

# Today

1. 지난 시간 복습 – 전처리문과 구조체
2. 메모리 할당하기
3. 파일입출력
4. 실습

# 표준 입출력

## ■ 표준 입출력

- 키보드로 입력
- 화면(모니터로) 출력

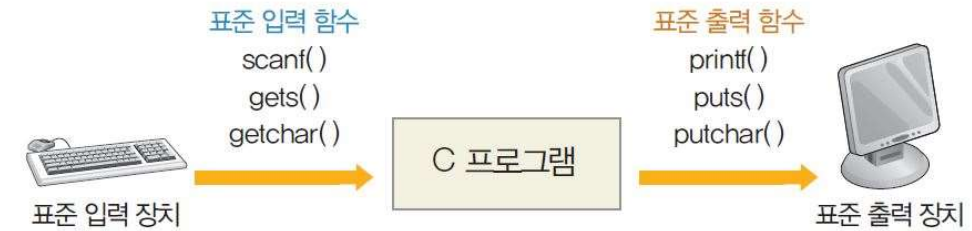


그림 11-1 표준 입출력의 개념

## ■ 서식화된 입출력

- printf( ), scanf( ): 모든 형식의 데이터 입출력 가능

구문	설명
printf("서식", 출력할 매개변수들 ...)	표준 출력 장치(모니터)에 서식을 맞춰 출력한다.
scanf("서식", 입력할 매개변수들 ...)	표준 입력 장치(키보드)에서 서식에 맞춰 입력받는다.

## ■ 문자열 입출력

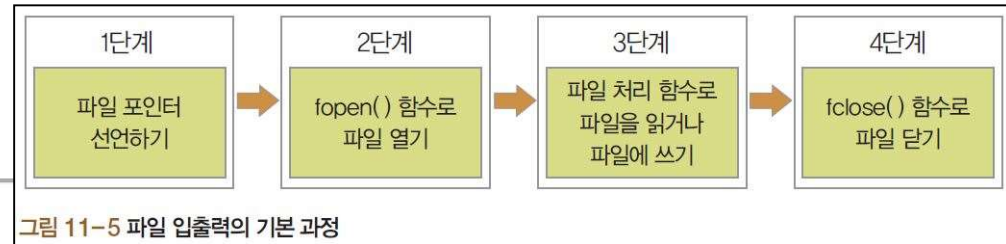
- puts( ), gets( ): 문자열만 입출력

# 파일 입출력

## ■ 파일 입출력 기본 과정

- 1단계: 파일 포인터 선언

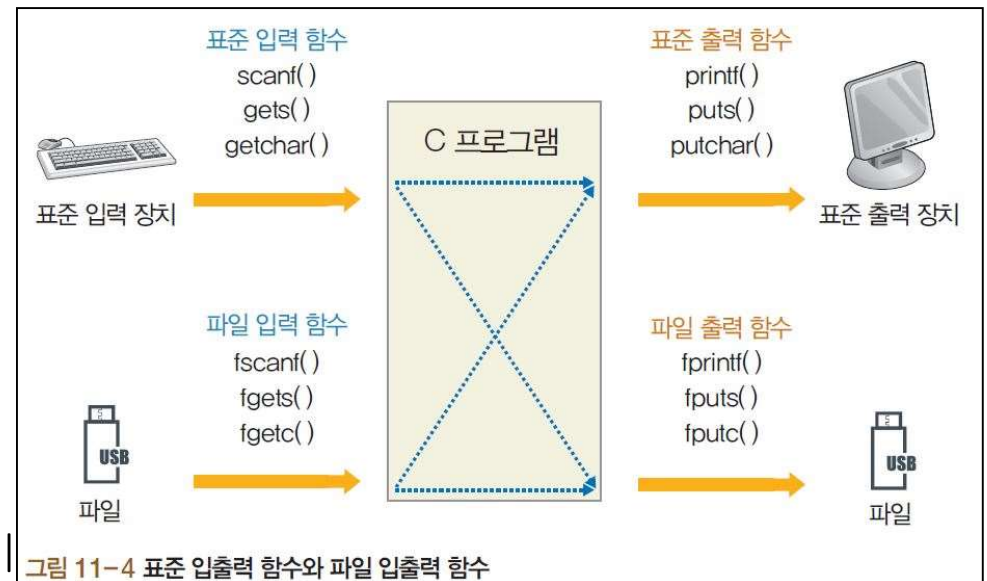
```
FILE *변수 이름;
```



- 2단계: fopen 함수로 파일 열기

```
변수 이름 = fopen("파일 이름", "열기 모드");
```

- 3단계 : 파일 처리 함수로 파일을 읽거나 파일에 쓰기
  - fgetc() 파일의 문자열 읽기



- 4단계 : fclose() 함수로 파일 닫기

```
fclose(파일 포인터);
```

# 파일의 문자열 읽기

## ■ fgets()

- 파일로부터 값을 입력받을 때 사용.
- 파일 포인터에 지정된 파일에서 문자열을 읽어 문자 배열에 대입함.
- 읽어올 문자열의 최대 길이는 '읽을 최대 문자수 -1'

```
fgets(문자 배열, 읽을 최대 문자 수, 파일 포인터);
```

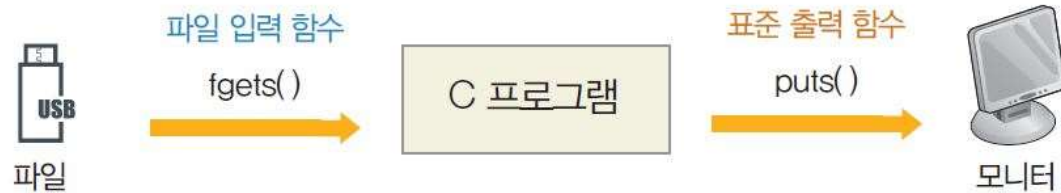


그림 11-6 파일 입력과 표준 출력

# fgets() 사용해보기

```
#include <stdio.h>

void main()
{
    char s[20];
    FILE *rfp;

    rfp = fopen("c:\\temp\\data1.txt", "r");

    fgets(s, 20, rfp);

    printf("파일에서 읽은 문자열 : ");
    puts(s);

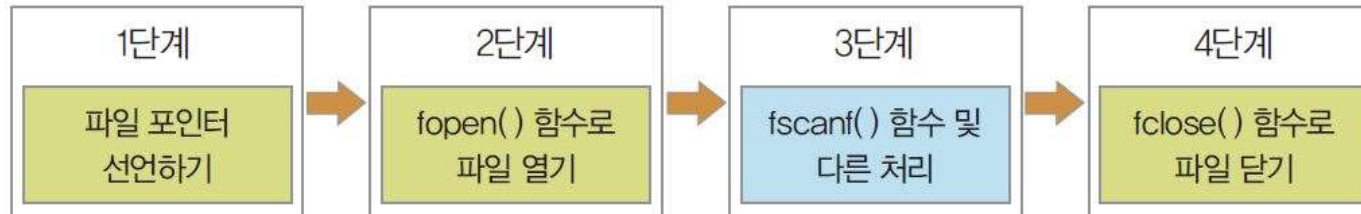
    fclose(rfp);
}
```

# 서식있는 파일 읽기

## ■ fscanf()

- 파일 포인터를 사용하는 것을 제외하고 scanf( )와 사용법이 동일

```
fscanf(파일 포인터, "서식", 입력할 매개변수들 ...);
```



# fscanf() 사용해 보기

```
#include <stdio.h>

void main()
{
    FILE *rfp;
    int sum = 0;
    int in, i;

    rfp = fopen("c:\\temp\\data2.txt", "r");

    for (i = 0; i < 5; i++)
    {
        fscanf(rfp, "%d", &in);
        sum += in;
    }

    printf("합계 = = > %d\n", sum);

    fclose(rfp);
}
```

# 파일에 문자열 쓰기

## ■ fputs()

- 파일에 문자열 쓰기

fputs(출력할 데이터, 파일 포인터);

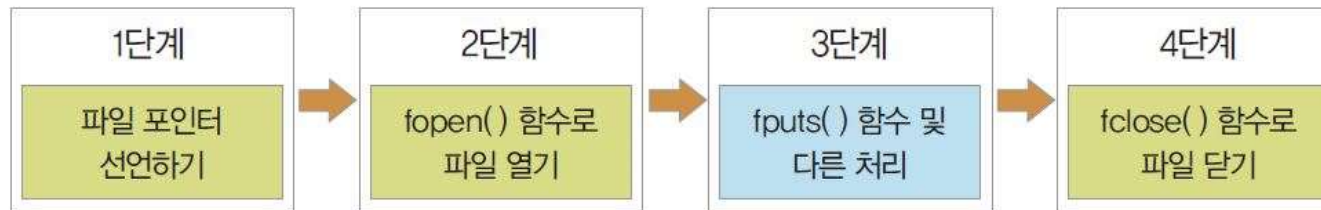


그림 11-7 표준 입력과 파일 출력



# fputs() 사용해보기

```
#include <stdio.h>

void main() {
    char s[20];
    FILE *wfp;

    wfp = fopen("c:\\temp\\data3.txt", "w");

    printf("문자열을 입력(최대 19자) : ");
    gets(s);

    fputs(s, wfp);

    fclose(wfp);
}
```

# 서식을 지정해서 파일에 쓰기

## ■ fprintf()

- 파일에 숫자를 출력할 때는 서식을 지정할 수 있는 fprintf( ) 함수를 사용하는 것이 편리함

fprintf(파일 포인터, "서식", 출력할 매개변수들 ...);

```
#include <stdio.h>

void main()
{
    FILE *wfp;
    int hap = 0;
    int in, i;

    wfp = fopen("c:\\temp\\data7.txt", "w");

    for (i = 0; i < 5; i++)
    {
        printf(" 숫자 %d : ", i + 1);
        scanf("%d", &in);
        hap = hap + in;
    }

    fprintf(wfp, "합계 = = > : %d\n", hap);

    fclose(wfp);
}
```

# Today

1. 지난 시간 복습 – 전처리문과 구조체
2. 메모리 할당하기
3. 파일입출력
4. 실습

## [실습2]

- ▶ **int**형 배열에 0에서 999까지 **random number**를 100개 생성하여 저장한 후, 100단위별로 생성된 숫자들의 빈도수를 출력해 보세요.
- ▶ **Hint.** 앞의 예제를 확장하여 빈도수를 저장할 크기가 10인 **int**형 배열 **bb**를 생성하여 모두 0으로 초기화한다. 다음으로 **for**문을 사용하여 100개의 숫자들을 하나씩 반복하면서 각각 10개의 구간에 속하는 지를 체크하여, 속하는 구간의 빈도수를 1씩 증가시킨다.

0 - 99:	10개
100 - 199:	14개
200 - 299:	8개
300 - 399:	7개
400 - 499:	9개
500 - 599:	12개
600 - 699:	9개
700 - 799:	8개
800 - 899:	12개
900 - 999:	11개

[실행 예]

# [실습] 배열 내 모든 원소의 합 구하기

- 개수를 나타내는 변수 N 을 전처리기로 정의하고 초기값은 10으로 설정하기
- int 형 배열 N개 원소를 담도록 생성

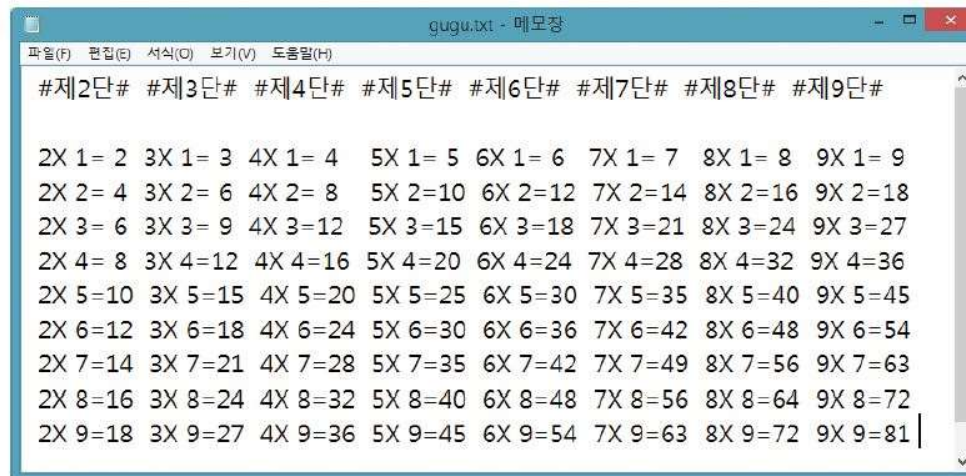
=====다음의 1~4는 모두 같은 결과를 나타냄=====

1. 반복문 안에서 배열을 사용해서 원소 N개 전체 합계를 계산하고 출력
2. 반복문 안에서 포인터를 사용해서 원소 전체 합계를 계산하고 출력
3. 합계계산하는 부분을 함수로 만들어서 계산하고 출력하기(함수명: sumTotal, 함수반환값: 정수형, 매개변수:없음)
4. 합계계산하는 부분을 함수로 만들어서 계산하고 출력하기(함수명: sumTotal, 함수반환값: 정수형, 매개변수:없음)

# [실습] 구구단을 파일에 출력

**예제 설명** [예제모음 14]의 내용을 모니터가 아닌 'gugu.txt' 파일에 쓰는 프로그램이다.

**실행 결과**

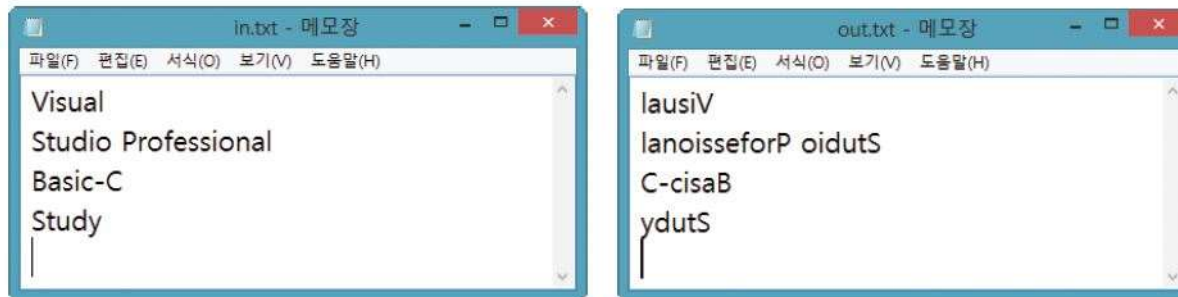


```
#제2단# #제3단# #제4단# #제5단# #제6단# #제7단# #제8단# #제9단#  
2X 1= 2 3X 1= 3 4X 1= 4 5X 1= 5 6X 1= 6 7X 1= 7 8X 1= 8 9X 1= 9  
2X 2= 4 3X 2= 6 4X 2= 8 5X 2=10 6X 2=12 7X 2=14 8X 2=16 9X 2=18  
2X 3= 6 3X 3= 9 4X 3=12 5X 3=15 6X 3=18 7X 3=21 8X 3=24 9X 3=27  
2X 4= 8 3X 4=12 4X 4=16 5X 4=20 6X 4=24 7X 4=28 8X 4=32 9X 4=36  
2X 5=10 3X 5=15 4X 5=20 5X 5=25 6X 5=30 7X 5=35 8X 5=40 9X 5=45  
2X 6=12 3X 6=18 4X 6=24 5X 6=30 6X 6=36 7X 6=42 8X 6=48 9X 6=54  
2X 7=14 3X 7=21 4X 7=28 5X 7=35 6X 7=42 7X 7=49 8X 7=56 9X 7=63  
2X 8=16 3X 8=24 4X 8=32 5X 8=40 6X 8=48 7X 8=56 8X 8=64 9X 8=72  
2X 9=18 3X 9=27 4X 9=36 5X 9=45 6X 9=54 7X 9=63 8X 9=72 9X 9=81
```

# [실습] 파일에서 읽어온 문자열 거꾸로 쓰기

**예제 설명** 미리 만들어둔 'in.txt' 파일의 내용을 읽어와 'out.txt' 파일에 쓰는데 각 행의 문자를 반대 순서로 저장하는 프로그램이다(반드시 'in.txt' 파일의 마지막 행에서 **Enter**를 누르고 저장한다).

**실행 결과**



The image shows two Notepad++ windows side-by-side. The left window, titled 'in.txt - 메모장', contains the text: Visual, Studio Professional, Basic-C, Study, followed by a blank line. The right window, titled 'out.txt - 메모장', contains the text: lausiV, lanoisseforP oidutS, C-cisaB, ydutS, followed by a blank line. This demonstrates the reversal of each line from the input file into the output file.

```
in.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
Visual
Studio Professional
Basic-C
Study
|

out.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
lausiv
lanoisseforP oidutS
C-cisaB
ydutS
|
```