

프로그래밍기초(계절)

수업 참여(오전+오후) 필수

오전: 강의

오후: 실습

실습과제 수행

제출 17:30까지 (최종 마감 23:59)

평가방법

중간 40%, **기말 45%**, 과제 10%, 출석 : 5%

기말시험

7/16(수) 13:30~17:00 필기(50분), 실기(150분, openbook)

후반부 수업 일정

1_15. 파일 처리 - 텍스트파일 (I)

2_11. 포인터 기초 (I)

3_11. 포인터 기초 (II)

4_16. 동적할당 (I)

5_12. 문자와 문자열

6_13. 구조체와 공용체

7_14. 함수와 포인터 활용

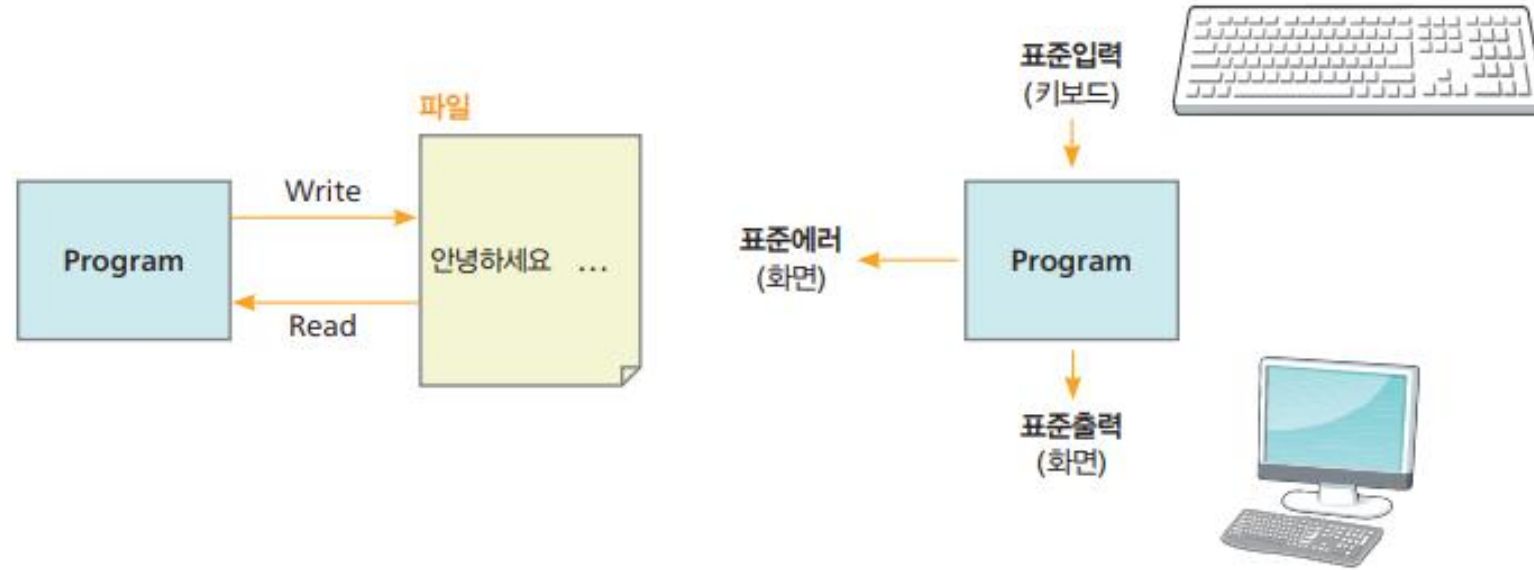
8_15. 파일 처리 (II) - 이진파일

9_16. 동적할당 (II) - 연결리스트

제 15 장 파일 처리 (I)

- 01 파일 기초
- 02 텍스트 파일 입출력

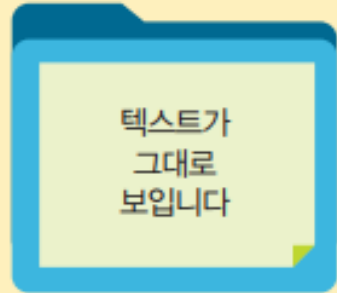
파일의 필요성



파일의 유형

텍스트 파일

이 텍스트 파일은 메모장과 같은 텍스트 편집기를 사용해 그 내용을 볼 수 있으며 필요하면 편집도 할 수 있다.



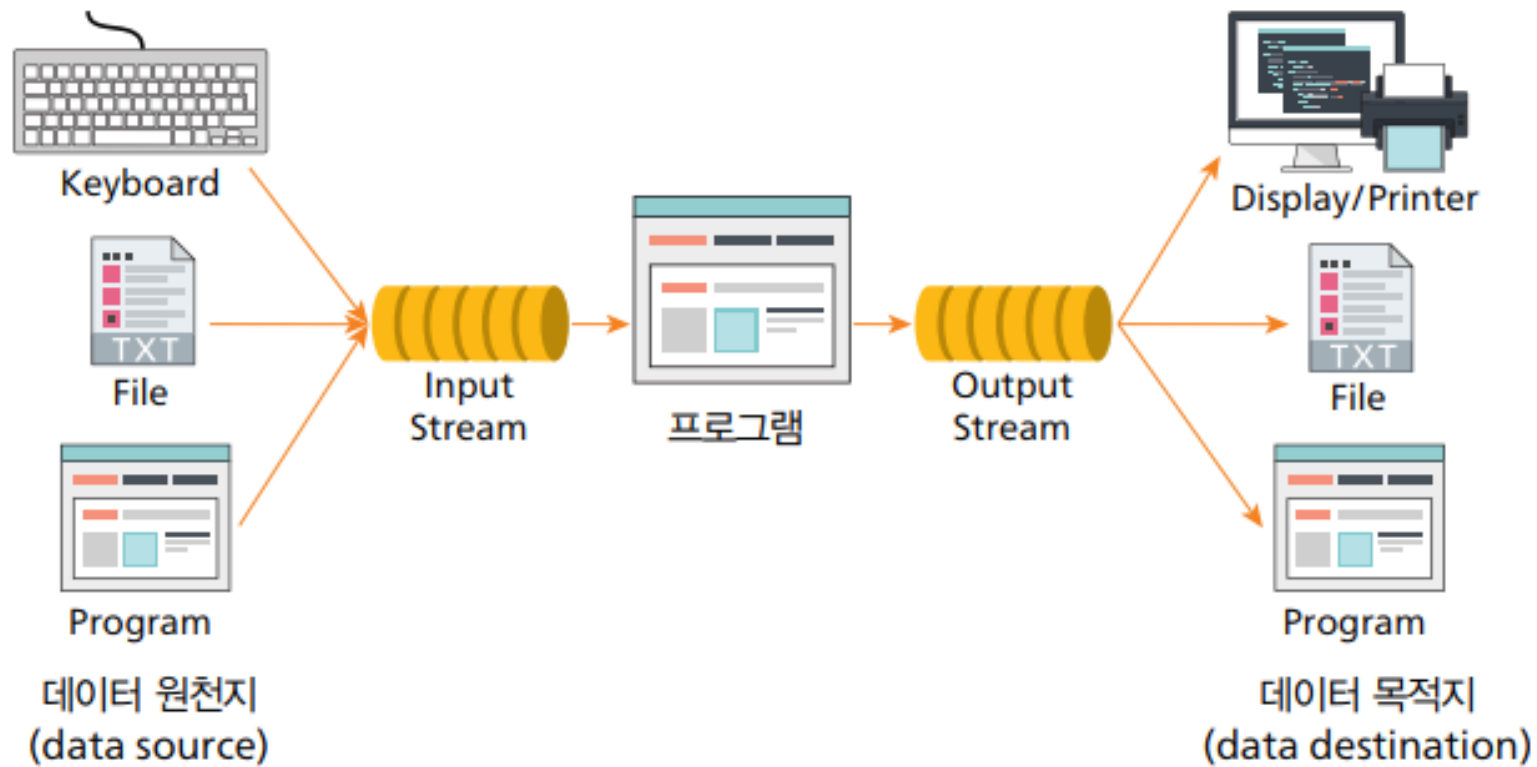
이진 파일

구조체의 변수 등 주기억장치의 내용을 그대로 저장



이미지 파일, 동영상 파일, 실행 파일과 같이 데이터로 구성된 파일로 일반 에디터로는 그 내용을 볼 수 없다.

입출력 스트림



- 파일 처리의 주요 단계

- 1. 파일 열기
- 2. 파일 작업 (읽기/쓰기)
- 3. 파일 닫기

- 단계별 작업

- `<stdio.h>` 헤더 파일에 정의된 함수들을 사용

파일(스트림) 열기 : 함수 fopen()

- 프로그램에서 특정한 파일과 파일 스트림을 연결

함수 fopen()과 fopen_s() 함수원형

```
FILE * fopen(const char * _Filename, const char * _Mode);  
errno_t fopen_s(FILE ** _File, const char * _Filename, const char * _Mode);
```

- 함수 fopen()은 파일명 _Filename의 파일 스트림을 모드 _Mode로 연결하는 함수이며, 스트림 연결에 성공하면 파일 포인터를 반환하며, 실패하면 NULL을 반환한다.
- 함수 fopen_s()는 스트림 연결에 성공하면 첫 번째 인자인 _File에 파일 포인터가 저장되고 정수 0을 반환하며, 실패하면 양수를 반환한다. 현재 Visual C++에서는 함수 fopen_s()의 사용을 권장하고 있다.

함수 fopen

```
FILE* f; //파일 포인터
char* fname = "basic.txt"; //파일이름

if ((f = fopen(fname, "w")) == NULL)
{
    printf("파일이 열리지 않습니다.\n");
    exit(1);
};
```

모드	의미	파일 존재 시	파일 없을 시
"r"	읽기 전용	열기 성공	오류 발생
"w"	쓰기 전용	내용 삭제	새로 생성
"a"	이어쓰기	끝에 추가	새로 생성
"r+"	읽기 + 쓰기	열기 성공	오류 발생
"w+"	읽기 + 쓰기	내용 삭제	새로 생성
"a+"	읽기 + 이어쓰기	끝에 추가	새로 생성

함수 fclose()로 파일 스트림 닫기

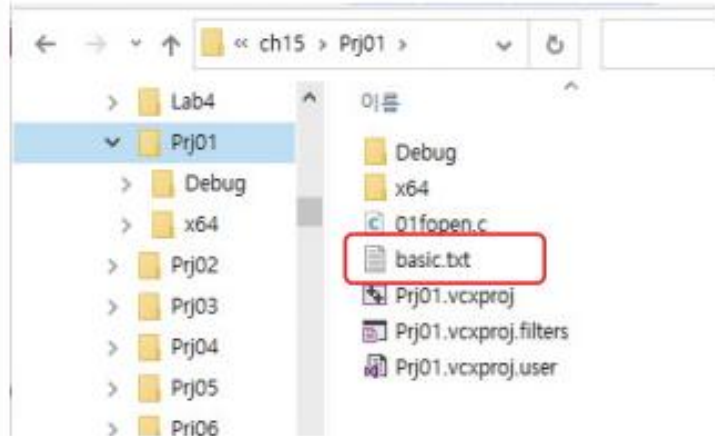
함수 fclose()

```
int fclose(FILE * _File);
```

함수 fclose()는 파일 스트림 f를 닫는 함수로서, 성공하면 0을 실패하면 EOF을 반환한다.

```
fclose(f);
```

예제: 파일 출력



파일처리 작업:

1. 파일 열기
2. 파일 작업
3. 파일 닫기

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

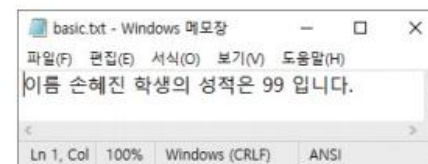
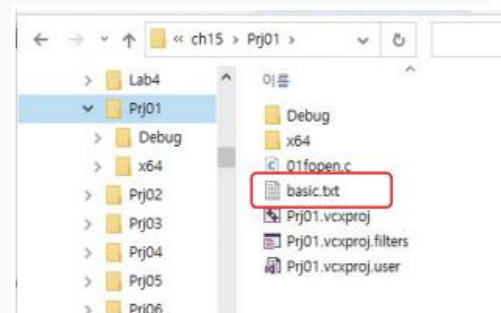
int main()
{
    char* fname = "basic.txt";
    FILE* f;
    char name[30] = "손혜진";
    int point = 99;

    if ((f = fopen(fname, "w")) == NULL) {
        printf("파일이 열리지 않아 종료합니다.\n");
        exit(1);
    }

    fprintf(f, "이름 %s 학생의 성적은 %d 입니다.\n", name, point);
    fclose(f);

    printf("이름 %s 학생의 성적은 %d 입니다.\n", name, point);
    puts("프로젝트 폴더에서 파일 basic.txt를 메모장으로 열어 보세요.");

    return 0;
}
```

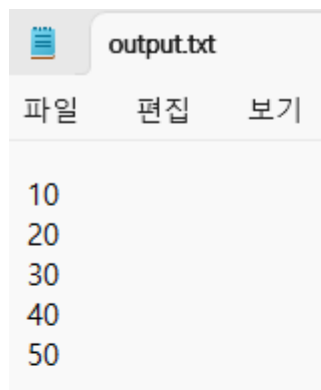
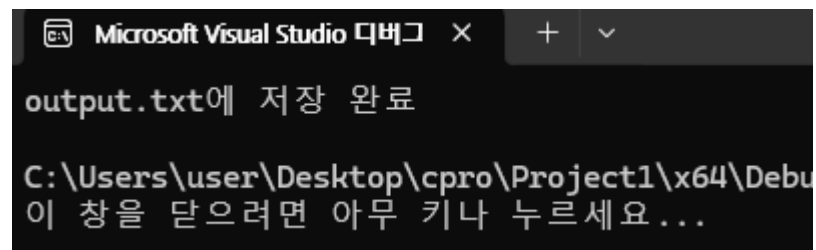


```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <stdlib.h> //for exit()
04
05 int main()
06 {
07     FILE* f;
08
09     if ((f = fopen("myinfo.txt", "w")) == NULL)
10     {
11         printf("파일이 열리지 않습니다.\n");
12         exit(1);
13     }
14
15     char tel[15] = "010-3018-3824";
16     char add[30] = "서초구 대검로 557";
17     int age = 22;
18
19     fprintf(f, "전화번호: %s, 주소:%s, 나이: %d\n", tel, add, age);
20
21     fclose(f);
22
23     printf("전화번호: %s, 주소:%s, 나이: %d\n", tel, add, age);
24     puts("프로젝트 폴더에서 파일 myinfo.txt를 메모장으로 열어 보세요.");
25
26     return 0;
27
28 }
29
```

다음과 같은 정수 배열이 있습니다:

```
int numbers[5] = { 10, 20, 30, 40, 50 };
```

이 배열의 모든 값을 `output.txt` 파일에 한 줄에 하나씩 저장하는 프로그램을 작성하세요.



파일 처리

- 01 파일 기초
- **02 텍스트 파일 입출력**
- 03 이진 파일 입출력
- 04 파일 접근 처리

**** 텍스트 파일 처리: 주요 함수 ****

파일 열기/닫기

fopen() // 파일 열기
fclose() // 파일 닫기

파일 읽기

fgetc() // 한 문자 읽기
fgets() // 한 줄 읽기
fscanf() // 형식화된 데이터 읽기

파일 쓰기

fputc() // 한 문자 쓰기
fputs() // 문자열 쓰기
fprintf() // 형식화된 데이터 쓰기

파일 상태 확인

feof() // 파일 끝(EOF) 여부 확인
ferror() // 파일 스트림 오류 여부 확인

함수 fprintf()와 fscanf()

- 텍스트 파일에 자료를 쓰거나 읽기 위한 함수

함수 fprintf()와 fscanf() 함수원형

```
int fprintf(FILE * _File, const char * _Format, ...);  
int fscanf(FILE * _File, const char * _Format, ...);  
int fscanf_s(FILE * _File, const char * _Format, ...);
```

위 함수에서 _File은 서식화된 입출력 스트림의 목적지인 파일이며, _Format은 입출력 제어 문자열이며, 이후 기술되는 인자는 여러 개의 출력될 변수 또는 상수이다.

- 기호 상수 stdin, stdout, stderr

표준파일	키워드	장치(device)
표준입력	stdin	키보드
표준출력	stdout	모니터 화면
표준에러	stderr	모니터 화면

```
FILE* f;
char name[30];
int point1, point2, cnt = 0;

// 사용자 입력
printf("이름, 점수1, 점수2 입력: ");
scanf("%s%d%d", name, &point1, &point2);

// 파일에 쓰기
f = fopen("grade.txt", "w");
fprintf(f, "%d %s %d %d\n", ++cnt, name, point1, point2);
fclose(f);

// 파일에서 다시 읽기
f = fopen("grade.txt", "r");
fscanf(f, "%d %s %d %d\n", &cnt, name, &point1, &point2);
fclose(f);

// 출력 확인
printf("번호: %d, 이름: %s, 점수1: %d, 점수2: %d\n", cnt, name, point1, point2);
```

예제 : 파일처리

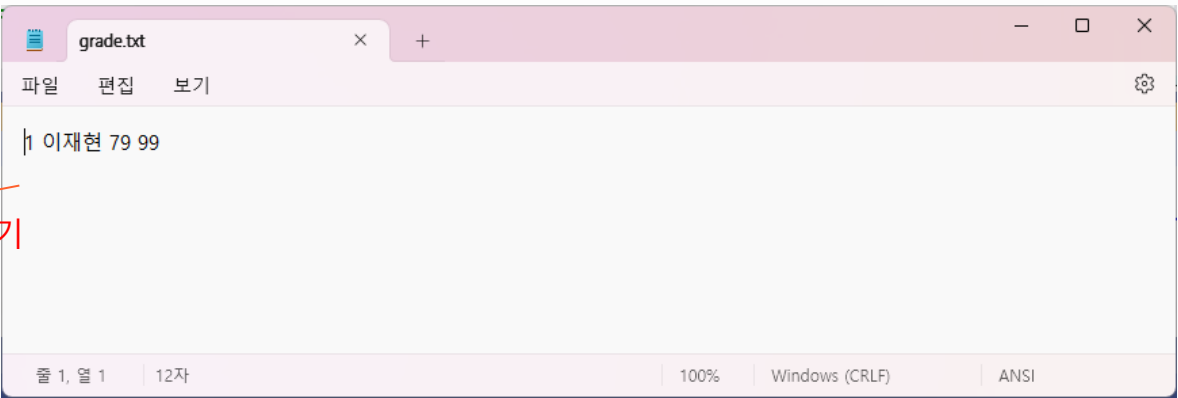
이름과 성적(중간, 기말)을 입력하세요.
이재현 79 99

표준입력

번호	이름	중간	기말
1	이재현	79	99

파일쓰기

파일읽기



grade.txt

파일 편집 보기

이재현 79 99

줄 1, 열 1 | 12자 | 100% | Windows (CRLF) | ANSI

```

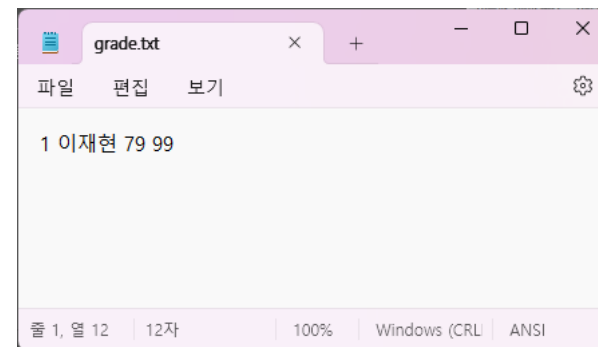
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <stdlib.h>
04
05 int main()
06 {
07     char fname[] = "grade.txt";
08     FILE* f;
09     char name[30];
10     int point1, point2, cnt = 0;
11
12
13
14     if ((f = fopen(fname, "w")) == NULL) {
15         printf("파일이 열리지 않습니다.\n");
16         exit(1);
17     }
18
19
20     printf("이름과 성적(중간, 기말)을 입력하세요.\n");
21     scanf("%s %d %d", name, &point1, &point2);
22     fprintf(f, "%d %s %d %d\n", ++cnt, name, point1, point2);
23     fclose(f);
24
25
26
27     if ((f = fopen(fname, "r")) == NULL) {
28         printf("파일이 열리지 않습니다.\n");
29         exit(1);
30     }
31
32
33     fscanf(f, "%d %s %d %d\n", &cnt, name, &point1, &point2);
34     fclose(f);
35
36
37     fprintf(stdout, "\n%6s%16s%10s%8s\n", "번호", "이름", "중간", "기말");
38     fprintf(stdout, "%5d%18s%8d%8d\n", cnt, name, point1, point2);
39
40     return 0;
41 }

```

이름과 성적(중간, 기말)을 입력하세요.

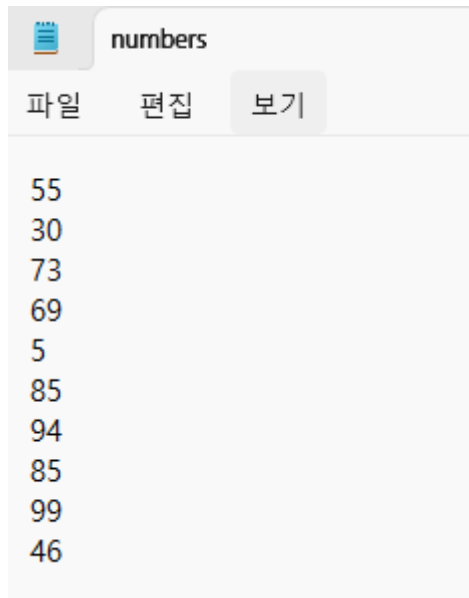
이재현 79 99

번호	이름	중간	기말
1	이재현	79	99



코드 작성

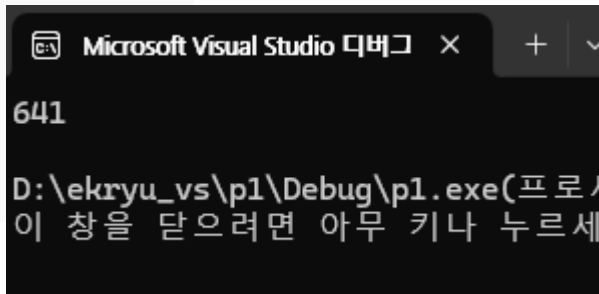
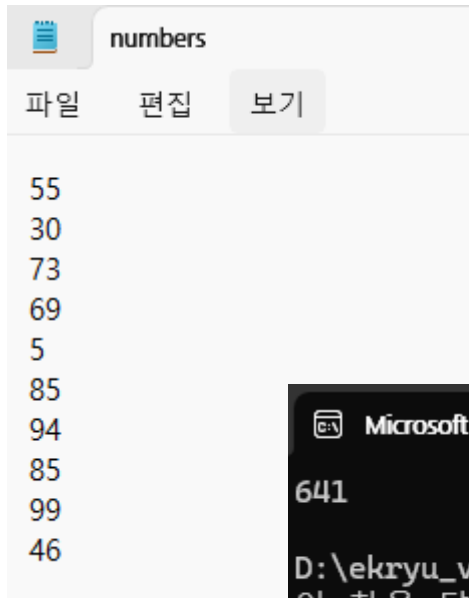
정수 난수(0~99) 10개를 numbers.txt에 출력



```
...  
srand( (unsigned int) time(NULL) ) ;  
...
```

코드작성

numbers.txt에서 정수 데이터 읽어 누적합 계산



함수 fgets()와 fputs()

함수 fgets()와 fputs() 함수원형

```
char * fgets(char * _Buf, int _MaxCount, FILE * _File);  
int  fputs(char * _Buf, FILE * _File);
```

- 함수 fgets()는 _File로부터 한 행의 문자열을 _MaxCount 수의 _Buf 문자열에 입력 수행
- 함수 fputs()는 _Buf 문자열을 _File에 출력 수행

```
char names[80];  
FILE *f;  
  
fgets(names, 80, f);  
fputs(names, f);
```

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

int main(void) {
    FILE *fp = fopen("1.c", "r");
    char line[256];

    if (fp == NULL) {
        perror("파일 열기 실패");
        return 1;
    }

    while (fgets(line, sizeof(line), fp) != NULL) {
        printf("%s", line); // 개행 포함됨 → 줄바꿈 자동 처리
    }

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


함수 feof()와 ferror()

함수 feof()와 ferror() 함수원형

```
int feof(FILE * _File);  
int ferror(FILE * _File);
```

- 함수 feof()은 _File의 EOF를 검사
- 함수 ferror()는 _File에서 오류발생 유무를 검사

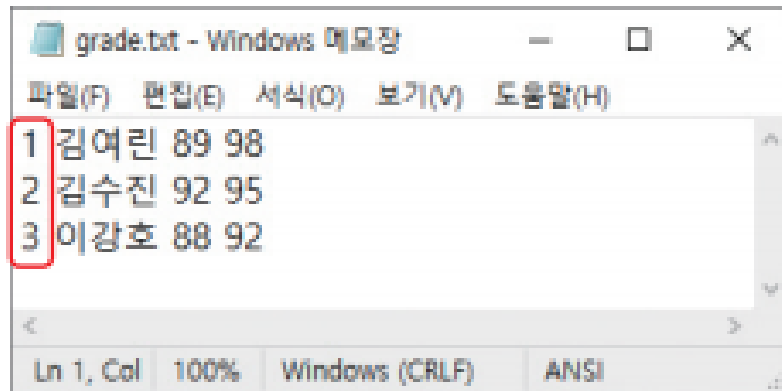
```
while ( !feof(stdin) )  
{  
    ...  
    fgets(names, 80, stdin);    //표준입력  
}
```

주의) feof는 EOF에 도달한 후 읽기를 시도해야 참 반환

이름과 성적(중간, 기말)을 입력하세요.

```
김여린 89 98  
김수진 92 95  
이강호 88 92  
^Z
```

표준입력으로 여러 줄에 걸쳐 적당한 형태로 입력하고 마지막
행에는 반드시 키 ctrl + Z를 입력한 후 Enter를 친다.



```
grade.txt - Windows 메모장  
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)  
1 김여린 89 98  
2 김수진 92 95  
3 이강호 88 92  
Ln 1, Col 100% Windows (CRLF) ANSI
```

```

char fname[] = "grade.txt";
char names[80];
int cnt = 0;
FILE* f;

f = fopen(fname, "w");
if (f == NULL) {
    printf("파일이 열리지 않습니다.\n");
    exit(1);
}

printf("이름과 성적(중간, 기말)을 입력하세요.\n");

fgets(names, 80, stdin);

while (!feof(stdin)) {
    fprintf(f, "%d ", ++cnt);
    fputs(names, f);
    fgets(names, 80, stdin);
}

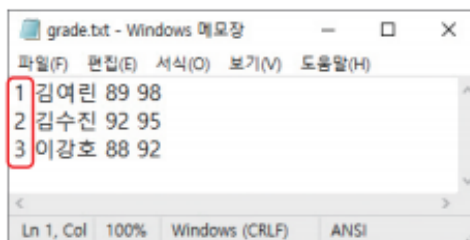
fclose(f);
return 0;

```

이름과 성적(중간, 기말)을 입력하세요.

김여린 89 98
 김수진 92 95
 이강호 88 92
 ^Z

표준입력으로 여러 줄에 걸쳐 적당한 형태로 입력하고 마지막
 행에는 반드시 키 ctrl + Z를 입력한 후 Enter를 친다.



grade.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

1 김여린 89 98
 2 김수진 92 95
 3 이강호 88 92

Ln 1, Col 100% Windows (CRLF) ANSI

```
char fname[] = "grade.txt";
char names[80];
int cnt = 0;
FILE* f;

f = fopen(fname, "w");
if (f == NULL) {
    printf("파일이 열리지 않습니다.\n");
    exit(1);
}

printf("이름과 성적(중간, 기말)을 입력하세요.\n");

fgets(names, 80, stdin);

while (!feof(stdin)) {
    fprintf(f, "%d ", ++cnt);
    fputs(names, f);
    fgets(names, 80, stdin);
}

fclose(f);
return 0;
```



```
printf("이름과 성적(중간, 기말)을 입력하세요.\n");
printf("(빈 줄 입력 시 종료됩니다)\n");

while (fgets(names, sizeof(names), stdin) != NULL && names[0] != '\n') {
    fprintf(f, "%d ", ++cnt);
    fputs(names, f);
}
```

함수 fgetc()와 fputc()

함수 fgetc()와 fputc() 함수원형

```
int fgetc(FILE * _File);  
int fputc(int _Ch, FILE * _File);  
  
int getc(FILE * _File);  
int putc(int _Ch, FILE * _File);
```

- 함수 fgetc()와 getc()는 _File에서 문자 하나를 입력받는 함수
- 함수 fputc()와 putc()문자 _Ch를 파일 _File 에 출력하는 함수

예제 : 문자 단위 출력

```
1: #include <stdio.h>
2: #include <stdlib.h>
3:
4: int main(void)
5: {
6:     FILE* f;
... 중간생략
24:
25:     return 0;
26: }
27:
```

줄 번호와 함께 파일 내용을 그대로 출력

```

01 #include <stdio.h>
02 #include <stdlib.h>
03
04 int main(void)
05 {
06     FILE* f;
07     int ch, cnt = 0;
08
09     f = fopen("05flist.c", "r");
10     if (f == NULL) {
11         printf("파일이 열리지 않습니다.\n");
12         exit(1);
13     }
14
15     printf("%4d: ", ++cnt);
16
17     while ((ch = fgetc(f)) != EOF) {
18         putchar(ch);
19         if (ch == '\n')
20             printf("%4d: ", ++cnt);
21     }
22
23     printf("\n");
24     fclose(f);
25
26     return 0;
27 }

```

```

1: #include <stdio.h>
2: #include <stdlib.h>
3:
4: int main(void)
5: {
6:     FILE* f;
... 중간생략
24:
25:     return 0;
26: }
27:

```

줄 번호와 함께 파일 내용을 그대로 출력

```

FILE *f1, *f2;

f1 = fopen("char.c", "r");
if (f1 == NULL) {
    perror("Failed to open input file (char.c)");
    return 1;
}

f2 = fopen("con.c", "w");
if (f2 == NULL) {
    perror("Failed to create output file (con.c)");
    fclose(f1);
    return 1;
}

int ch;
while ((ch = getc(f1)) != EOF) {
    putchar(ch);

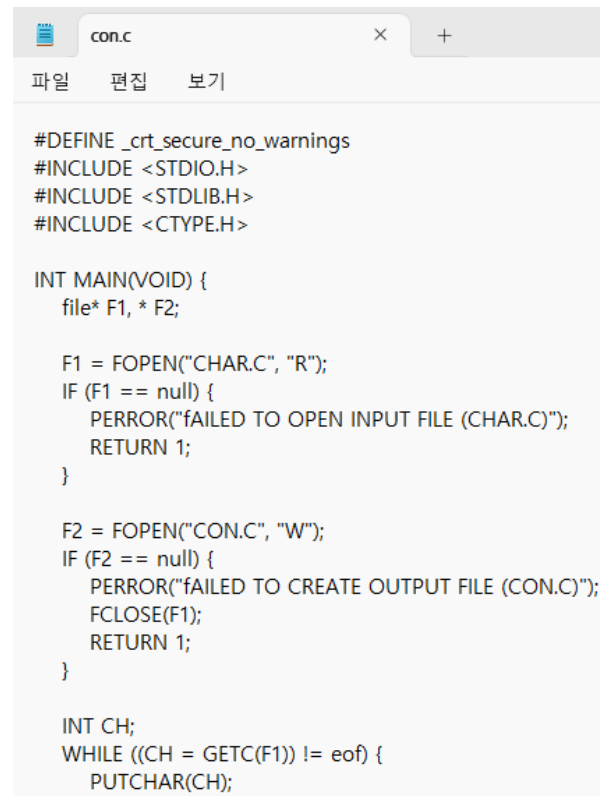
    if (isalpha(ch)) {
        ch = islower(ch) ? toupper(ch) : tolower(ch);
    }

    putc(ch, f2);
}

fclose(f1);
fclose(f2);

printf("\nFile con.c is created!\n");
return 0;

```



```

con.c
파일 편집 보기

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>

int main(void) {
    file* f1, * f2;

    f1 = fopen("char.c", "r");
    if (f1 == null) {
        perror("failed to open input file (char.c)");
        return 1;
    }

    f2 = fopen("con.c", "w");
    if (f2 == null) {
        perror("failed to create output file (con.c)");
        fclose(f1);
        return 1;
    }

    int ch;
    while ((ch = getc(f1)) != eof) {
        putchar(ch);

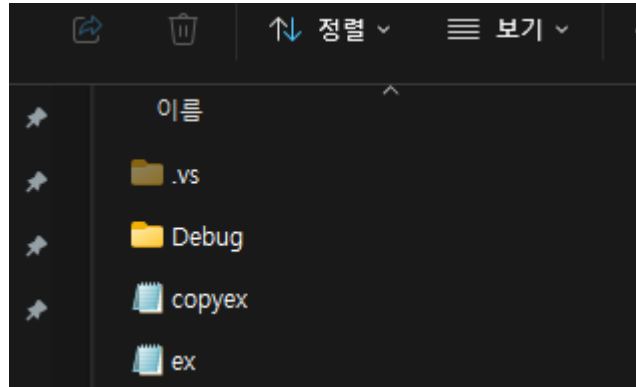
```


ctype.h 문자 처리 함수

함수명	분류	설명	예시 (c)	반환 결과
<code>isalpha(c)</code>	분류	알파벳인지 확인 (A-Z, a-z)	'A', 'z'	참 (1)
<code>isdigit(c)</code>	분류	숫자인지 확인 (0-9)	'3'	참 (1)
<code>isalnum(c)</code>	분류	알파벳 또는 숫자인지 확인	'B', '7'	참 (1)
<code>isspace(c)</code>	분류	공백 문자인지 확인 (' ', \t, \n 등)	'\n', ' '	참 (1)
<code>islower(c)</code>	분류	소문자인지 확인 (a-z)	'x'	참 (1)
<code>isupper(c)</code>	분류	대문자인지 확인 (A-Z)	'Q'	참 (1)
<code>ispunct(c)</code>	분류	구두점 문자인지 확인 (., ,, ?, ! 등)	!', ', ,, ?, !	참 (1)
<code>isxdigit(c)</code>	분류	16진수 문자 확인 (0-9, A-F, a-f)	'F', '8'	참 (1)
<code>tolower(c)</code>	변환	대문자를 소문자로 변환	'G' → 'g'	'g'
<code>toupper(c)</code>	변환	소문자를 대문자로 변환	'k' → 'K'	'K'
<code>iscntrl(c)</code>	기타	제어 문자 확인 (\n, \t, \b 등)	'\t'	참 (1)
<code>isprint(c)</code>	기타	출력 가능한 문자 (ASCII 32~126)인지 확인	'A', '@'	참 (1)
<code>isascii(c)</code>	기타	ASCII 범위 (0~127)인지 확 인	'Z', 127	참 (1)

코드작성 : 파일 복사

ex.c를 copyex.c로 복사 (fgetc, fputc)



파일 처리

- **01** 파일 기초
- **02** 텍스트 파일 입출력
- **03** 이진 파일 입출력
- **04** 파일 접근 처리

} 파일 입출력 처리
텍스트 파일 I/O 관련 함수
파일 포인터
stdin, stdout