

บทที่ 13 เพิ่มข้อมูล

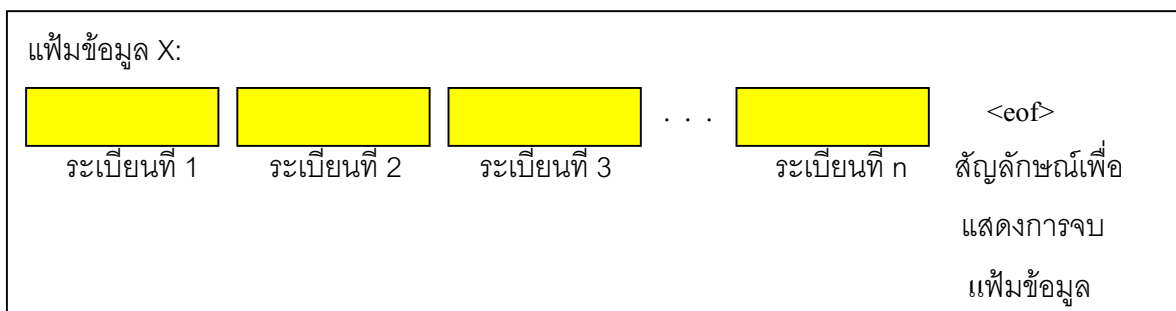
(Files)

ในบทที่ผ่านมา การทำงานในโปรแกรมส่วนใหญ่ มีการดำเนินการโดยนำข้อมูลเข้าทางแป้นพิมพ์เป็นอักขระ และแสดงผลทางจอภาพ ข้อมูลที่นำมาใช้อาจจะเป็นการกำหนดค่าเริ่มต้นภายในโปรแกรม หรือรับข้อมูลเข้าทางแป้นพิมพ์ การรันโปรแกรมที่ใช้แค่การป้อนข้อมูลในลักษณะนี้ เมื่อสิ้นสุดการทำงานของโปรแกรม ข้อมูลนำเข้าจะไม่สามารถเรียกมาดำเนินการซ้ำได้ แต่บางครั้งเราอาจจำเป็นต้องเก็บข้อมูลนำเข้าไว้ดำเนินการต่อไปในภายหลัง ในกรณีนี้ผู้ใช้สามารถเก็บบันทึกข้อมูลลงบนหน่วยความจำรอง เช่นจานแม่เหล็ก หรือเทปแม่เหล็ก ซึ่งอาจจัดเก็บในลักษณะของเพิ่มข้อมูลภาษา C มีฟังก์ชันจำนวนหนึ่งที่สามารถจัดการทำงานในลักษณะต่าง ๆ กับเพิ่มข้อมูล เช่น การอ่านข้อมูลจากเพิ่มข้อมูล การบันทึกข้อมูลในลักษณะบันทึกใหม่ทั้งหมด หรือบันทึกต่อจากข้อมูลเดิม เป็นต้น ซึ่งในที่นี้จะกล่าวถึงเฉพาะการจัดการกับเพิ่มข้อมูลในลักษณะของการอ่านหรือเขียนข้อมูลผ่านที่พักข้อมูล(buffer)เท่านั้น

13.1 เพิ่มข้อมูล

ในภาษา C เพิ่มข้อมูลหมายถึงโครงสร้างข้อมูลที่ประกอบด้วยกลุ่มของข้อมูลที่สัมพันธ์กันหรือที่เกี่ยวข้องกัน ซึ่งไม่เหมือนข้อมูลแถวลำดับที่มีการระบุจำนวนข้อมูลที่คงที่

เพิ่มข้อมูลหนึ่ง ๆ ประกอบด้วยกลุ่มของระเบียบ และภายในแต่ละระเบียบจะประกอบไปด้วยเขตข้อมูลย่อยที่มีประเภทข้อมูลต่างกัน และจะมีสัญลักษณ์เพื่อแสดงการจบเพิ่มข้อมูล <eof> (end of file) อยู่ต่อจากระเบียบสุดท้าย



รูปที่ 13.1 ภาพจำลองเพิ่มข้อมูล X ซึ่งประกอบด้วยระเบียบ n ตัว

ตัวอย่างของเพิ่มข้อมูล

ตัวอย่างที่ 13.1ก เพิ่มข้อมูลของจำนวนเต็มบวก ซึ่งแต่ละระเบียนประกอบด้วยหนึ่งเขตข้อมูลย่อยของจำนวนเต็มบวก

```
13 7 15 6 4 157 3 ... 92 <eof>
```

ข้อมูลในระเบียนที่หนึ่งคือ 13 ข้อมูลในระเบียนที่สองคือ 7 และตัวสุดท้ายคือ 92

ตัวอย่างที่ 13.1ข เพิ่มข้อมูลของตัวอักษร

```
'a' 'b' 'c' '!' '%' ... '$' <eof>
```

ข้อมูลในระเบียนที่หนึ่งคือ 'a' ข้อมูลในระเบียนที่สองคือ 'b' และตัวสุดท้ายคือ '\$'

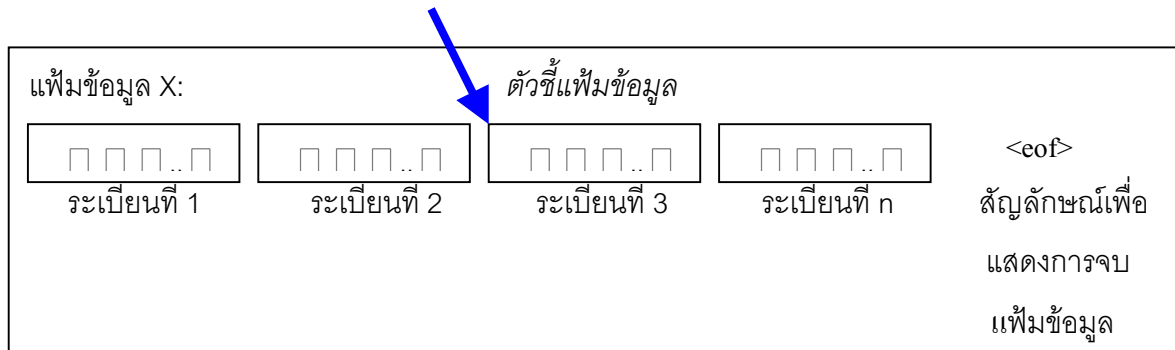
ตัวอย่างที่ 13.1ค รายการหนึ่ง ๆ ในสมุดโทรศัพท์ อาจกำหนดชนิดของระเบียน ชื่อPhoneNum ได้ดังนี้

```
typedef struct
{
    char name[25];
    char number[9];
} PhoneNum;
```

ในที่นี้เขตข้อมูล name เก็บชื่อ และเขตข้อมูล number เก็บหมายเลขโทรศัพท์

13.2 การทำงานกับเพิ่มข้อมูลของภาษา C

ภาษา C สามารถเข้าถึงเขตข้อมูลย่อยในแต่ละเพิ่มทั้งแบบเรียงลำดับและแบบโดยตรง ภาษา C จะมีตัวชี้เพิ่มข้อมูล (file pointer) ที่เป็นตัวระบุตำแหน่งของเขตข้อมูลย่อยที่ดำเนินการในปัจจุบัน



รูปที่ 13.2 ภาพจำลองเพิ่มข้อมูล X ซึ่งประกอบด้วยระเบียน n ตัว และตัวชี้เพิ่มข้อมูล แสดงตำแหน่งของระเบียนย่อยปัจจุบัน

เมื่อประกาศตัวแปร X ว่าเป็นเพิ่มข้อมูลและมีการเรียกใช้ฟังก์ชัน eof(X) ฟังก์ชันนี้จะคืนค่าเป็น 1 ถ้าตัวชี้เพิ่มข้อมูลอยู่ที่ตำแหน่งจบเพิ่มข้อมูล<eof> และจะคืนค่าเป็น 0 ถ้าตัวชี้เพิ่มข้อมูลอยู่ที่ตำแหน่งอื่น ๆ

ในการทำงานกับเพิ่มข้อมูลหนึ่ง จะต้องประกอบด้วยขั้นตอนทั่ว ๆ ไป ดังต่อไปนี้

- (1) เปิดเพิ่มข้อมูล (open file) เมื่อต้องการเริ่มต้นจะทำงานกับเพิ่มข้อมูล
- (2) บันทึกข้อมูลลงเพิ่มข้อมูล(write file) หรือ อ่านข้อมูลจากเพิ่มข้อมูล(read file)
- (3) ปิดเพิ่มข้อมูล (close file) เมื่อต้องการจะเลิกทำงานกับเพิ่มข้อมูล

13.3 การเปิดเพิ่มข้อมูล

การประกาศว่าจะทำการใช้เพิ่มข้อมูล แสดงโดยคำสั่งการเปิดเพิ่มข้อมูล(open file) ซึ่งจะแสดงการประกาศพร้อมกับการระบุตัวชี้ตำแหน่งข้อมูลในเพิ่ม ให้กับเพิ่มข้อมูลที่ต้องการเปิดใช้ ผู้ใช้เพียงแต่ประกาศตัวชี้ไปยังโครงสร้างของเพิ่มข้อมูลเท่านั้น และระบุชื่อของเพิ่มข้อมูล รวมถึงจุดประสงค์ของการใช้เพิ่มข้อมูลได้แก่การอ่าน(read) การบันทึก(write) การบันทึกต่อท้าย (append) กับข้อมูลในเพิ่มข้อมูลหนึ่ง ๆ

รูปแบบคำสั่งที่ใช้ในการเปิดเพิ่มข้อมูล

```
file *fp;
fp = fopen(filename,mode);
```

คำอธิบายคำสั่ง

file	เป็นตัวระบุที่กำหนดไว้ใน <stdio.h> ว่าเป็นข้อมูลประเภทเพิ่มข้อมูล
fp	เป็นการกำหนดตัวชี้เพิ่มข้อมูล
fopen	ระบุการเปิดเพิ่มข้อมูล ซึ่งมีอาร์กิวเมนต์สองตัว คือ
filename	เป็นชื่อของเพิ่มข้อมูลที่ต้องการเปิด
mode	จุดประสงค์ของการใช้งานกับเพิ่มข้อมูล ซึ่งมีหลายลักษณะดังนี้
r	สำหรับการเปิดเพิ่มข้อมูล เพื่อการอ่าน
w	สำหรับการเปิดเพิ่มข้อมูล เพื่อการบันทึก
a	สำหรับการเปิดเพิ่มข้อมูล เพื่อการบันทึกข้อมูลต่อจากข้อมูลสุดท้ายในเพิ่ม



ระบบปฏิบัติการจะทำการสร้างเพิ่มข้อมูลใหม่ หากมีการระบุชื่อเพิ่มข้อมูลที่ไม่ได้อยู่ก่อน
ในบางระบบปฏิบัติการจะแยกความแตกต่างระหว่างเพิ่มเก็บข้อความ และเพิ่มเก็บรหัสคำสั่ง
ซึ่งมักบันทึกเป็นแฟ้มทวิภาค(binary file) ในการระบุจุดประสงค์การใช้งานของแฟ้มทวิภาคจะต่อท้าย
ด้วย 'b' ดังนี้ 'rb' 'wb' 'ab'

ในการเปิดเพิ่มข้อมูลหนึ่ง ๆ อาจมีข้อผิดพลาดที่ทำให้ไม่สามารถเปิดเพิ่มข้อมูลได้สำเร็จ ทั้ง
นี้อาจเนื่องจาก

1. มีการระบุการป้องกันการบันทึกข้อมูลทับ(write protect)
2. เนื้อที่ในหน่วยความจำรองเต็ม
3. ไม่มีชื่อเพิ่มข้อมูลที่ระบุ

ในกรณีเหล่านี้ฟังก์ชัน fopen() จะให้ค่า null ถ้าไม่สามารถเปิดเพิ่มข้อมูลได้ แต่ถ้าเปิดเพิ่มข้อมูล
สำเร็จ จะคืนค่าตำแหน่งตัวชี้ไปที่ตำแหน่งระเบียบแรกของแฟ้มข้อมูล

ตัวอย่างที่ 13.2 แสดงการระบุการเปิดเพิ่มข้อมูล outfile1.txt เพื่อทำการบันทึกเพิ่มข้อมูล

```
...  
file *fp;  
fp = fopen("outfile1.txt", "w");  
...
```

ตัวอย่างที่ 13.3 แสดงการทดสอบการเปิดเพิ่มข้อมูล

```
....  
file *filename;  
filename = fopen("outfile1.txt", "r");  
if (filename == NULL)  
{  
    printf("File doesn't exist\n");  
}  
...
```

ถ้าไม่มีเพิ่มข้อมูลชื่อ outfile1.txt โปรแกรมจะแสดงข้อความ File doesn't exist ให้ผู้ใช้

ทราบ



เมื่อเริ่มทำงานกับโปรแกรมภาษา C ใด ๆ ระบบปฏิบัติการจะเปิดเพิ่มข้อมูลสามเพิ่มให้โดยอัตโนมัติ เพิ่มทั้งสามนี้คือเพิ่มข้อมูลนำเข้ามาตรฐาน (stdin) เพิ่มข้อมูลส่งออกมาตรฐาน (stdout) เพิ่มข้อมูลข้อผิดพลาดมาตรฐาน (stderr)

13.4 การปิดเพิ่มข้อมูล

หลังจากที่สิ้นสุดการใช้งานเพิ่มข้อมูลหนึ่ง ๆ แล้ว ผู้ใช้ควรที่จะทำการปิดเพิ่มข้อมูล(close file) เพื่อให้ระบบนำข้อมูลที่เหลือในบัฟเฟอร์ข้อมูลทั้งหมดเข้าไปบันทึกในเพิ่มข้อมูลเพื่อป้องกันข้อมูลสูญหาย

รูปแบบ คำสั่งที่ใช้ในการปิดเพิ่มข้อมูล

```
fclose(fp);
```

คำอธิบาย fp เป็นชื่อของเพิ่มข้อมูลที่ต้องการปิด

ถ้าการปิดเพิ่มข้อมูลด้วยฟังก์ชัน fclose สำเร็จจะคืนค่าเป็นศูนย์ (0) แต่ถ้าการปิดเพิ่มข้อมูลไม่สำเร็จจะคืนค่าอื่น

ตัวอย่างที่ 13.4 แสดงการเปิด และปิดเพิ่มข้อมูล outfile1.txt

```
...
FILE *filename;
filename = fopen("outfile1.txt", "r");
...
fclose(filename);
...
```

13.5 การอ่านและเขียนเพิ่มข้อมูล

การอ่านและเขียนเพิ่มข้อมูล สามารถทำได้หลายวิธี ดังนี้

13.5.1 อ่านและเขียนข้อมูลในระดับตัวอักษร

เป็นการประมวลผลทีละ 1 ตัวอักษร โดยใช้ฟังก์ชัน putc () และ getc () ทั้งนี้ฟังก์ชันทั้งสองจะมีพารามิเตอร์ตัวชี้เพิ่มข้อมูลเพื่อที่จะได้รู้ว่าจะทำงานกับเพิ่มข้อมูลใด

หน้าที่ของ getc คือการอ่านอักขระตัวต่อไปในสายอักขระภายในเพิ่มข้อมูลซึ่งถูกอ้างอิงด้วยตัวชี้เพิ่มข้อมูล จนพบ eof เมื่อจบเพิ่มข้อมูล และหน้าที่ของ putc คือการนำตัวอักขระในสายอักขระที่กำหนด บันทึกลงในเพิ่มข้อมูลซึ่งถูกอ้างอิงด้วยตัวชี้เพิ่มข้อมูล



รูปแบบ ฟังก์ชัน getc และ putc

```
ch = getc(fp);
putc (ch,fp);
```

คำอธิบาย

ch เป็นตัวแปรชนิดอักขระ

fp เป็นตัวชี้ตำแหน่งข้อมูลในแฟ้ม

ตัวอย่างที่ 13.5 โปรแกรมแสดงการนำข้อความที่กำหนดในตัวแปร contents บันทึกลงในแฟ้มข้อมูล outfile1.txt ซึ่งเป็นแฟ้มข้อมูลที่ประสงค์จะสร้างใหม่

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

void main()
{
file *anyname;
char contents[400];
int i;

strcpy(contents, "A builder has a new mobile phone which he carries with him
everywhere. One day, while eating his sandwiches at the top of some scaffolding, his
phone
rings. It's his friend, calling to tell him the horse he backed in the two o'clock at
Newmarket has won. \"Excellent!\" exclaims the builder,\"but how did you know where I
was?\" source : http://www.readersdigest.co.uk/mjokes/webjokes ");
anyname = fopen("outfile1.txt", "w"); /* open for appending */

for (i = 0 ; contents[i] ; i++)
{
putc(contents[i], anyname); /* output one character */
}
putc('\n', anyname); /* output a linefeed */
fclose(anyname);
}
```



ผลลัพธ์ที่ได้จากการใช้คำสั่ง type outfile1.txt ที่บรรทัดคำสั่ง

```
C:\TC\BIN>type outfile1.txt
A builder has a new mobile phone which he carries with him everywhere. One day,
while eating his sandwiches at the top of some scaffolding, his phone rings. It'
s his friend, calling to tell him the horse he backed in the two o'clock at Newn
arket has uon. "Excellent!" exclaims the builder, "but how did you know where I u
as?" source : http://www.readersdigest.co.uk/mjokes/webjokes
C:\TC\BIN>_
```

ตัวอย่างที่ 13.6 โปรแกรมแสดงการอ่านข้อความจากเพิ่มข้อมูล outfile1.txt ออกทางจอภาพ

```
#include <stdio.h>
void main()
{
    FILE *filename;
    char c;
    filename = fopen("outfile1.TXT", "r");
    if (filename == NULL)
    {
        printf("File doesn't exist\n");
    }
    else
    {
        do
        {
            c = getc(filename); /* get one character from the file */
            putchar(c);        /* display it on the monitor */
        }while (c != EOF);     /* repeat until EOF (end of file) */
    }
    fclose(filename);
}
```

ผลลัพธ์ที่ได้

```
C:\TC\BIN>tc
A builder has a new mobile phone which he carries with him everywhere. One day,
while eating his sandwiches at the top of some scaffolding, his phone rings. It'
s his friend, calling to tell him the horse he backed in the two o'clock at Newn
arket has uon. "Excellent!" exclaims the builder, "but how did you know where I u
as?" source : http://www.readersdigest.co.uk/mjokes/webjokes
```



ตัวอย่างที่ 13.7 โปรแกรมแสดงการอ่านข้อความจากเพิ่มข้อมูล outfile1.txt ระบุให้แสดงผลผ่านทาง

stdout

```
#include <stdio.h>

void main()
{
    int c;
    FILE *fp;
    fp = fopen("outfile1.txt", "r");
    if (fp == NULL)
        printf ("Can't find file ");
    else
        while (( c = getc(fp)) != EOF)
            putc (c,stdout);
    fclose(fp);
}
```

ผลลัพธ์ที่ได้

เหมือนตัวอย่างที่ 13.6

13.5.2 อ่านและเขียนข้อมูลเป็นคำ

ในการนำข้อมูลเข้าหรือออกที่ละคำนั้น ภาษา C มีฟังก์ชันที่ให้เรียกใช้ดังนี้

(1) ฟังก์ชัน fscanf และ fprintf

การทำงานของฟังก์ชัน fscanf และ fprintf นี้คล้ายกับ scanf และ printf กล่าวคือทำการแปลงค่า จัดรูปแบบ และพิมพ์อาร์กิวเมนต์ ไปยังอุปกรณ์นำข้อมูลเข้ามาตรฐาน และ อุปกรณ์ส่งออกมาตรฐาน ให้เป็นตัวอักขระซึ่งอ่านเข้าใจได้ และอยู่ในรูปแบบที่กำหนด

รูปแบบ ฟังก์ชัน fscanf และ fprintf

```
int fscanf (file *fp, " control ",arg-list);
int fprintf(file *fp, " control ",arg-list);
```



คำอธิบาย

fp	เป็นตัวชี้ตำแหน่งข้อมูลในแฟ้ม
control	เป็นรูปแบบของการบันทึกข้อมูลเหมือนกับรูปแบบที่ใช้ใน printf() หรือ scanf()
arg-list	เป็นรายการค่าคงที่ ตัวแปร หรือ นิพจน์ ที่จะบันทึกลงในแฟ้มข้อมูล ซึ่งจะ เป็นชนิดตัวเลขหรือตัวอักขระก็ได้

ฟังก์ชัน fprintf() จะทำการบันทึกข้อมูลลงในแฟ้มตามรูปแบบที่กำหนด ในขณะที่ fscanf() เป็นฟังก์ชันที่ใช้อ่านข้อมูลเข้ามาเหมือนกับ scanf() ต่างกันตรงที่ fscanf() จะอ่านข้อมูลจากแฟ้มข้อมูล และจะหยุดอ่านถ้าพบ eof

ตัวอย่างที่ 13.8 โปรแกรมแสดงการใช้ฟังก์ชัน fscanf อ่านข้อมูลจากแฟ้มข้อมูล outfile1.txt ที่ใช้ในตัวอย่างข้างต้น

```
#include <stdio.h>

void main()
{
    file *filename;
    char eachword[100];
    char c;

    filename = fopen("outfile1.txt", "r");

    do
    {
        c = fscanf(filename, "%s", eachword);
        printf("%s\n", eachword);           /*display*/
    } while (c != EOF);

    fclose(filename);
}
```



ผลลัพธ์ที่ได้ (ดัดมาเพียงส่วนปลายของผลลัพธ์)

```
backed
in
the
two
o'clock
at
Newmarket
has
won.
'Excellent!'
exclaims
the
builder, "but
how
did
you
know
where
I
was?"
source
:
http://www.readersdigest.co.uk/mjokes/webjokes
http://www.readersdigest.co.uk/mjokes/webjokes
```

จากตัวอย่างที่ 13.8 จะเห็นว่าข้อความสุดท้ายแสดงซ้ำกัน การแก้ไขแสดงในตัวอย่างต่อไป
ตัวอย่างที่ 13.9 ทำการแก้ไขโปรแกรมในตัวอย่างที่ 13.8 ให้แสดงผลลัพธ์ถูกต้อง โดยการตรวจสอบว่า
คำที่อ่านมาคือ EOF หรือไม่ ถ้าใช่จะออกจากวงวน while

```
#include "stdio.h"

void main()
{
    file *filename;
    char eachword[200];
    char c;

    filename = fopen("outfile1.txt", "r");

    do
    {
        c = fscanf(filename, "%s", eachword);
        if (c != EOF)
            printf("%s\n", eachword);          /* display */
    } while (c != EOF);

    fclose(filename);
}
```



ผลลัพธ์ที่ได้

```
he
backed
in
the
two
o'clock
at
Newmarket
has
won.
"Excellent!"
exclaims
the
builder, "but
how
did
you
know
where
I
was?"
source
:
http://www.readersdigest.co.uk/mjokes/webjokes
```

ตัวอย่างที่ 13.10 โปรแกรมแสดงการใช้ fprintf เพื่อสร้างเพิ่มข้อมูลจำนวนเต็มบวกคู่ จาก 2 ถึง 20 โดยให้ผู้ใช้สามารถป้อนชื่อเพิ่มข้อมูลที่จะสร้างได้

```
#include <stdio.h>
void main()
{
    file *fp;
    int i;
    char filename[25];

    printf("Enter filename : ");
    scanf("%s", filename);

    fp = fopen(filename, "w");      /* open for writing */

    for (i = 2 ; i <= 20 ; i=i+2)
        fprintf(fp, " %d ", i);

    fclose(fp);                    /* close the file */
}
```



ผลลัพธ์ที่ได้

```
C:\TC\BIN>tc
Enter filename :   intfile1.dat
-
```

ในที่นี้ผู้ใช้ระบุชื่อเพิ่มข้อมูลที่จะสร้างคือ intfile1.dat จากนั้นใช้คำสั่ง type ที่บรรทัดคำสั่งเพื่อดูเนื้อหาที่บรรจุภายในเพิ่มข้อมูล

```
C:\TC\BIN>type intfile1.dat
 2  4  6  8 10 12 14 16 18 20
C:\TC\BIN>
```

13.5.3 อ่านหรือเขียนข้อมูลเป็นบรรทัด

ในการนำข้อมูลเข้าหรือออกที่ละบรรทัด นั้น ภาษา C มีฟังก์ชันให้เรียกใช้คือ fgets กับ fputs

❖ ฟังก์ชัน fgets

ฟังก์ชัน fgets ทำหน้าที่อ่านข้อมูลจากแฟ้มที่ละบรรทัดรวมทั้ง “\n”

รูปแบบ

```
fgets (line, MAXLINE, fp)
```

ฟังก์ชัน fgets จะอ่านข้อมูลจากแฟ้มที่ถูกชี้โดยตัวชี้ fp ไปเก็บไว้ที่ตัวแปรแถวลำดับ line ซึ่งจำนวนอักขระที่จะอ่านในแต่ละบรรทัดไม่เกิน MAXLINE-1 เมื่ออ่านจนถึงจุดสิ้นสุดแฟ้มข้อมูล จะเติม ‘\0’ เป็นอักขระปิดท้ายในบรรทัดสุดท้าย การทำงานจะคืนค่า null เมื่อทำงานเสร็จสิ้น

❖ ฟังก์ชัน fputs

ฟังก์ชัน fputs ทำหน้าที่บันทึกข้อมูลจากสายอักขระ line (ซึ่งไม่จำเป็นต้องมี ‘\n’ เป็นตัวระบุการขึ้นบรรทัดใหม่)

รูปแบบ

```
fputs (line, fp)
```

ฟังก์ชัน fputs จะบันทึกข้อมูลลงในแฟ้มข้อมูลที่ถูกชี้โดยตัวชี้ fp ที่มีตัวชี้เพิ่มข้อมูลระบออยู่ การทำงานจะคืนค่า 1 ถ้าเกิดการทํางานผิดพลาด และจะคืนค่าเป็น 0 ในกรณีอื่นๆ



ตัวอย่างที่ 13.11 โปรแกรมแสดงการใช้ fgets อ่านข้อมูลจากเพิ่มข้อมูล data1.txt ให้แสดงบรรทัดละ 19 ตัวอักษร ทั้งนี้ข้อมูลภายในของ data1.txt มีข้อมูลดังนี้

```
C:\IC\BIN>type data1.txt
A builder has a new mobile phone which he carries with him everywhere.
One day, while eating his sandwiches at the top of some scaffolding,
his phone rings. It's his friend, calling to tell him the horse he backed
in the two o'clock at Newmarket has won. "Excellent!" exclaims the builder,
"but how did you know where I was?"
source : http://www.readersdigest.co.uk/njokes/webjokes
C:\IC\BIN>_
```

```
#include <stdio.h>

void main()
{
    file *fp1;
    char eachline[20];
    char *c;

    fp1 = fopen("data1.txt", "r");

    do
    {
        c = fgets(eachline, 20, fp1); /* get one line from the file */
        if (c != NULL)
            printf("\n%s", eachline); /* display it on the monitor */
    } while (c != NULL); /* repeat until NULL */

    fclose(fp1);
}
```



ผลลัพธ์ที่ได้

```

he carries with hi
m everywhere.

One day, while eati
ng his sandwiches a
t the top of some s
caffolding,

his phone rings. It
's his friend, call
ing to tell him the
horse he backed

in the two o'clock
at Newmarket has wo
n. "Excellent!" exc
lains the builder,

"but how did you kn
ow where I was?"

source : http://www
.readersdigest.co.u
k/njokes/webjokes

```

ตัวอย่างที่ 13.12 โปรแกรมแสดงการใช้ fputs บันทึกสายอักขระสามบรรทัดที่ป้อนโดยผู้ใช้งานจากแฟ้มเป็นอักขระ ลงในแฟ้มข้อมูล data1.txt ซึ่งเป็นการบันทึกต่อจากข้อมูลเดิม

```

#include<stdio.h>
void main( )
{
    file *fp;
    char anytext[20];
    int i;
    if((fp= fopen("data1.txt" , "a")) == NULL)
    {
        printf("cannot open file \n");
    }
    else {
        for(i=0;i<3;i++)
        {
            printf("\nEnter data line %d :",i+1);
            gets(anytext);
            fputs(anytext,fp);
        }
    }
    fclose(fp);
}

```



ผลลัพธ์ที่ได้

```
Enter data line 1 :INSERT DATA LINE 1
Enter data line 2 :INSERT DATA LINE 2
Enter data line 3 :INSERT DATA LINE 3
```

ข้อมูลภายในของแฟ้ม data1.txt หลังจากการรันโปรแกรมในตัวอย่างที่ 13.12 ที่แสดงด้วยคำสั่ง type มีลักษณะดังนี้

```
C:\IC\BIN>type data1.txt
A builder has a new mobile phone which he carries with him everywhere.
One day, while eating his sandwiches at the top of some scaffolding,
his phone rings. It's his friend, calling to tell him the horse he backed
in the two o'clock at Newmarket has won. "Excellent!" exclaims the builder,
"but how did you know where I was?"
source : http://www.readersdigest.co.uk/njokes/webjokesINSERT DATA LINE 1INSERT
DATA LINE 2INSERT DATA LINE 3
C:\IC\BIN>
```

ตัวอย่างที่ 13.13 โปรแกรมแสดงการสร้างแฟ้มข้อมูลชื่อ address.dat โดยให้แต่ละรายการประกอบด้วย ชื่อ (ความยาวสูงสุดคือ 20 ตัวอักษร) และหมายเลขโทรศัพท์ (ความยาวสูงสุดคือ 15 ตัวอักษร) โดยผู้ใช้ป้อนข้อมูลทั้งหมดจากแผงแป้นอักขระบันทึกที่รายการหนึ่ง ๆ จะมีการถามความมั่นใจจากผู้ใช้ก่อนที่ จะบันทึก

ถ้าตอบ 'n' หรือ 'N' (ในที่นี้หมายถึงต้องการแก้ไขข้อมูลอีก) ข้อมูลรายการดังกล่าวจะไม่ บันทึกที่รายการนั้น ๆ ลงในแฟ้มข้อมูล

ถ้าตอบ 'y' หรือ 'Y' แสดงว่ารายการนั้นๆ จะถูกบันทึกลงในแฟ้มข้อมูล โปรแกรมจะถามต่อ ว่าต้องการทำรายการใหม่เพิ่มอีกหรือไม่

ถ้าตอบ 'y' หรือ 'Y' (ในที่นี้หมายถึงต้องการบันทึกข้อมูลอีก) โปรแกรมจะแสดงข้อความให้ทำงานต่อในลักษณะเดียวกับข้างต้น แต่ถ้าตอบ 'n' หรือ 'N' แสดงว่าต้องการหยุดการทำงาน



```
#include <stdio.h>

char Response1,Response2;
int Flag1,Flag2 ;

void main()
{

    file *fp1;
    char Name[20],Number[15];

    fp1 = fopen("address.dat", "w");
    Flag2 = 1;
    while (Flag2)
    {
        Flag1 = 1;
        while (Flag1)
        {
            printf ("\t \nInput phone directory entry.");
            printf ("\nName (max 20 characters).      : ");
            fgets(Name,20,stdin);
            printf ("\nNumber (format xx-xxx-xxxx).    : ");
            fgets(Number,15,stdin);
            printf ("\nDo you want to save change? (Y/N) : ");
            Response1 = getche(); /* display monitor ทีละ character */
            if((Response1 == 'Y') || (Response1 == 'y'))
                Flag1 = 0;
        }
        fputs(Name,fp1);
        fputs(Number,fp1);

        printf      ("\nAnother Entry (Y/N)?          : ");
        Response2 = getche();
        if((Response2 == 'N') || (Response2 == 'n'))
            Flag2 = 0;
    }
    fclose(fp1);
}
```



ผลลัพธ์ที่ได้

```

Input phone directory entry.
Name (max 20 characters).      : SOMSAK
Number (format xx-xxx-xxxx).   : 01-849-2552
                                Do you want to save change? (Y/N) : y
                                Another Entry (Y/N)?              : y
Input phone directory entry.
Name (max 20 characters).      : SOMSRI
Number (format xx-xxx-xxxx).   : 01-849-2553
                                Do you want to save change? (Y/N) : y
                                Another Entry (Y/N)?              :
Input phone directory entry.
Name (max 20 characters).      :
Number (format xx-xxx-xxxx).   :
                                Do you want to save change? (Y/N) : y
                                Another Entry (Y/N)?              : n
Input phone directory entry.
Name (max 20 characters).      : SOMSSAK
Number (format xx xxx xxxx).   : 01 849 2552
                                Do you want to save change? (Y/N) : y
                                Another Entry (Y/N)?              : y
Input phone directory entry.
Name (max 20 characters).      : SOMSRI
Number (format xx-xxx-xxxx).   : 01-849-2553
                                Do you want to save change? (Y/N) : n
Input phone directory entry.
Name (max 20 characters).      : SOMSRI
Number (format xx-xxx-xxxx).   : 01-849-2558
                                Do you want to save change? (Y/N) : y
                                Another Entry (Y/N)?              : n

```

ผลลัพธ์ที่ได้ จากการใช้คำสั่ง type ที่บรรทัดคำสั่ง จากข้อมูลภายในแฟ้ม address.dat หลังจากการรันโปรแกรม ในตัวอย่างที่ 13.13

```

C:\TC\BIN>type address.dat
SOMSAK
01-849-2552
SOMSRI
01-849-2553
C:\TC\BIN>

```



13.5.4 การอ่านและเขียนข้อมูลแบบเข้าถึงตรง

ในการอ่านและบันทึกข้อมูลจากหน่วยความจำไปเก็บไว้ในเพิ่มข้อมูลในตำแหน่งที่กำหนด นั้น ภาษา C มีฟังก์ชันให้จำนวนหนึ่ง แต่ในที่นี้จะกล่าวถึงเฉพาะ `fseek()` `fread()` และ `fwrite()` โดยจะแบ่งออกเป็น

❖ ฟังก์ชันเกี่ยวกับตำแหน่งในเพิ่มข้อมูล

`fseek` ทำหน้าที่กำหนดตำแหน่งในเพิ่มข้อมูลสำหรับกระแส(ข้อมูล) (stream) การอ่านหรือเขียนข้อมูลครั้งต่อไปจะเกิดขึ้น ณ ตำแหน่งนี้

รูปแบบ `fseek()`

```
fseek (fp, offset, origin);
```

คำอธิบาย

`fp` เป็นตัวชี้ตำแหน่งข้อมูล

`offset` สำหรับเพิ่มข้อมูลเลขฐานสอง ตำแหน่งจะเป็นระยะห่าง `offset` ตัวอักษร จากตำแหน่งที่กำหนดโดยตัวแปร `origin`

`origin` `offset` `mode` เป็นตัวเลขที่บ่งถึงสถานะของ `offset` ซึ่งอาจมีค่า `offset` `mode` ดังนี้

0 ตำแหน่งเริ่มต้นเพิ่มข้อมูล(`seek_set`)

1 ตำแหน่งปัจจุบัน(`seek_cur`)

2 ตำแหน่งท้ายเพิ่มข้อมูล(`seek_end`)

❖ ฟังก์ชันนำข้อมูลเข้าและออกโดยตรงได้แก่

`fread` อ่านวัตถุจากกระแส(ข้อมูล)เข้ามาเก็บในเพิ่มข้อมูล และ `fwrite` เขียนวัตถุจากเพิ่มข้อมูลออกไปยังกระแส(ข้อมูล)

รูปแบบ `fread()`

```
fread(ptr,size,num,fp);
```



คำอธิบาย

- ptr เป็นตำแหน่งของข้อมูลภายในหน่วยความจำที่ต้องการอ่านข้อมูลจากเพิ่มข้อมูล
- size เป็นขนาดของข้อมูลที่จะทำการอ่านจากเพิ่มข้อมูล
ว่าเป็นขนาดกี่ไบต์
- num ใช้แสดงจำนวนรายการข้อมูลที่ต้องการที่จะอ่าน
- fp เป็นตัวชี้ตำแหน่งข้อมูลนำเข้า

รูปแบบ fwrite()

```
fwrite(ptr,size,num,fp);
```

คำอธิบาย

- ptr เป็นตำแหน่งของข้อมูลภายในหน่วยความจำที่ต้องการนำข้อมูลไปบันทึกลงในเพิ่มข้อมูล
- size เป็นขนาดของข้อมูลที่จะทำการบันทึกลงในเพิ่มข้อมูล
ว่าเป็นขนาดกี่ไบต์
- num ใช้แสดงจำนวนรายการข้อมูลที่ต้องการที่จะบันทึก
- fp เป็นตัวชี้ตำแหน่งข้อมูลผลลัพธ์



ตัวอย่างที่ 13.14 โปรแกรมแสดงการใช้ fwrite() fseek() และ fread()

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

int main(void)
{
    FILE *stream;
    char msg[] = "TEST STATEMENT";
    char buf[30];
    long seek_set;
    seek_set = 0; /* seeks from the beginning of the file */

    if ((stream = fopen("test.dat", "w+")) == NULL)
    {
        fprintf(stderr, "Cannot open output file.\n");
        return 1;
    }

    /* write some data to the file */
    fwrite(msg, strlen(msg)+1, 1, stream); /* strlen(s) = return lengths of s */

    /* seek to the beginning of the file */
    fseek(stream, seek_set, 0);

    /* read the data and display it */
    fread(buf, strlen(msg)+1, 1, stream);
    printf("%s\n", buf);

    fclose(stream);
    return 0;
}
```

ผลลัพธ์ที่ได้

```
C:\>cd tc\bin
C:\TC\BIN>tc
TEST STATEMENT
_
```



13.6 การแสดงผลจากเพิ่มข้อมูลบนเครื่องพิมพ์

อุปกรณ์นำข้อมูลเข้าหรือออกบางประเภท จะเปิดไว้โดยอัตโนมัติเมื่อเริ่มทำงานกับภาษา C เช่น แปงแป้นอักขระ กำหนดโดย stdin หรือ จอภาพ กำหนดโดย stdout แต่อุปกรณ์บางตัวเช่นเครื่องพิมพ์ ผู้ใช้ต้องทำการกำหนดเองหากต้องการใช้โดยเปิดเป็นเพิ่มข้อมูลเพิ่มอีกหนึ่งตัว

รูปแบบ

```
file *fp;
fp = fopen("PRN", "w");
```

ตัวอย่างที่ 13.16 โปรแกรมแสดงการพิมพ์ข้อมูลออกทางเครื่องพิมพ์

```
#include "stdio.h"

void main()
{
    file *anyname, *printer;
    char c;

    anyname = fopen("outfile1.txt", "r"); /* open input file */
    printer = fopen("PRN", "w"); /* open printer file */

    do
    {
        c = getc(anyname); /* get one character from the file */
        if (c != EOF)
        {
            putchar(c); /* display it on the monitor */
            putc(c, printer); /* print the character */
        }
    } while (c != EOF); /* repeat until EOF (end of file) */
    fclose(anyname);
    fclose(printer);
}
```

ผลลัพธ์ที่ได้

พิมพ์ผลทางเครื่องพิมพ์



แบบฝึกหัด

1. จงบอกข้อแตกต่างระหว่าง text file และ binary file
2. จงเขียนโปรแกรมรับชื่อเพิ่มข้อมูลจากผู้ใช้ และพิมพ์ข้อมูลภายในเพิ่มข้อมูลดังกล่าวที่ละบรรทัด ระบุหมายเลขบรรทัดที่พิมพ์
3. ให้เขียนโปรแกรมที่ผู้ใช้สามารถป้อนชื่อเพิ่มข้อมูลนำเข้า และเพิ่มข้อมูลผลลัพธ์ ให้เปิดเพิ่มข้อมูลทั้งสองพร้อม ๆ กัน จากนั้นให้อ่านข้อมูลจากเพิ่มข้อมูลนำเข้า และบันทึกลงในเพิ่มข้อมูลผลลัพธ์ ทั้งนี้ให้แสดงผลลัพธ์บนจอภาพ และพิมพ์ผลลัพธ์ทางเครื่องพิมพ์ด้วย
4. ให้เขียนโปรแกรมแสดงการแปลงองศาเซลเซียสเป็นองศาฟาเรนไฮด์ โดยกำหนดอุณหภูมิเริ่มต้นเป็น 32 องศาเซลเซียส จนถึง 110 องศาเซลเซียส การเปลี่ยนระดับอุณหภูมิให้เพิ่มขึ้นทีละสิบ กำหนดให้บันทึกผลลัพธ์เก็บลงในเพิ่มข้อมูลชื่อ degreeout
5. ให้เขียนโปรแกรมเพื่อทำสำเนาเพิ่มข้อมูล degreeout จากข้อที่ 3 ให้เป็นเพิ่มข้อมูลใหม่ชื่อ degeeco ซึ่งหลังจากการทำสำเนาสำเร็จให้ขึ้นข้อความระบุว่างานสำเร็จ คล้ายตัวอย่างภาพผลลัพธ์นี้

```
C:\TC\BIN>tc
Enter source file name : degreeou
Enter target file name : degeeco
Success! _
```

6. ให้สร้างเพิ่มข้อมูลชื่อ filein.txt รับข้อมูลชื่อ อายุ และเงินเดือนจากผู้ใช้ผ่านวงเป็นอักขระ ทั้งนี้จำนวนรายการข้อมูลจะกำหนดโดยผู้ใช้ และให้แสดงเนื้อหาของข้อมูลในเพิ่มที่พิมพ์ออกทางจอภาพ

.....#####.....

