

컴프 #7 수업에서 다루어지는 것들: 파일 입출력

■ 스트림과 FILE

- 스트림(stream): 입력과 출력을 바이트(byte)들의 흐름으로 생각
- 스트림은 구체적으로 FILE 구조체를 통하여 구현
- FILE은 stdio.h에 정의되어 있다.

□ 표준 입출력 스트림(standard input/output stream): 필수적 몇 개의 스트림

- stdin: 표준 입력 스트림(키보드)
- stdout: 표준 출력 스트림(모니터의 화면)
- stderr: 표준 에러 스트림(모니터의 화면)

■ 파일

- 파일이 필요한 이유:
- 텍스트 파일 vs 이진 파일(사람은 못읽고 컴퓨터는 읽음. Ex 실행파일)
- 파일 처리

파일 열기 -> 파일 읽기, 쓰기 -> 파일 닫기

□ 파일 포인터의 선언

FILE *fp; // 파일을 가리킬 포인터 변수의 선언

□ 파일 열기: fopen

원형: FILE *fopen(const char *filename, const char * mode);

기능: 파일을 열고 성공시 해당 파일의 FILE 구조체 변수의 주소값, 실패 시 NULL 포인터를 반환한다.

파일에서 데이터를 읽거나 쓸 수 있도록 모든 준비

```
fp1 =
if (fp1 == NULL)
{
    printf("파일 오픈 에러... !!!\n");
    return 1;
}
```

□ 파일에 읽기, 쓰기

입력 파일 포인터에 stdin을 쓰면?

출력 파인 포인터에 stdout를 쓰면?

1. 문자 단위 파일 입출력

입력		출력	
ch = fgetc(fp1);	conio.h	fputc(ch, fp2);	conio.h
ch = getc(fp1);	stdio.h	putc(ch, fp2);	stdio.h

연습1: 문자를 파일에서 읽기

ch = getc(fp1);

연습2: 문자를 파일에 쓰기

putc(ch, fp2);

연습3: 파일의 끝까지 읽어서 쓰기

- feof(파일포인터) 함수는 파일 끝이 아닌 경우 0을, 파일 끝에 도달하면 0이 아닌 값을 반환

// version 1 ch = <u>getc(fp1)</u> ; while(!feof(fp1)) // fp가 파일의 끝에 다다르면 { putc(ch, fp2); //fp2는 출력 파일 포인터 ch = <u>getc(fp1)</u> ; }	// version 2 while ((ch = getc(fp1)) != EOF) putc(ch, fp2);
---	---

2. 문자열 단위 파일 입출력

입력	출력
fgets(str, size, fp); // 'W0' 을 포함해서 size의 크기만큼 // 혹은 엔터가 나올때까지 읽어 str에 넣는다. //(이때, 엔터까지 str에 넣음!!) // sizeof(str) 이상이면 (size - 1)까지 읽고 다음에 'W0' 넣음	fputs(str, fp2); // str이 가르키는 곳부터 // 'W0이 나올때까지

char buf[30];

연습1: 문자열 단위로 파일에서 읽기

연습2: 문자열 단위로 파일에서 쓰기

연습3: 문자열 단위로 파일의 끝까지 읽어서 쓰기

```
fgets(buf, sizeof(buf), fp1);
while(!feof(fp1))
{
    fputs(buf, fp2);
    fgets(buf, sizeof(buf), fp1);
}
```

3. 데이터 타입을 지정한 입출력

입력	출력
fscanf(fp, ...);	fprintf(fp, ...);

연습1: 파일내의 숫자 num 읽기

fscanf(fp1, "%d", &num);

연습2: 파일 내의 숫자 num 쓰기

fprintf(fp1, "%d", num);

연습3: 데이터 타입을 지정하면서 파일의 끝까지 읽어서 쓰기

```
fscanf(fp1, "%d", &score);
while(!feof(fp1)) {
    fprintf(fp2, "%d\n", score + 9);
    fscanf(fp1, "%d", &score);
}
```

□ 파일 닫기: fclose

원형: int fclose(FILE *stream);

기능: 파일 닫는 것을 성공시 0, 실패시 EOF를 반환한다.

```
fclose(fp1);
fclose(fp2);
```

□ 파일의 임의 접근: fseek을 이용한 파일위치표지자 조작

파일위치표지자는 파일을 오픈할 때, 읽을때, 쓸때 이동한다.

아래처럼 파일위치표지자의 위치를 이동할 수있다.

fseek(fp, 0, SEEK_SET); // 파일의 처음으로 이동

fseek(fp, 0, SEEK_END); // 파일의 끝으로 이동

LAB 12 파일 처리(수정본)

아래의 두 가지에 유의하시오.

- 파일이 만들어진 경우 해당 파일을 click해서 notepad로 파일 내용을 확인해본다.
- 파일을 열거나 닫을 때 아래와 같이 if문을 사용하여 error를 체크한다(이는 프로그램의 좋은 습관이다)

```
FILE * fp;
int state;
```

```
fp = fopen(...);
if (fp == NULL) // error 체크
{
    printf("file open error!\n");
    return 1;
}
```

.. // 여기서 뭔가를 하고

```
state = fclose(fp);
if (state != 0) // error 체크. fclose에서의 error 체크는 주로 생략한다.
{
    printf("file close error!\n");
    return 1;
}
```

- 파일의 입출력의 기본을 실습하자.

- **LAB12.1.1(fopen, fclose, fprintf)** (출력 파일) 파일 "hello.out"에 Hello World를 출력하는 프로그램을 작성하라.

프로그램의 구성은 다음과 같다.

- 파일을 출력모드로 오픈: fopen() 함수의 파일 오픈 모드는 "w"로 한다. (즉, text 모드의 파일로 생성. "wt"와 동일한 효과)
- 파일에 쓰기: fprintf 사용
- 파일 닫기: fclose 사용

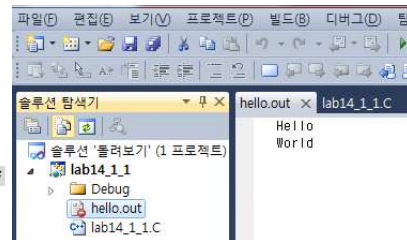
실행시킨 후(화면에는 아무것도 나타나지 않는다!), [솔루션 탐색기]에서 LAB12_1_1 폴더에 hello.out이 성공적으로 만들어져 있나 열어서 확인한다.

```
#include <stdio.h>
int main(void)
{
    int state;
    FILE * fp;
    char ch;

    fp = fopen(______);
    if (fp == NULL)
    {
        printf("file open error!\n");
        return 1;
    }

    fprintf(______);
    fprintf(______);

    state = fclose(fp);
    if (state != 0) // 이 체크는 생략 가능
    {
        printf("file close error!\n");
        return 1;
    }
}
```



- **LAB12.1.2(fopen, fclose, getc, putc, feof)**(입력 파일)"hello.in"파일의 내용을 화면에 출력하는 프로그램을 작성하라.

입력파일로 hello.in가 존재해야한다.

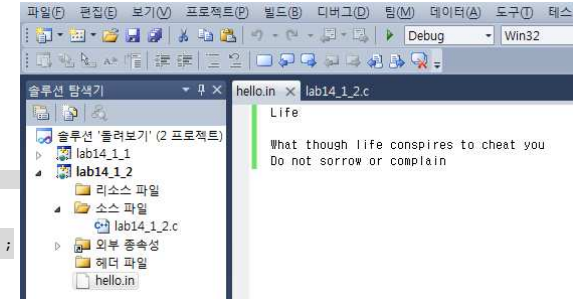
프로그램의 구성은 다음과 같다.

- 파일을 입력모드로 오픈("r" 혹은 "rt"를 사용)
- 파일로부터 읽어서 화면에 출력하기(getc, putc의 사용)

```
#include <stdio.h>
int main(void)
{
    int state;
    FILE * fp;

    char ch;

    fp = fopen(
    if (fp == NULL)
    {
        printf("파일 오픈 에러입니다!!!\n");
        return 1;
    }
}
```



// getc, putc를 사용하여 파일의 끝까지 읽고 쓴다

```
fclose(fp);

return 0;
}
```

□ **LAB12_1_3 (입력파일, 출력파일) (fopen, fclose, fscanf, fprintf, feof)**

score.in에 있는 점수를 읽어들이면서 그 점수에 9점을 더해서 score.out에 출력하는 프로그램을 작성하라.

```
#include <stdio.h>
int main(void)
{
    FILE * fp1, *fp2;

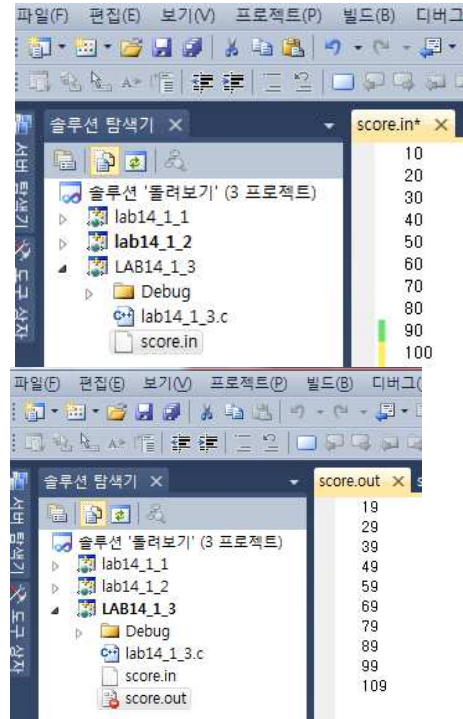
    int score;

    fp1 = fopen("score.in", "r");
    if (fp1 == NULL)
    {
        printf("파일 오픈 에러입니다!!!\n");
        return 1;
    }

    fp2 = fopen("score.out", "w");
    if (fp2 == NULL)
    {
        printf("file open error!\n");
        return 1;
    }

    // 읽고 처리(+9를 더함)해서 쓴다

    fclose(fp1);
    fclose(fp2);
}
```



□ **LAB12_1_4** 위의 LAB12_1_1과 LAB12_1_2의 내용을 하나로 합쳐서 하나의 프로그램으로 작성할 수 있다.

즉, Hello와 World가 쓰여진 출력 파일을 만들고, 이를 다시 입력 파일로 읽어서 화면에 출력하는 프로그램을 작성하라.

프로그램의 구성은 다음과 같다.

- 파일을 출력모드로 오픈
- 파일에 쓰기: fprintf 사용
- 파일 닫기
- 파일을 입력모드로 오픈
- 파일로부터 읽어서 화면에 출력: getc, putc 사용
- 파일 닫기

- 파일 입출력을 조금 더 실습해보자.

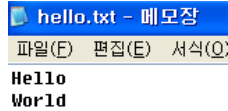
□ **LAB12.2.1(fopen, fclose, fprintf)** ((파일에 내용 추가하기)

“hello.txt”파일의 내용에 Hi, Everybody라는 내용을 추가하는 프로그램을 작성하라.

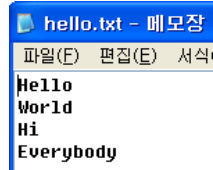
입력파일로 hello.txt가 존재해야한다.

파일의 끝에 새로운 라인을 두 줄 추가(fprintf사용)하려 한다. Hello.txt파일을 fopen() 함수를 통해 “읽어들일 때 어떤 파일 오픈 모드로 fopen() 함수를 호출하는 것이 적합한 지를 생각해서 코딩한다. 실행시킨후 반드시 hello.txt 파일을 열어 내용을 확인하라.

<프로그램 수행 전 파일>>



추가된 후



프로그램의 구성은 아래와 같다.

- 파일 오픈
- 파일에 추가해서 쓰기
- 파일 닫기

□ **LAB12.2.2(fopen, fclose, fgets, fputs)** “hello.txt” 파일과 동일한 내용을 가지는 “hello2.txt”를 생성하는 프로그램을 작성하라.

한 줄 씩 읽는 fgets와 fputs를 반드시 사용해보라. 실행후 hello2.txt가 제대로 생성되었는가를 확인한다.

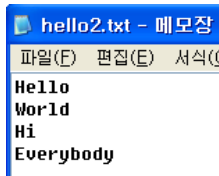
입력파일로 hello.txt가 존재해야한다.

- Notepad를 사용하여 hello.txt파일을 만들거나
- [윈도우 탐색기]에서 앞의 **LAB12.2.1**에 있는 hello.txt파일을 현재 폴더로 복사해 온다. (hello.txt 파일이 없으면 file open error가 발생한다)

<입력 파일>



<출력 파일>



프로그램의 구성은 아래와 같다.

- 입력 파일 오픈
- 출력 파일 오픈
- 입력파일에서 읽어 출력파일에 쓰기(fgets, fputs)

```
char buf[30];
.
.
.
fgets(buf, sizeof(buf), fp); // 입력 파일에서 한줄 읽어서
while(!feof(fp)) // 파일의 끝이 아니면 계속 반복
{
    fputs(buf, fp2); // 출력 파일에 출력
    fgets(buf, sizeof(buf), fp); // 입력 파일에서 또 읽어(다시) 한줄을 읽
}
```

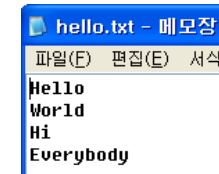
- 입력파일 닫기
- 출력파일 닫기

□ **LAB12.2.2 수행(fseek의 사용)** hello2.txt에 hello.txt의 내용이 두 번 반복되도록 하라.

즉, hello.txt의 내용을 읽어 hello2.txt에 적고, 다시 hello.txt를 읽어 hello2.txt에 계속 적는다. Hello.txt의 맨 앞으로 이동하기위해서 fseek을 사용하라.

실행 후, hello2.txt 파일이 성공적으로 생성되었는가 확인하라.

<입력 파일>



<출력 파일>



프로그램의 구성은 아래와 같다.

- 입력 파일 오픈
- 출력 파일 오픈
- 입력파일에서 읽어 출력파일에 쓰기(fgets, fputs)
- **입력 파일의 처음을 *다시* 읽을 준비하기(fseek 사용!!)**
- **입력파일에서 읽어 출력파일에 쓰기(fgets, fputs)**
- 입력파일 닫기
- 출력파일 닫기

HW 12

■ HW12.1 (난이도 중) 아래의 순서로 프로그램을 작성하라.

1. **파일에 출력:** 0부터 99까지의 난수 10개를 파일 random.txt에 한 줄에 한 개씩 쓰는 프로그램을 작성하라. (rand(), srand(time(NULL))을 사용한다.)
2. **파일에서 입력 받아서 화면에 출력:** 파일 random.txt를 읽어서 화면에 난수와 그 총 합을 출력하라. (파일에서 읽을 때 fscanf를 사용하고 파일의 끝을 확인하기위해 feof를 사용한다)
3. **또 다른 파일에 출력:** 2에서 화면에 출력 시 그 내용을 output.txt에 출력한다.

```
random.txt - 메모장
파일(F) 편집(E) 서식(O) ?
11
58
15
91
30
31
67
59
64
33
```

모니터 출력결과

```
C:\WINDOWS\system32\cmd.exe
11 58 15 91 30 31 67 59 64 33
합은 459
계속하려면 아무 키나 누르십시오 . . .
```

```
output.txt - 메모장
파일(F) 편집(E) 서식(O) ?
11
58
15
91
30
31
67
59
64
33
합은 459
```

■ HW12.2 (난이도 중) 아래의 순서로 프로그램을 작성하라.

1. 입력파일로 input.txt를 작성한다. 영어 문장 아무거나...

```
input.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
Near far
wherever you are
I believe that the heart does go on
Once more, you open the door
And you're here in my heart
And my heart will go on and on
```

2. 위의 input.txt를 읽어서 아래와 같이 모두 대문자로, 그다음은 모두 소문자로 위 영어문장을 output.txt파일에 출력하는 프로그램을 작성하라.

```
output.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
NEAR FAR
WHEREVER YOU ARE
I BELIEVE THAT THE HEART DOES GO ON
ONCE MORE, YOU OPEN THE DOOR
AND YOU'RE HERE IN MY HEART
AND MY HEART WILL GO ON AND ON

near far
wherever you are
i believe that the heart does go on
once more, you open the door
and you're here in my heart
and my heart will go on and on
```

PROJECT 03(3) – 비디오 관리 프로그램(파일 입출력)

■ Project3_3(최종)

```
#define MAX_VIDEO 100
#define MAX_CUST 200 // max customer
#define MAX_CHAR 100 // 문자열의 max 문자
#include <stdio.h>
#include <string.h>

typedef struct { // 재고 대장: 현재 보유하고 있는 video 정보 저장
    char title[MAX_CHAR];
    int qty; // 수량
} VideoInfo;

typedef struct { // 대출 대장: 대출해간 (고객 id와 video id)들이 저장
    int custId; // 고객 id : 1, 2, 3
    char title[MAX_CHAR]; // Video 제목
} RentInfo;

void printAllVideo(VideoInfo videoList[], int videoCount) {
    int i;
    printf("Video제목\t수량\n");
    printf("-----\n");
    for (i = 0; i < videoCount; i++)
        printf("%s\t%d\n", videoList[i].title, videoList[i].qty);
}

void purchaseVideo(VideoInfo videoList[], int *videoCountPtr, char *title, int qty) { /*구현*/}

int searchVideoByTitle(VideoInfo *videoList, int videoCount, char *title) { /*구현*/}

void rentVideo(// 매개변수 리스트 넣기) { /*구현*/}

void printAllRent(RentInfo rentList[], int rentCount) {
    int i;
    printf("고객id\tVideo제목\n");
    printf("-----\n");
    for (i = 0; i < rentCount; i++)
        printf("%d\t%s\n", rentList[i].custId, rentList[i].title);
}

void readVideo(VideoInfo videoList[], int *videoCountPtr)
{
    FILE *fp;
    VideoInfo video;
    if ((fp = fopen("video.txt", "r")) == NULL) {
        printf("입력 파일 오픈 실패\n"); exit(1);
    }
    // 비디오 정보를 입력파일에서 읽기
    ...
    fclose(fp);
}

void writeVideo(VideoInfo videoList[], int videoCount)
{
    FILE *fp;
    int i;

    if ((fp = fopen("video.txt", "w")) == NULL) {
        printf("출력 파일 오픈 실패\n"); exit(1);
    }
    // 비디오 정보를 출력 파일로 쓰기
    ...
    fclose(fp);
}
```

```
void writeRent(RentInfo rentList[], int rentCount)
{
    FILE *fp;
    int i;

    fp = fopen("rent.txt", "w");
    if (fp == NULL) {
        printf("출력 파일 오픈 실패\n"); exit(1);
    }
    for (i = 0; i < rentCount; i++)
        fprintf(fp, "%d %s\n", rentList[i].custId, rentList[i].title);
    fclose(fp);
}

int main(void)
{
    int videoCount = 0; // videoCount도 0으로 초기화
    VideoInfo videoList[MAX_VIDEO];
    int rentCount = 0; // 현재 대출 건수는 0임
    RentInfo rentList[MAX_CUST];

    char *title[MAX_CHAR];
    int qty, custId;
    int choice;
    int indexSearched;

    readVideo(videoList, &videoCount);

    printf("1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): ");
    scanf("%d", &choice);
    while (choice != 6) {
        switch(choice) {
            case 1: printAllVideo(videoList, videoCount); break;
            case 2:
                printf("Enter video 제목: ");
                scanf("%s", title);
                printf("Enter video 수량: ");
                scanf("%d", &qty);
                purchaseVideo(videoList, &videoCount, title, qty); break;
            case 3:
                printf("Enter video 제목: ");
                scanf("%s", title);
                if ((indexSearched = searchVideoByTitle(videoList, videoCount, title)) == -1)
                    printf("그런 비디오는 없습니다.\n");
                else if (videoList[indexSearched].qty == 0)
                    printf("다 대여중입니다.\n");
                else
                    printf("대여 가능합니다.\n"); break;
            case 4:
                printf("Enter video 제목: ");
                scanf("%s", title);
                printf("Enter 고객 id: ");
                scanf("%d", &custId);
                rentVideo(videoList, videoCount, rentList, &rentCount, title, custId); break;
            case 5:
                printAllRent(rentList, rentCount); break;
        }
        printf("1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): ");
        scanf("%d", &choice);
    }

    writeVideo(videoList, videoCount);
    writeRent(rentList, rentCount);
}
```