

C – 구조체(structure)

struct



구조체의 개념

◆ 구조체는 왜 필요할까?

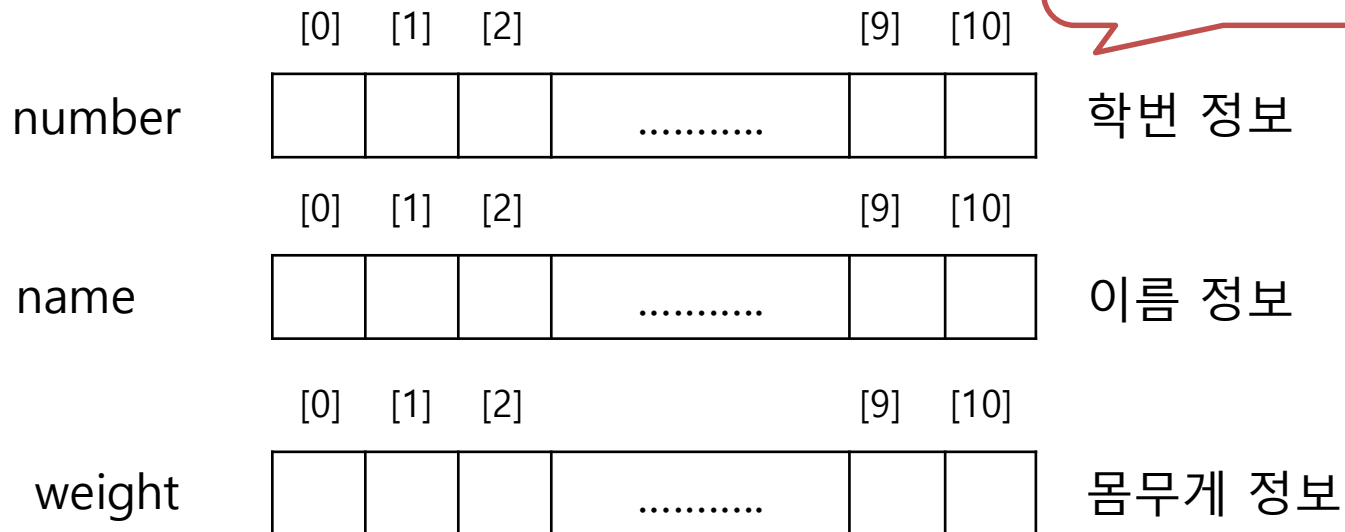
학생 10명의 학번과 이름, 몸무게 정보 저장 – 배열 자료형 이용

```
int number[10];
```

```
char name[20];
```

```
double weight[10];
```

정보가 흩어져서 저장되는 한계 발생!



구조체란 무엇인가?

◆ 구조체(structure)란?

다양한 자료형을 그룹화하여 하나의 변수로 처리할 수 있게 만든 자료형이다.
개발자가 다양한 정보를 저장하기 위해 필요에 따라 생성하는 자료형을 **사용자 정의 자료형** 또는 **구조체**라 한다.

- 구조체 정의

```
struct 구조체이름{  
    자료형 멤버이름;  
};
```

- 구조체 변수 선언

```
struct 구조체이름 변수이름;
```



구조체의 정의 및 사용

◆ 구조체 정의

```
struct Person {  
    //이름, 나이, 키  
    char name[20];  
    int age;  
    float height;  
};
```

※ 멤버 변수는 일반 변수
처럼 초기화 할 수 없음
int age = 0 (x)

- 구조체 객체(변수) 선언

```
struct Person p1
```

- 구조체 객체(변수) 선언과 초기화

```
struct Person p1 = {"알파고", 11, 170.4f}
```



구조체의 정의 및 사용

◆ 구조체 변수 생성 및 사용

```
//구조체 변수 선언
struct Person p1;

//p1.name = "알파고";
strcpy(p1.name, "알파고");
p1.age = 11;
p1.height = 170.5f;

//구조체 변수 선언과 초기화
//struct Person p1 = { "알파고", 11, 165.5f };

printf("이름: %s\n", p1.name);
printf("나이: %d\n", p1.age);
printf("키: %.1f\n", p1.height);
```



구조체 배열

◆ 구조체 배열 – 객체를 여러 개 생성

```
struct Person p[3] = {  
    {"이산", 15, 171.9f},  
    {"한강", 35, 163.3f},  
    {"박봄", 22, 178.4f},  
};  
int i;
```

구조체 배열 선언과
동시에 초기화

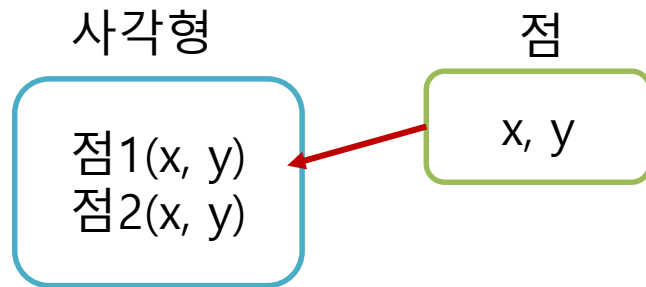
```
//p[0]의 정보  
/*printf("이름: %s\n", p[0].name);  
printf("나이: %d\n", p[0].age);  
printf("키: %.1f\n", p[0].height);*/  
  
for (i = 0; i < 3; i++)  
{  
    printf("이름: %s, 나이: %d, 키: %.1f\n",  
        p[i].name, p[i].age, p[i].height);  
}
```



중첩 구조체

◆ 중첩 구조체

구조체의 멤버 변수가 다른 구조체 자료형인 경우



//점(좌표) 구조체 정의

```
struct Point
{
    int x;
    int y;
};
```

//사각형 구조체 정의

```
struct Rectangle
{
    //Point 구조체 자료형 참조
    struct Point p1;
    struct Point p2;
};
```



중첩 구조체

◆ 중첩 구조체

```
int main()
{
    //두 점을 이용한 직사각형 만들기
    //좌측 상단 좌표, 우측 하단 좌표 생성

    /*struct Rectangle rect;
    rect.p1.x = 1;
    rect.p1.y = 5;

    rect.p2.x = 4;
    rect.p2.y = 2;*/

    struct Rectangle rect = {
        .p1 = {1, 5},
        .p2 = {5, 2}
    };
    int width, height;
```

점 1(1, 5), 점 2(5, 2)
너비:4, 높이: 3



중첩 구조체

◆ 중첩 구조체

```
//좌표 출력
printf("점1(%d, %d), 점2(%d, %d)\n", rect.p1.x, rect.p1.y,
      rect.p2.x, rect.p2.y);

//너비와 높이 계산
width = abs(rect.p2.x - rect.p1.x);
height = abs(rect.p2.y - rect.p1.y);

printf("너비:%d, 높이: %d\n", width, height);

return 0;
}
```



구조체 typedef 키워드 사용

- **typedef struct** 구조체

typedef 를 이용한 구조체 정의하면 구조체 변수 선언시 struct를 생략할 수 있어 코드가 간결해 짐

```
typedef struct {  
    자료형 멤버이름;  
} 구조체이름;
```



```
typedef struct {  
    char name[20],  
    int age;  
    float height;  
} Person;
```

```
Person p1;
```



구조체 typedef 키워드 사용

- typedef struct 구조체

```
#define _CRT_SECURE_NO_WARNINGS //strcpy()
#include <stdio.h>

//Employee 구조체 정의
typedef struct {
    int id;           //사원 아이디
    char name[20];    //사원 이름
    int salary;       //급여
}Employee;
```



구조체 typedef 키워드 사용

- typedef struct 구조체

```
//구조체 변수 선언
Employee e1;

//입력(초기화)
e1.id = 1;
strcpy(e1.name, "이사원");
e1.salary = 3000000;

//Employee e1 = { 1, "이사원", 3000000 };

//정보 출력
printf("사원 ID: %d, 이름: %s, 급여: %d\n",
      e1.id, e1.name, e1.salary);
```



구조체 typedef 키워드 사용

- **typedef struct** 구조체 – 함수 사용

```
//사원 구조체 정의
typedef struct {
    int id;
    char name[20];
    int salary;
}Employee;

//사원 정보 출력
void printInfo(Employee e) {
    printf("사원 ID: %d, 이름: %s, 급여: %d\n",
        e.id, e.name, e.salary);
}
```



구조체 typedef 키워드 사용

- **typedef struct** 구조체 – 함수 사용

```
int main()
{
    //구조체 변수 선언
    Employee emp1;

    //입력
    emp1.id = 1001;
    strcpy(emp1.name, "박상희");
    emp1.salary = 3000000;

    //출력
    printInfo(emp1);
}
```



구조체 typedef 키워드 사용

- typedef struct 구조체 배열

```
//구조체 배열 선언
Employee employees[3] = {
    {1001, "임시연", 2500000},
    {1002, "우상영", 2000000},
    {1003, "이정우", 3000000}
};

printf("===== 사원 명단 =====\n");
for (int i = 0; i < 3; i++) {
    printInfo(employees[i]);
}

return 0;
}
```

```
사원 ID: 1001, 이름: 박상희, 급여: 3000000
===== 사원 명단 =====
사원 ID: 1001, 이름: 임시연, 급여: 2500000
사원 ID: 1002, 이름: 우상영, 급여: 2000000
사원 ID: 1003, 이름: 이정우, 급여: 3000000
```



구조체 typedef 키워드 사용

- 과목의 점수 관리

```
#define _CRT_SECURE_NO_WARNINGS //strcpy()
#include <stdio.h>
#include <string.h>

//Subject 구조체
typedef struct {
    char subjectName[20]; //과목명
    int scorePoint;        //점수
}Subject;

//과목 출력 함수
void printInfo(Subject subject) {
    printf("과목명: %s, 점수: %d\n",
        subject.subjectName, subject.scorePoint);
}
```



구조체 typedef 키워드 사용

- 과목의 점수 관리

```
int main()
{
    //구조체 변수 국어, 수학, 영어 생성
    Subject kor, math, eng;
    int sum;
    float avg;

    //과목 입력
    strcpy(kor.subjectName, "국어");
    kor.scorePoint = 92;

    strcpy(math.subjectName, "수학");
    math.scorePoint = 82;

    strcpy(eng.subjectName, "영어");
    eng.scorePoint = 86;
```



구조체 typedef 키워드 사용

- 과목의 점수 관리

```
//평균 계산
sum = kor.scorePoint + math.scorePoint + eng.scorePoint;
avg = (float)sum / 3;

//과목 출력
printInfo(kor);
printInfo(math);
printInfo(eng);
printf("평균 점수: %.1f\n", avg);

return 0;
}
```

```
과목명 : 국 어 , 점수 : 92
과목명 : 수 학 , 점수 : 82
과목명 : 영 어 , 점수 : 86
평균 점수 : 86.7
```



구조체 typedef 키워드 사용

- 과목의 점수 관리

```
//구조체 배열 생성
Subject subjects[SIZE] = {
    {"국어", 92},
    {"수학", 82},
    {"영어", 86}
};
int i, sum = 0;
float avg;

//구조체 배열 검색(1번 위치)
printf("%s, %d\n", subjects[1].subjectName,
        subjects[1].scorePoint);
printf("-----\n");
```



구조체 typedef 키워드 사용

- 과목의 점수 관리

```
//평균 계산
for (int i = 0; i < SIZE; i++) {
    sum += subjects[i].scorePoint;
}
avg = (float)sum / SIZE;

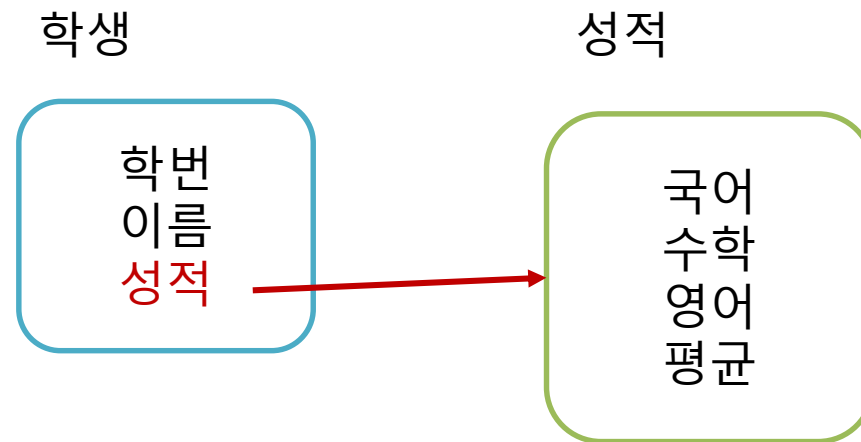
//과목 출력
for (i = 0; i < SIZE; i++) {
    printInfo(subjects[i]);
}
printf("평균 점수: %.1f\n", avg);
```



성적 관리 프로그램

◆ 성적 관리 프로그램

- 학생 구조체가 성적 구조체 자료형을 사용함



성적 관리 프로그램

◆ 성적 관리 프로그램

```
학번 입력 : 101
이름 입력 : 박상희
국어점수 입력 : 90
수학점수 입력 : 80
영어점수 입력 : 73
학번 입력 : 102
이름 입력 : 오상식
국어점수 입력 : 80
수학점수 입력 : 84
영어점수 입력 : 75
학번 입력 : 103
이름 입력 : 한강
국어점수 입력 : 95
수학점수 입력 : 80
영어점수 입력 : 92

===== 성적표 =====
학번   이름   국어   수학   영어   평균
101    박상희  90    80    73    81.0
102    오상식  80    84    75    79.7
103    한강    95    80    92    89.0

===== 과목별 평균점수 =====
국어 평균 : 88.3
수학 평균 : 81.3
영어 평균 : 80.0
```



성적 관리 프로그램

◆ 성적 관리 프로그램

```
#define _CRT_SECURE_NO_WARNINGS //strcpy()
#include <stdio.h>
#define SIZE 3          //구조체 배열의 크기
#define NAME_LEN 20     //이름의 크기

typedef struct{
    int kor;    //국어
    int math;   //수학
    int eng;    //영어
    float avg;  //평균
}Subject;

typedef struct{
    int number;    //학번
    char name[NAME_LEN]; //이름
    Subject subject; //과목 구조체 변수(참조)
}Student;
```



성적 관리 프로그램

◆ 성적 관리 프로그램

```
//학생의 정보
void printInfo(Student st){
    printf("%d\t%s\t%d\t%d\t%d\t%.1f\n", st.number, st.name,
        st.subject.kor, st.subject.math, st.subject.eng, st.subject.avg);
}

int main()
{
    Student students[SIZE]; //구조체 배열 생성
    int i;
    int subject_sum[3] = {0, 0, 0}; //국어, 수학, 영어 총점
    double subject_avg[3]; //국어, 수학, 영어 평균

    //사용자 입력
    for (i = 0; i < SIZE; i++){
        printf("학번 입력: ");
        scanf("%d", &students[i].number);

        printf("이름 입력: ");
        scanf("%s", students[i].name);
    }
}
```



성적 관리 프로그램

◆ 성적 관리 프로그램

```
printf("국어점수 입력: ");
scanf("%d", &students[i].subject.kor);

printf("수학점수 입력: ");
scanf("%d", &students[i].subject.math);

printf("영어점수 입력: ");
scanf("%d", &students[i].subject.eng);

//개인별 평균 계산
students[i].subject.avg = (students[i].subject.kor +
    students[i].subject.math + students[i].subject.eng) / 3.0;
}

//과목별 총점 계산
for (i = 0; i < SIZE; i++){
    subject_sum[0] += students[i].subject.kor; //국어 총점
    subject_sum[1] += students[i].subject.math; //수학 총점
    subject_sum[2] += students[i].subject.eng; //영어 총점
}
```



성적 관리 프로그램

◆ 성적 관리 프로그램

```
//과목별 평균 계산
subject_avg[0] = (double)subject_sum[0] / SIZE; //국어 평균
subject_avg[1] = (double)subject_sum[1] / SIZE; //수학 평균
subject_avg[2] = (double)subject_sum[2] / SIZE; //영어 평균

//학생 정보 출력
printf("\n===== 성 적 표 =====\n");
printf("학번\t이름\t국어\t수학\t영어\t평균\n");
for (i = 0; i < SIZE; i++){
    printInfo(students[i]);
}

printf("\n===== 과목별 평균 점수 =====\n");
printf("국어 평균: %.11f\n", subject_avg[0]);
printf("수학 평균: %.11f\n", subject_avg[1]);
printf("영어 평균: %.11f\n", subject_avg[2]);

return 0;
}
```



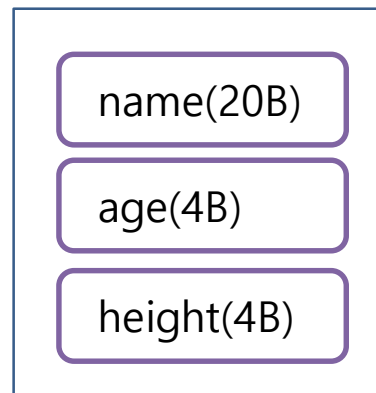
구조체 포인터 변수

- 구조체 포인터 사용

구조체 변수가 닷(.) 연산자를 사용하는 반면, 구조체의 포인터 변수는 참조 연산자(->)를 사용한다.

```
typedef struct
{
    //이름, 나이, 키
    char name[20];
    int age;
    float height;
}Person;
```

Person



ptr(8B)

Person의 주소저장

```
Person p1 = { "알파고", 11, 171.9f };
Person* ptr = &p1; //구조체 포인터 선언
```

ptr->name, ptr->age, ptr->height



구조체 포인터 변수

- 구조체 포인터 사용

```
typedef struct {  
    int no;  
    char name[20];  
    int age;  
}Hero;  
  
int main()  
{  
    //구조체 변수 선언과 초기화  
    Hero p1 = { 1, "고담덕", 39 };  
  
    //점(.) 연산자로 속성에 접근함  
    printf("번호: %d, 이름: %s, 나이: %d\n",  
           p1.no, p1.name, p1.age);  
}
```



구조체 포인터 변수

- 구조체 포인터 사용

```
//구조체 포인터 선언과 초기화
Hero p2 = { 2, "이순신", 54 };
Hero* ptr = &p2;

//화살표(->) 연산자로 속성에 접근함
printf("번호: %d, 이름: %s, 나이: %d\n",
      ptr->no, ptr->name, ptr->age);

return 0;
}
```

```
번호 : 1, 이름 : 고담덕, 나이 : 39
번호 : 2, 이름 : 이순신, 나이 : 54
```



날짜와 시간 구현 구조체

- 현재 날짜와 시간 표시하기

```
#define _CRT_SECURE_NO_WARNINGS //localtime()
#include <stdio.h>
#include <time.h>

int main()
{
    time_t ct = time(NULL); //현재 시간
    struct tm* now = localtime(&ct); //현재 날짜와 시간 저장

    printf("현재 날짜: %d. %d. %d.\n", now->tm_year + 1900,
        now->tm_mon + 1, now->tm_mday);

    printf("현재 시간: %d : %d : %d.\n", now->tm_hour,
        now->tm_min, now->tm_sec);
}
```

```
현재 날짜: 2025. 5. 20.
현재 시간: 11 : 19 : 26.
오전 11시 19분 26초
현재 요일: 2
화요일입니다.
```



날짜와 시간 구현 구조체

- 현재 날짜와 시간 표시하기

```
//12시각제 사용
int hour = (now->tm_hour > 12) ? now->tm_hour - 12 : now->tm_hour;
char* ampm = (now->tm_hour < 12) ? "오전" : "오후";
printf("%s %d시 %d분 %d초\n", ampm, hour, now->tm_min, now->tm_sec);

//요일 (0-일, 1-월, 2-화...)
printf("현재 요일: %d\n", now->tm_wday);

//요일 출력
char* days[] = {"일", "월", "화", "수", "목", "금", "토"};
int idx = now->tm_wday;

printf("%s요일입니다.\n", days[idx]);

return 0;
}
```



구조체 포인터 변수

- 구조체 포인터 변수

```
typedef struct {  
    char name[20]; //과일명  
    int quantity; //수량  
    char* brand; //브랜드  
}Fruit;  
  
int main()  
{  
    char *brand[] = { "청송사과", "상주사과", "음성사과" };  
  
    Fruit f = { "사과", 100, brand[0]};  
    Fruit* p;
```



구조체 포인터 변수

- 구조체 포인터 변수

```
puts("--- 구조체 변수로 접근 ---");
printf("브랜드: %s\n", f.name);
printf("수량: %d\n", f.quantity);
f.brand = "상주사과";
printf("과일 종류: %s\n", f.brand);

p = &f; //구조체 주소 저장
puts("\n--- 구조체 포인터로 접근 ---");
printf("브랜드: %s\n", p->name);
printf("수량: %d\n", p->quantity);
f.brand = "상주사과";
printf("과일 종류: %s\n", p->brand);
```

--- 구조체 변수로 접근 ---

브랜드: 사과

수량: 100

과일 종류: 상주사과

--- 구조체 포인터로 접근 ---

브랜드: 사과

수량: 100

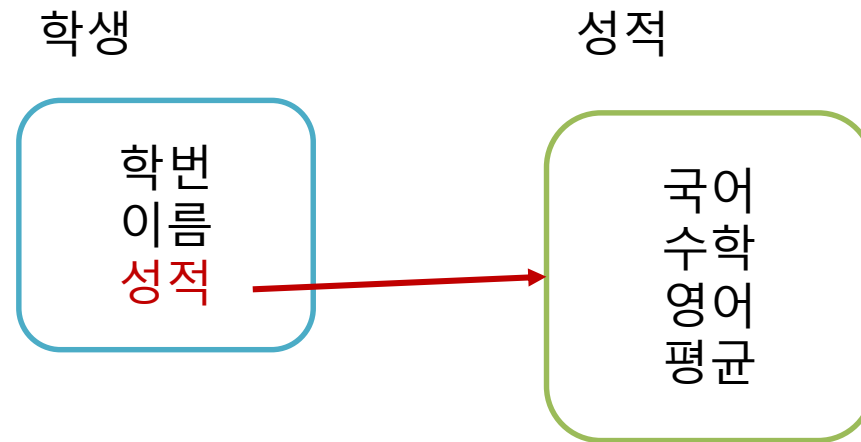
과일 종류: 상주사과



성적 관리 프로그램

◆ 성적 관리 프로그램

- 학생 구조체가 성적 구조체 자료형을 참조함



```
void printInfo(Student* st) {  
    printf("%d\t%s\t%d\t%d\t%d\t%.1f\n",  
        st->number,  
        st->name,  
        st->subject.kor,  
        st->subject.math,  
        st->subject.eng,  
        st->subject.avg);  
}
```



성적 관리 프로그램

- 헤더 파일 분리 – Student.h

```
#ifndef STUDENT_H //조건부 컴파일 블록
#define STUDENT_H //매크로 이름 정의(구조체 중복 오류 막는 용도)

#include <stdio.h>
#define NAME_LEN 20 //이름의 크기
#define SIZE 3 //사람 수

typedef struct {
    int kor; //국어
    int math; //수학
    int eng; //영어
    float avg; //평균
}Subject;

typedef struct {
    int number; //학번
    char name[NAME_LEN]; //이름
    Subject subject; //과목 구조체 변수(참조)
}Student;

void printInfo(Student* st); //학생의 정보 출력 함수 선언
#endif
```



성적 관리 프로그램

- 헤더 파일 분리 – Student.c

```
//헤더파일 구현부
#include "Student.h"

void printInfo(Student* st) {
    printf("%d\t%s\t%d\t%d\t%d\t%.1f\n",
        st->number,
        st->name,
        st->subject.kor,
        st->subject.math,
        st->subject.eng,
        st->subject.avg);
}
```



성적 관리 프로그램

- 헤더 파일 분리 – Main.c

```
#define _CRT_SECURE_NO_WARNINGS
#include "Student.h"

int main()
{
    Student students[SIZE]; //구조체 배열 생성
    int i, j;
    int subject_sum[3] = { 0, 0, 0 }; //국어, 수학, 영어 총점
    double subject_avg[3]; //국어, 수학, 영어 평균
    const char* subjects[3] = { "국어", "수학", "영어" };

    //사용자 입력
    for (i = 0; i < SIZE; i++) {

        printf("학번 입력: ");
        scanf("%d", &students[i].number);

        printf("이름 입력: ");
        scanf("%s", students[i].name);
    }
}
```



성적 관리 프로그램

- 헤더 파일 분리 – Main.c

```
//과목 점수 포인터 배열
int* scores[3] = { &students[i].subject.kor,
                  &students[i].subject.math,&students[i].subject.eng };

for (j = 0; j < SIZE; j++) {
    printf("%s점수 입력: ", subjects[j]);
    scanf("%d", scores[j]);
}

//개인별 평균 계산
students[i].subject.avg = (float)(students[i].subject.kor +
                                   students[i].subject.math + students[i].subject.eng) / 3;
}

//과목별 총점 계산
for (i = 0; i < SIZE; i++) {
    subject_sum[0] += students[i].subject.kor; //국어 총점
    subject_sum[1] += students[i].subject.math; //수학 총점
    subject_sum[2] += students[i].subject.eng; //영어 총점
}
```



성적 관리 프로그램

- 헤더 파일 분리 – Main.c

```
//과목별 평균 계산
for (i = 0; i < SIZE; i++) {
    subject_avg[i] = (double)subject_sum[i] / SIZE;
}

//학생 정보 출력
printf("\n===== 성 적 표 =====\n");
printf("학번\t이름\t국어\t수학\t영어\t평균\n");
for (i = 0; i < SIZE; i++) {
    printInfo(&students[i]);
}

printf("\n===== 과목별 평균 점수 =====\n");
for (i = 0; i < SIZE; i++) {
    printf("%s 평균: %.11f\n", subjects[i], subject_avg[i]);
}

return 0;
}
```

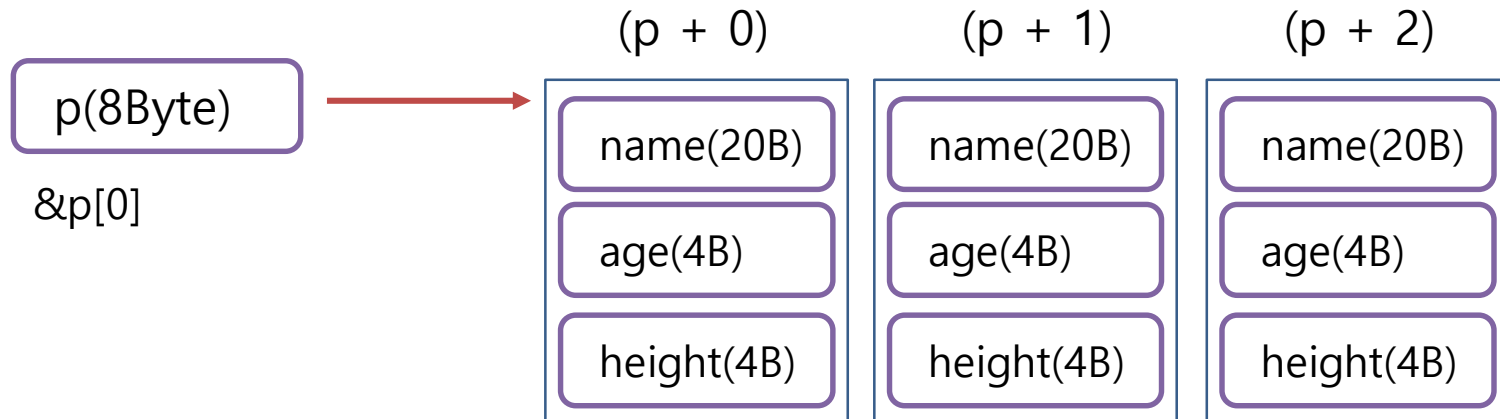


구조체 배열 동적 할당

- 구조체 포인터 사용 필요성

1. 함수의 매개 변수로 구조체 복사(전달)시 용량과 시간을 줄일 수 있음 (모든 포인터의 크기는 8byte 이다)
2. 동적 메모리 할당이 가능하다.

```
Person* p = (Person*)malloc(sizeof(Person) * 3);
```



구조체 배열 동적 할당

- 구조체 배열 동적 할당

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

```
//Employee 구조체 정의
```

```
typedef struct {
```

```
    int id;
```

```
    char name[20];
```

```
    int salary;
```

```
}Employee;
```

```
//사원 정보 출력 함수
```

```
void displayInfo(Employee* e, int len) {
```

```
    for (int i = 0; i < len; i++) {
```

```
        printf("사원 ID: %d, 이름: %s, 급여: %d\n",
```

```
            (e + i)->id, (e + i)->name, (e + i)->salary);
```

```
    }
```

```
}
```

```
사원 ID: 1001, 이름: 강사원, 급여: 2000000
사원 ID: 1002, 이름: 박대리, 급여: 3000000
사원 ID: 1003, 이름: 한과장, 급여: 4000000
```



구조체 배열 동적 할당

- 구조체 배열 동적 할당

```
int main()
{
    //구조체 배열 동적 할당
    Employee* emp;
    emp = (Employee *)malloc(sizeof(Employee) * 3);
    if (emp == NULL) {
        printf("동적 메모리 할당에 실패했습니다.\n");
        exit(1);
    }

    //emp 1명 생성
    emp->id = 1001;
    strcpy_s(emp->name, sizeof(emp->name), "강사원");
    emp->salary = 2000000;
}
```



구조체 배열 동적 할당

- 구조체 배열 동적 할당

```
//emp 2명 생성
(emp + 1)->id = 1002,
strcpy_s((emp + 1)->name, sizeof(emp->name), "박대리");
(emp + 1)->salary = 3000000;

//emp 3명 생성
(emp + 2)->id = 1003,
strcpy_s((emp + 2)->name, sizeof(emp->name), "한과장");
(emp + 2)->salary = 4000000;

displayInfo(emp, 3); //displayInfo() 호출

free(emp); //메모리 해제

return 0;
}
```



구조체 배열 동적 할당

- 구조체 배열 동적 할당 – 배열 인덱스로 접근

```
//사원 정보 출력 함수
void displayInfo(Employee* e, int len) {
    for (int i = 0; i < len; i++) {
        printf("사원 ID: %d, 이름: %s, 급여: %d\n",
            e[i].id, e[i].name, e[i].salary);
    }
}

int main()
{
    //구조체 배열 동적 할당
    Employee* emp;
    emp = (Employee*)malloc(sizeof(Employee) * 3);
    if (emp == NULL) {
        printf("동적 메모리 할당에 실패했습니다.\n");
        exit(1);
    }
}
```



구조체 배열 동적 할당

- 구조체 배열 동적 할당 – 배열 인덱스로 접근

```
//emp 1명 생성
emp[0].id = 1001;
strcpy_s(emp[0].name, sizeof(emp[0].name), "강사원");
emp[0].salary = 2000000;

//emp 2명 생성
emp[1].id = 1002;
strcpy_s(emp[1].name, sizeof(emp[1].name), "박대리");
emp[1].salary = 3000000;

//emp 3명 생성
emp[2].id = 1003;
strcpy_s(emp[2].name, sizeof(emp[2].name), "한과장");
emp[2].salary = 4000000;


displayInfo(emp, 3); //displayInfo() 호출

free(emp); //메모리 해제
return 0;
}
```



실습 문제 – 구조체 배열 동적 할당

Book 구조체를 만들어서 책 2권을 출력하는 프로그램을 작성하세요

실행결과 

```
번호 : 201, 제목 : 모두의 C언어  
번호 : 202, 제목 : 채식주의자
```

