ROBIT
ROBOT SPORT GAME TEAM

C Language – 7일차

1. 구조체 포인터
2. 선행처리
3. 파일입출력

ROBIT
ROBOT SPORT GAME TEAM

구조체 포인터

구조체 포인터


```
#include <stdio.h>

typedef struct _Student
{
    int number;
    char name[10];
    double grade;
}Student;

void main(void)
{
    Student *p;
    Student s = {44,"kim",4.5};

    p = &s;

    printf("학번=%d 이름=%s 학점=%f ₩n", s.number, s.name, s.grade);
    printf("학번=%d 이름=%s 학점=%f ₩n", (*p).number,(*p).name,(*p).grade);

}
```

```
학번=44 이름=kim 학점=4.500000
학번=44 이름=kim 학점=4.500000
계속하려면 아무 키나 누르십시오 . . .
```

int num= 10;
int * iptr = #

일반적인 포인터 변수의 선언 및 연산과 동일
포인터 변수 p가 구조체 변수 s를 가리킴

구조체 포인터

연산자 -> : 구조체 포인터로 구조체 멤버를 참조

```c
#include <stdio.h>

typedef struct _Student
{
    int number;
    char name[10];
    double grade;
}Student;

void main(void)
{
    Student *p;
    Student s = {44,"kim",4.5};

    p = &s;

    printf("학번=%d 이름=%s 학점=%f ₩n", p->number,p->name,p->grade);

}
```

```
학번=44 이름=kim 학점=4.500000
계속하려면 아무 키나 누르십시오 . . .
```

*연산과 .연산 = ->연산

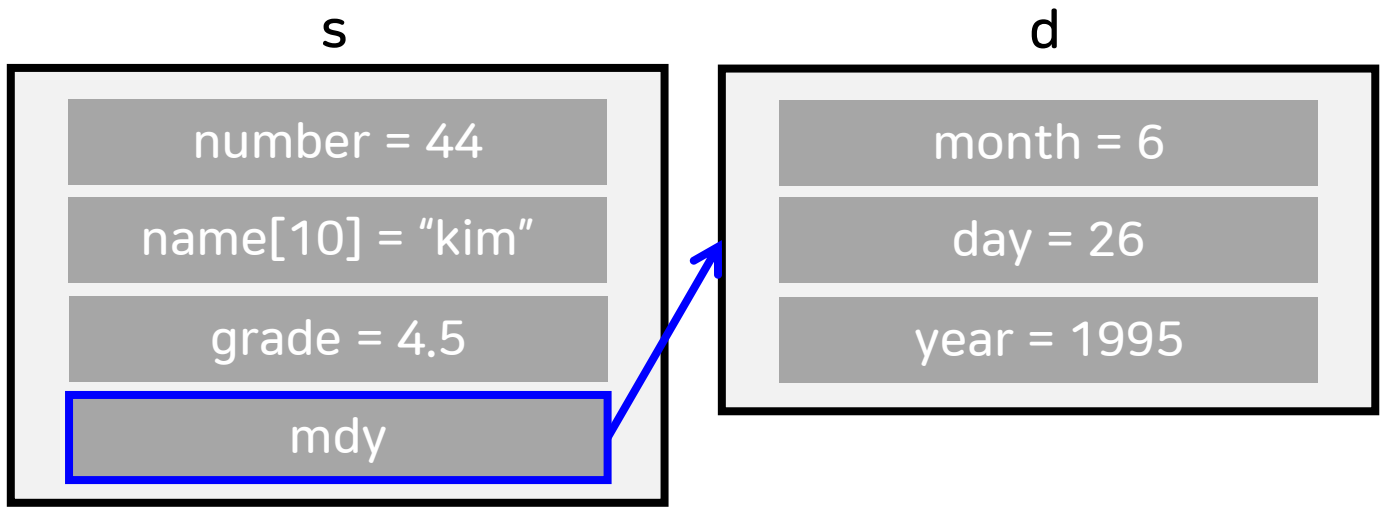
포인터를 멤버로 가지는 구조체


```c
#include <stdio.h>

typedef struct _Date
{
    int month;
    int day;
    int year;
}Date;

typedef struct _Student
{
    int number;
    char name[10];
    double grade;
    Date *mdy;
}Student;

void main(void)
{
    Student s = {44,"kim",4.5};
    Date d = {6,26,1995};

    s.mdy = &d;

    printf("학번 : %d, 이름 : %s, 학점 : %lf \n", s.number,s.name,s.grade);
    printf("생년월일: %d년 %d월 %d일\n", s.mdy->year, s.mdy->month, s.mdy->day);
}
```

```
학번 : 44, 이름 : kim, 학점 : 4.500000
생년월일: 1995년 6월 26일
계속하려면 아무 키나 누르십시오 . . .
```

포인터를 맴버로 가지는 구조체


```c
#include <stdio.h>

typedef struct _Date
{
    int month;
    int day;
    int year;
}Date;

typedef struct _Student
{
    int number;
    char name[10];
    double grade;
    Date *mdy;
}Student;

void main(void)
{
    Student s = {44,"kim",4.5};
    Date d = {6,26,1995};

    s.mdy = &d;

    printf("학번 : %d, 이름 : %s, 학점 : %lf \n", s.number,s.name,s.grade);
    printf("생년월일: %d년 %d월 %d일\n", s.mdy->year, s.mdy->month, s.mdy->day);
}
```



구조체 배열 동적할당

구조체도 malloc을 통해 동적할당을 할 수 있다.

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

//학생 정보 구조체
typedef struct _Student
{
    int number;
    char name[20];

}Student;

void ADD(Student* list,int number)
{
    (*(list + number)).number = number;
    printf("이름을 입력하시오 : ");
    scanf("%s", (*(list + number)).name);

    //list[number].number = number;
    //printf("이름을 입력하시오 : ");
    //scanf("%s", list[number].name);
}
```

```c
int main(void)
{
    Student* list;

    int student_num = 2;

    //배열 동적 할당
    list = (Student*)malloc(sizeof(Student) * student_num);

    ADD(list, 0);
    ADD(list, 1);

    for (int i = 0; i < student_num; i++)
    {
        printf("--------------------₩n");
        printf("이름 : %s₩n", list[i].name);
        printf("번호 : %d₩n₩n", list[i].number);
    }
```

realloc

malloc을 통해 할당된 메모리의 공간을 더 늘리거나 줄이기 위해 사용한다.

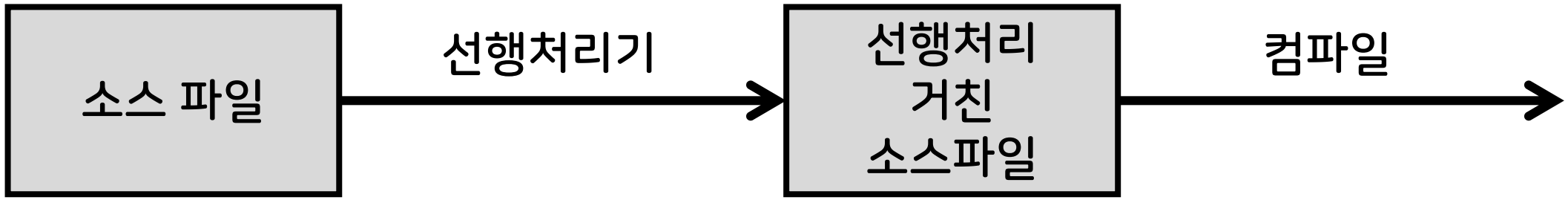
realloc(pointer, size);

```
int main(void)
{
    Student* list;

    int student_num = 2;

    //배열 동적 할당
    list = (Student*)malloc(sizeof(Student) * student_num);
    ADD(list, 0);
    ADD(list, 1);

    for (int i = 0; i < student_num; i++)
    {
        printf("-------------------₩n");
        printf("이름 : %s₩n", list[i].name);
        printf("번호 : %d₩n₩n", list[i].number);
    }

    student_num = 3;
    list = (Student*)realloc(list, sizeof(Student) * (student_num));
    ADD(list, 2);

    for (int i = 0; i < student_num; i++)
    {
        printf("-------------------₩n");
        printf("이름 : %s₩n", list[i].name);
        printf("번호 : %d₩n₩n", list[i].number);
    }

    //메모리 해제
    free(list);
    return 0;
}
```

ROBIT
ROBOT SPORT GAME TEAM

선행처리

선행처리(Preprocess)란?



선행처리 : 실행 파일을 생성하는 과정에서 소스 파일 내에 존재하는 선행처리 지시문을 처리하는 작업을 의미

- 선행처리 작업은 컴파일하기 전 선행처리기(preprocessor)에 의해 처리된다

- 선행처리기는 컴파일러가 컴파일하기 좋도록 소스를 재구성해 주는 역할을 한다

선행처리 지시문

ROBIT
ROBOT SPORT GAME TEAM

- 선행처리문은 선행처리 문자(#)로 시작한다
- 선행처리문은 코드 내에서 하나의 라인을 모두 차지하며, 선행처리문 뒤에 C언어 코드를 추가하여 같이 사용할 수 없다
- 선행처리문은 뒤에 세미콜론(;)을 붙이지 않는다
- 선행처리문은 소스 파일 어디에나 위치할 수 있지만, 선행처리문이 위치한 곳에서부터 파일의 끝까지만 영향을 미친다

선행처리 지시자

선행처리 지시자	설명
#include	외부에 선언된 함수나 상수 등을 사용하기 위해, 함수나 상수가 포함된 외부 파일을 현재 파일에 포함할 때 사용함.
#define	함수나 상수를 단순화해주는 매크로를 정의할 때 사용함.
#undef	#define 지시자로 이미 정의된 매크로를 삭제할 때 사용함.
#line	__LINE__ 매크로와 __FILE__ 매크로를 재정의할 때 사용함.
#error	지정한 오류 메시지를 출력하고, 컴파일 과정을 중단하고자 할 때 사용함.
#pragma	프로그램의 이식성을 위해 운영체제별로 달라지는 지시사항을 컴파일러에 전달할 때 사용함.
#if, #ifdef, #ifndef, #elif, #else, #endif	조건부 컴파일 지시자

#include

```
#include <stdio.h>
#include "myStdio.h"
```

- C언어에서 제공하는 표준 헤더 파일을 포함할 때에는 보통 꺾쇠괄호(< >)를 사용한다
- 사용자가 직접 작성한 헤더 파일을 포함할 때에는 보통 큰따옴표(“ ”)를 사용한다

  두 방법에 큰 차이는 없지만 통상적으로 위 기준에 맞춰서 코드를 작성한다.

#define

#define	식별자	대체리스트
#define	PI	3.14

- #define 선행처리 지시자는 함수나 상수를 단순화 해주는 매크로를 정의할 때 사용한다
- 함수 혹은 상수의 이름을 붙임으로써, 해당 매크로가 무엇을 가르키고 있는지 명확하게 나타내준다
- 가독성 증가

#define

ROBIT
ROBOT SPORT GAME TEAM

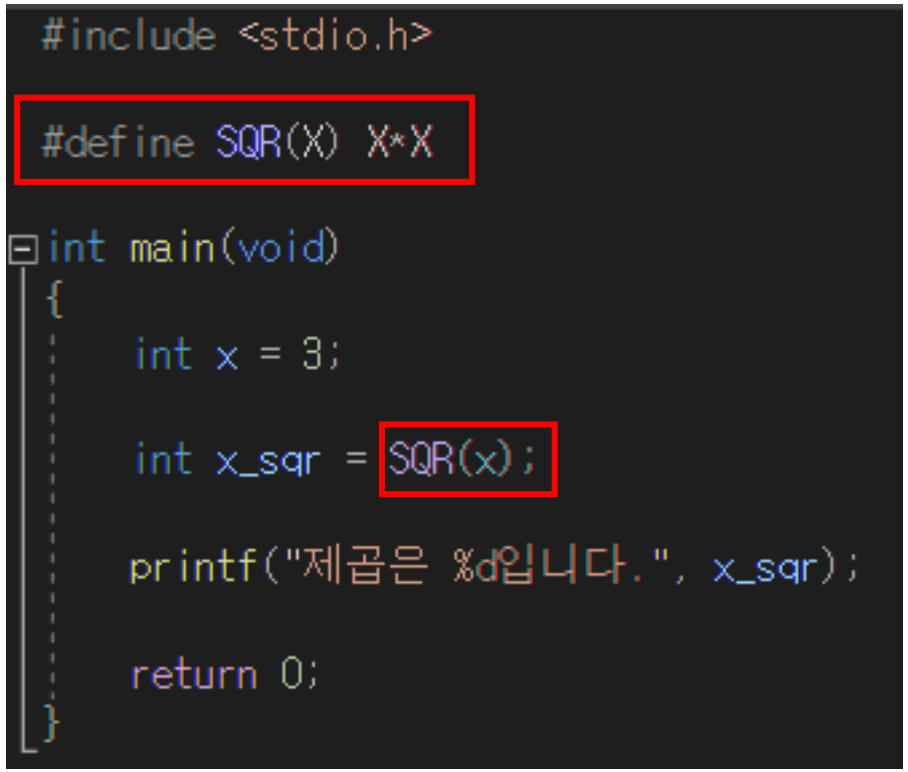
소스파일

```c
#include <stdio.h>

#define PI 3.1415926535

double GetCircleArea(int radius);

int main(void)
{
    int radius = 1;

    double area = GetCircleArea(radius);

    printf("반지름이 %d인 원의 넓이는 %lf입니다.", radius, area);

    return 0;
}

double GetCircleArea(int r)
{
    double area;
    area = r * r * PI;

    return area;
}
```

선행처리 후

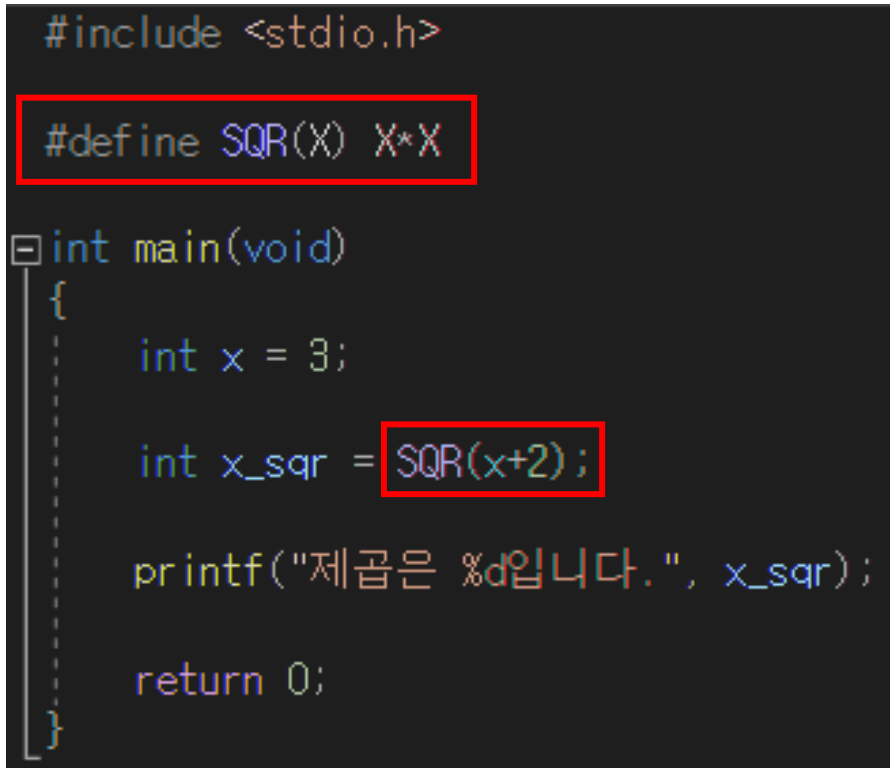
```c
#include <stdio.h>

double GetCircleArea(int radius);

int main(void)
{
    int radius = 1;

    double area = GetCircleArea(radius);

    printf("반지름이 %d인 원의 넓이는 %lf입니다.", radius, area);

    return 0;
}

double GetCircleArea(int r)
{
    double area;
    area = r * r * 3.1415926535;

    return area;
}
```

#define



```c
#include <stdio.h>

#define SQR(X) X*X

int main(void)
{
    int x = 3;

    int x_sqr = SQR(x);

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

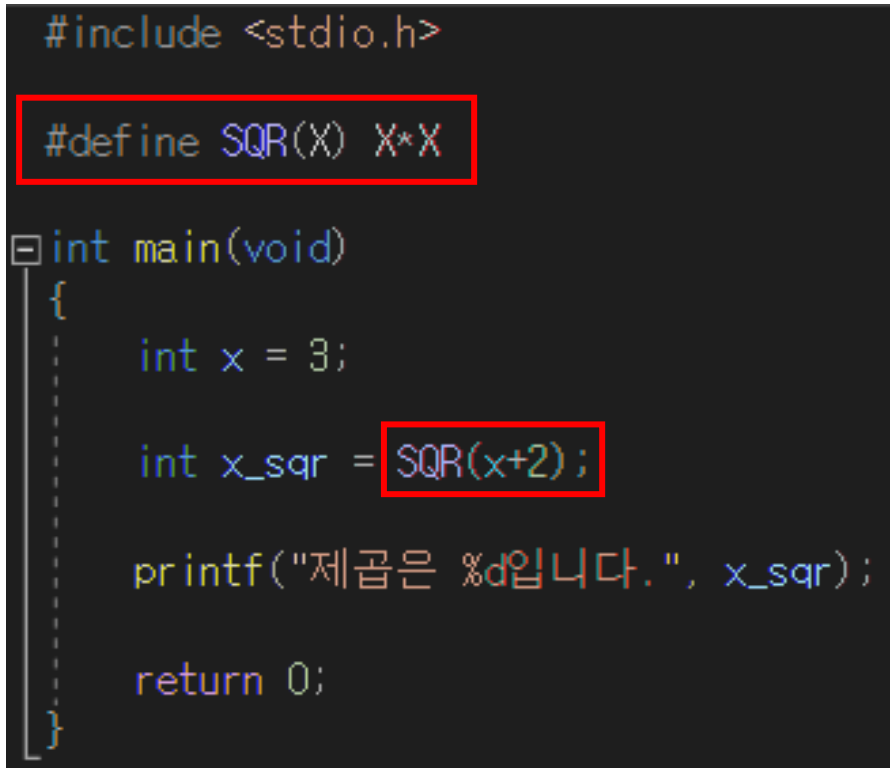
#define



```c
#include <stdio.h>

#define SQR(X) X*X

int main(void)
{
    int x = 3;

    int x_sqr = SQR(x+2);

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

?

#define



```c
#include <stdio.h>

#define SQR(X) X*X

int main(void)
{
    int x = 3;

    int x_sqr = SQR(x+2);

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

제곱은 11입니다.

#define

ROBIT
ROBOT SPORT GAME TEAM

소스파일

```c
#include <stdio.h>

#define SQR(X) X*X

int main(void)
{
    int x = 3;

    int x_sqr = SQR(x+2);

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

선행처리 후

```c
#include <stdio.h>

int main(void)
{
    int x = 3;

    int x_sqr = x+2*x+2;

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

3+2*3+2 = 11

#define

ROBIT
ROBOT SPORT GAME TEAM

소스파일

```c
#include <stdio.h>

#define SQR(X) ((X)*(X))

int main(void)
{
    int x = 3;

    int x_sqr = SQR(x+2);

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

선행처리 후

```c
#include <stdio.h>

int main(void)
{
    int x = 3;

    int x_sqr = ((x+2)*(x+2));

    printf("제곱은 %d입니다.", x_sqr);

    return 0;
}
```

(3+2)*(3+2) = 25

#if, #elif, #else, #endif

ROBIT
ROBOT SPORT GAME TEAM

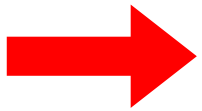
- 컴파일되기 전에 조건을 평가하여, 코드를 컴파일 대상에 포함시키거나 제외시키는 역할을 한다

#if, #elif, #else, #endif

ROBIT ROBOT SPORT GAME TEAM

소스파일

```c
#include <stdio.h>

#define A 1
#define B 0

int main(void)
{

#if A
    char alphabet = 'A';
    printf("A = 1");
#endif

#if B
    char alphabet = 'B';
    printf("B = 1");
#endif

    return 0;
}
```



선행처리 후

```c
#include <stdio.h>

int main(void)
{
    char alphabet = 'A';
    printf("A = 1");

    return 0;
}
```

#if, #elif, #else, #endif

ROBIT ROBOT SPORT GAME TEAM

소스파일

```c
#include <stdio.h>

#define MODE 2

int main(void)
{
#if MODE==1
    printf("MODE is 1₩n");
#elif MODE==2
    printf("MODE is 2₩n");
#elif MODE==3
    printf("MODE is 3₩n");
#else
    printf("MODE is %d₩n", MODE);
#endif
    return 0;
}
```

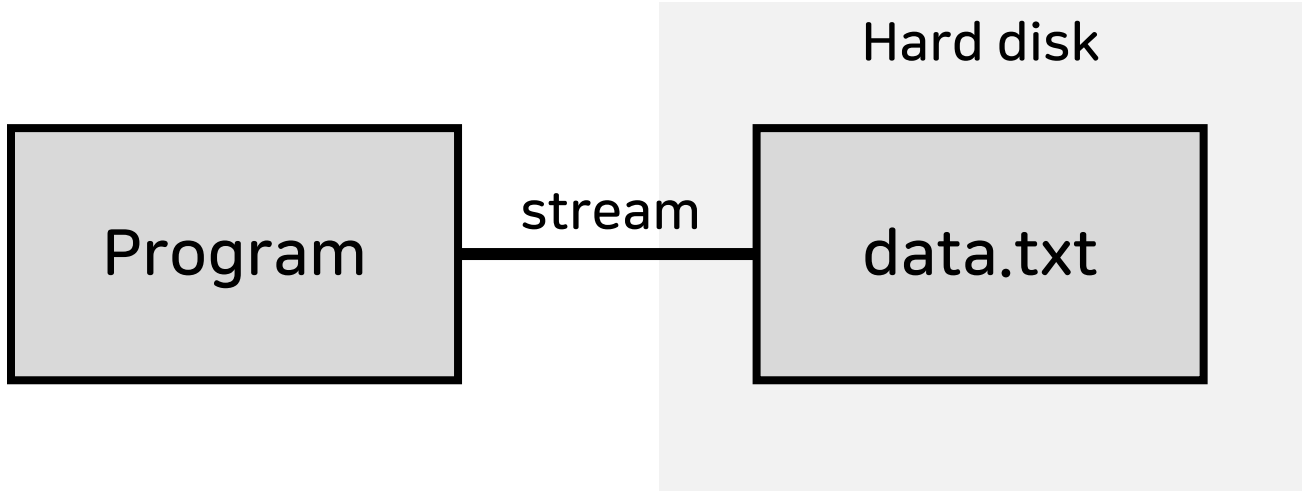
선행처리 후

```c
#include <stdio.h>

int main(void)
{
    printf("MODE is 2₩n");

    return 0;
}
```

ROBIT
ROBOT SPORT GAME TEAM

파일 입출력

stream



프로그램상에서 파일에 저장되어 있는 데이터를 참조하길 원할 때,
프로그램과 참조할 데이터가 저장되어 있는 파일 사이에
데이터가 이동할 수 있는 다리의 역할

=>stream을 형성해야 데이터를 주고 받을 수 있음

파일 입출력

ROBIT
ROBOT SPORT GAME TEAM

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

- 스트림을 형성할 때 호출하는 함수
- 해당 파일과의 스트림을 형성하고 스트림 정보를 FILE구조체에 담아 그 변수의 주소 값 반환
- 실패 시 NULL 포인터 반환

- 첫 번째 인자 : 스트림을 형성할 파일 이름
- 두 번째 인자 : 형성할 스트림의 종류

int fclose(FILE * stream)

- 스트림을 해제하는 함수
- 파일을 닫는 함수
- 성공 시 0 반환

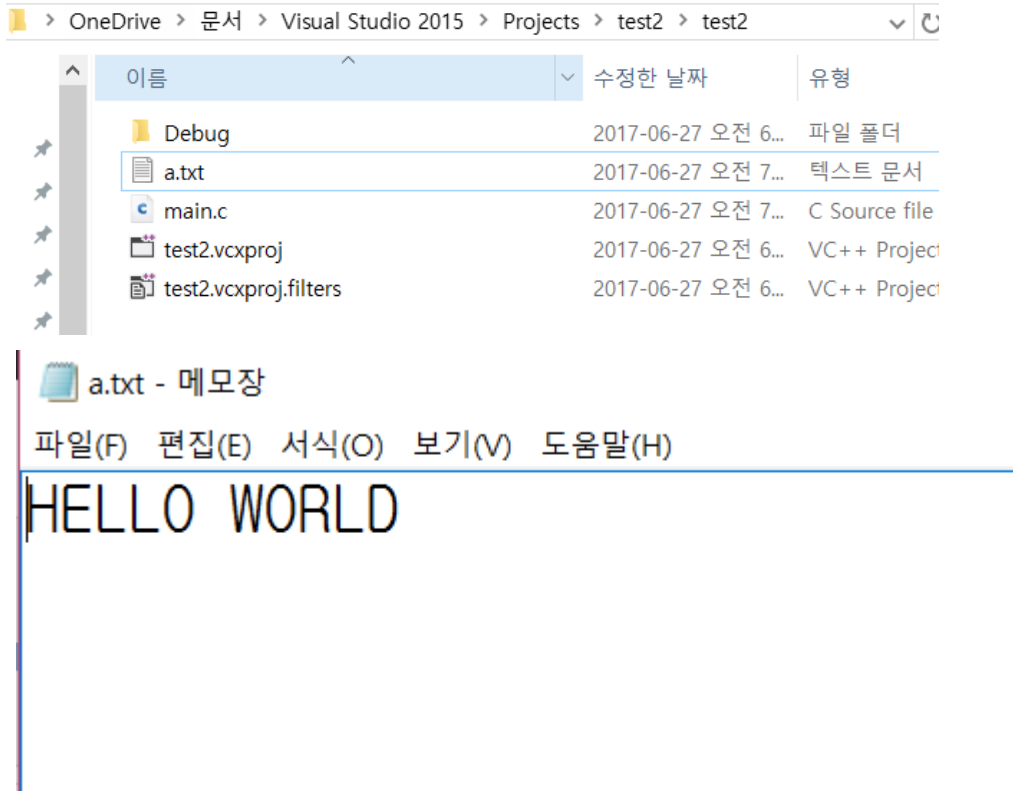
파일 입출력

ROBIT
ROBOT SPORT GAME TEAM

모드	설명
r	읽기만 가능하며 파일이 존재해야 한다. 존재하지 않을 경우 에러가 리턴된다.
w	쓰기만 가능하며 파일을 생성한다. 파일이 존재하면 그 파일을 지우고 생성한다.
a	파일 끝에 추가만 가능하며 파일이 존재하지 않으면 파일을 새로 생성한다.
r+	읽기, 쓰기 모두 가능하며 파일이 존재해야 한다. 존재하지 않을 경우 에러가 리턴된다.
w+	읽기, 쓰기 모두 가능하며 파일을 생성한다. 파일이 존재하면 그 파일을 지우고 생성한다.
a+	파일을 읽어서 파일 끝에 추가만 가능하다. 파일이 존재하지 않으면 파일을 새로 생성한다.
t	파일을 text 모드로 개방한다. 입력 시, CF/LF를 "₩n"으로 자동 변환하고 출력 시에는 "₩n"을 CF/LF로 자동 변환한다. t나 b 생략시 텍스트 모드로 개방된다.
b	파일을 이진(binary) 모드로 개방한다. 개행 문자 "₩n"을 변환없이 그대로 읽고 쓴다.

파일 입출력

```c
int main()
{
    FILE *F;

    F = fopen("a.txt", "w");
    fprintf(F, "HELLO WORLD");
    fclose(F);

    return 0;
}
```



파일입출력

```c
#include <stdio.h>

int main() {
    FILE * fp = fopen("write_test.txt", "w");

    fputc('a', fp);
    fputs("₩nname : 김땡땡", fp);
    fprintf(fp, "₩nage : %d", 20);

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

write_test.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

a
name : 김땡땡
age : 20

파일 입출력

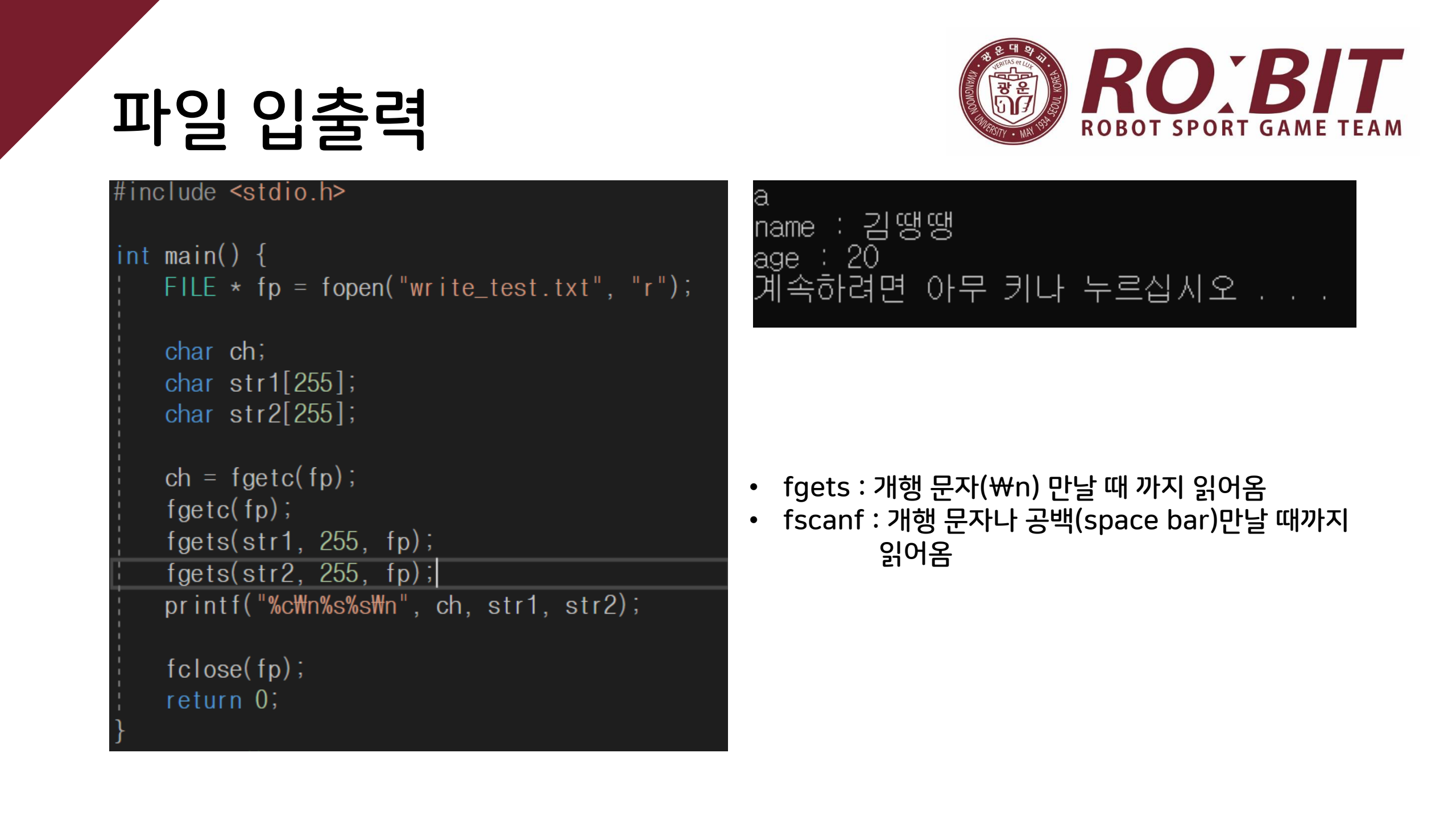
ROBIT
ROBOT SPORT GAME TEAM

```
HELLO
WORLD
계속하려면 아무 키나
```

```c
#include <stdio.h>

int main()
{
    FILE *F;
    char string[15];
    char string2[15];

    F = fopen("a.txt","r");

    fscanf(F,"%s",string);
    fscanf(F,"%s",string2);

    printf("%s\n",string);
    printf("%s\n",string2);
    fclose(F);
    return 0;
}
```

모드	설명
r	읽기만 가능하며 파일이 존재해야 한다. 존재하지 않을 경우 에러가 리턴된다.
w	쓰기만 가능하며 파일을 생성한다. 파일이 존재하면 그 파일을 지우고 생성한다.
a	파일 끝에 추가만 가능하며 파일이 존재하지 않으면 파일을 새로 생성한다.
r+	읽기, 쓰기 모두 가능하며 파일이 존재해야 한다. 존재하지 않을 경우 에러가 리턴된다.
w+	읽기, 쓰기 모두 가능하며 파일을 생성한다. 파일이 존재하면 그 파일을 지우고 생성한다.
a+	파일을 읽어서 파일 끝에 추가만 가능하다. 파일이 존재하지 않으면 파일을 새로 생성한다.
t	파일을 text 모드로 개방한다. 입력 시, CF/LF를 "₩n"으로 자동 변환하고 출력 시에는 "₩n"을 CF/LF로 자동 변환한다. t나 b 생략시 텍스트 모드로 개방된다.
b	파일을 이진(binary) 모드로 개방한다. 개행 문자 "₩n"을 변환없이 그대로 읽고 쓴다.

파일 입출력




```c
#include <stdio.h>

int main() {
    FILE * fp = fopen("write_test.txt", "r");

    char ch;
    char str1[255];
    char str2[255];

    ch = fgetc(fp);
    fgetc(fp);
    fgets(str1, 255, fp);
    fgets(str2, 255, fp);
    printf("%c₩n%s%s₩n", ch, str1, str2);

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

```
a
name : 김땡땡
age : 20
계속하려면 아무 키나 누르십시오 . . .
```

- fgets : 개행 문자(₩n) 만날 때 까지 읽어옴
- fscanf : 개행 문자나 공백(space bar)만날 때까지 읽어옴

과제 공지

1. 제출 기한 : 목요일 12시 까지

2. 제출 형식 : 프로젝트 파일, 보고서(PDF)를 (로빛_18기_수습단원_이름)으로 압축 후
   kwrobit2023@gmail.com 으로 제출
   (보고서는 코드 설명과 실행 화면 첨부, 메일 제목은 C언어_n일차_이름)

3. 채점 기준:
   1) 프로그램의 실행가능 여부
   2) 교육하지 않은 C언어 개념 사용시 감점
   3) 예외처리
   4) 효율적인 코드 작성
   5) 제출 형식

과제

ROBIT
ROBOT SPORT GAME TEAM

[출석부 프로그램]

Student 구조체 : 번호, 이름, 주소-시(ex. Seoul), 성적(4.5점 만점)
구조체 배열 사용(동적할당 이용하여 학생수에 따라 할당크기 변경할 것)

1. 학생 정렬(sort) 기능

   번호순, 이름순, 주소순 성적순 출력 모두 가능 (이름,주소, 성적이 같은 학생이 있을 시, 해당 학생들에 한해 번호순 정렬)

2. 학생 찾기(find) 기능

   번호or주소or성적을 입력하면 해당하는 모든 학생 이름 출력

3. 학생 추가(add), 삭제(remove) 기능

   번호, 이름, 주소, 성적 입력하면 출석부에서 해당 학생 추가/삭제

   중복된 경우 선택하여 삭제

4. 출석부 저장(save) 및 불러오기(load)

   파일입출력을 이용해 출석부를 저장하고 불러옴.

*모든 항목 예외처리(ex 숫자, 문자)

*출석부 기능은 구조체 포인터를 이용한 함수로 제작하여 제출

add
번호:2
이름:jin
주소:seoul
성적:4.1

_____
번호:2
이름:jin
주소:seoul
성적:4.1
_____

add
번호:1
이름:jeongseok
주소: mokpo
성적:4.5

_____
번호:2
이름:jin
주소:seoul
성적:4.1

번호:1
이름:jeongseok
주소: mokpo
성적:4.5
_____

find
기준을 선택하세요
1 번호
2 이름
3 주소
2
찾으려는 이름 입력:jin

_____
번호:2
이름:jin
주소:seoul
성적:4.1
_____

save
저장되었습니다.

>>프로그램 재시작>>

load
불러오기 완료
sort
기준을 선택하세요
1 번호
2 이름
3 주소
1

_____
번호:1
이름:jeongseok
주소: mokpo
성적:4.5

번호:2
이름:jin
주소:seoul
성적:4.1
_____

remove
기준을 선택하세요
1 번호
2 이름
3 주소
3
삭제하려는 주소 입력:seoul

_____
번호:1
이름:jeongseok
주소: mokpo
성적:4.5
_____