the file first if it doesn't exist.
  - t Open in text (translated) mode. In this mode, CTRL+Z is interpreted as an end-of-file character on input. ... (more)
  - b Open in binary (untranslated) mode; translations involving carriage-return and linefeed characters are suppressed. ... (more)
- If t or b is not given in mode, the default translation mode is defined by the global variable _fmode. If t or b is prefixed to the argument, the function fails and returns NULL.

การปิดไฟล์

- **int fclose(FILE* *stream*);**

  stream **Pointer to FILE structure.**

  fclose returns 0 if the stream is successfully closed.

การเขียนไฟล์ (Text file)

- **int fprintf(FILE* *stream*, const char* *format* [, *argument*]...);**

stream	**Pointer to FILE structure.**
format	**Format-control string.**
argument	**Optional arguments.**

fprintf returns the number of bytes written.

ตัวอย่างการเขียนไฟล์ (Text file)

ex08_05.c

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

void main() {
   FILE * fp = fopen("test.txt", "wt");
   fprintf(fp, "Hello World!!\n");
   fclose(fp);
}
```

การอ่านไฟล์ (Text file)

- **int fscanf(FILE* *stream*, const char* *format* [, *argument*]...);**

stream	**Pointer to FILE structure.**
format	**Format-control string.**
argument	**Optional arguments.**

The function returns the number of fields successfully converted and assigned; the return value does not include fields that were read but not assigned. A return value of 0 indicates that no fields were assigned. If an error occurs, or if the end of the file stream is reached before the first conversion, the return value is EOF.

ตัวอย่างการอ่านไฟล์ (Text file)

ex08_06.c

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

void write() {
   FILE * fp = fopen("test.txt", "wt");
   fprintf(fp, "10 100\n");
   fprintf(fp, "11 120\n");
   fprintf(fp, "12 130\n");
   fclose(fp);
}

void read() {
   int a, b, ret;
   FILE * fp = fopen("test.txt", "rt");

   while (1) {
      ret = fscanf(fp, "%d %d", &a, &b);
      if (ret == EOF)
           break;
      printf("a = %d, b = %d\n", a, b);
   }
}
```

(#2)

```
void main() {
   write();
   read();
}
```

การเขียนไฟล์ (Binary file)

- size_t fwrite(const void* *buffer*, size_t *size*, size_t *count*, FILE* *stream*);

buffer	Pointer to data to be written.
size	Item size in bytes.
count	Maximum number of items to be written.
stream	Pointer to FILE structure.

fwrite returns the number of full items actually written, which might be less than count if an error occurs. Also, if an error occurs, the file-position indicator cannot be determined.

การอ่านไฟล์ (Binary file)

- size_t fread(void* *buffer*, size_t *size*, size_t *count*, FILE* *stream*);

buffer	Storage location for data.
size	Item size in bytes.
count	Maximum number of items to be read.
stream	Pointer to FILE structure.

fread returns the number of full items actually read, which may be less than count if an error occurs or if the end of the file is encountered before reaching count. Use the feof or ferror function to distinguish a read error from an end-of-file condition. If size or count is 0, fread returns 0 and the buffer contents are unchanged.

ตัวอย่างการอ่านเขียนไฟล์ (Binary file)

ex08_07.c

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

typedef struct _Duple {
   int a;
   int b;
} Duple;

void write() {
   Duple d;
   FILE * fp = fopen("test.txt", "wb");
   d.a = 10; d.b = 100;
   fwrite(&d, sizeof(Duple), 1, fp);
   d.a = 11; d.b = 120;
   fwrite(&d, sizeof(Duple), 1, fp);
   d.a = 12; d.b = 130;
   fwrite(&d, sizeof(Duple), 1, fp);
   fclose(fp);
}
```

(#2)

```
void read() {
   Duple d;
   FILE * fp = fopen("test.txt", "rb");

   while (1) {
      fread(&d, sizeof(Duple), 1, fp);
      if (feof(fp))
         break;
      printf("a = %d, b = %d\n",
         d.a, d.b);
   }
   fclose(fp);
}

void main() {
   write();
   read();
}
```

ตัวอย่างการอ่านเขียนไฟล์ (Binary file)

ex08_08.c

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

typedef struct _Duple {
   int a;
   int b;
} Duple;

void write() {
   Duple d;
   FILE * fp = fopen("test.txt", "wb");
   d.a = 10; d.b = 100;
   fwrite(&d, sizeof(Duple), 1, fp);
   d.a = 11; d.b = 120;
   fwrite(&d, sizeof(Duple), 1, fp);
   d.a = 12; d.b = 130;
   fwrite(&d, sizeof(Duple), 1, fp);
   fclose(fp);
}
```

(#2)

```
void read() {
   Duple d[3];
   int i;
   FILE * fp = fopen("test.txt", "rb");

   fread(&d, sizeof(Duple), 3, fp);
   fclose(fp);

   for (i = 0; i < 3; i++)
      printf("a = %d, b = %d\n",
         d[i].a, d[i].b);
}

void main() {
   write();
   read();
}
```

จบบทที่ 8

The End of Chapter 8

บทที่ 9 ...

...