컴프 #6 수업에서 다루어지는 것들: 구조체 고급

다음과 같이 구조체 명 student를 정의하자.

```
struct student {
  int id;
  int midterm;
  int final;
};
```

■ **구조체 포인터**

```
struct student aStudent;
struct student *sp = &aStudent;
```

□ **-> 연산자**: 구조체 포인터 sp를 사용하여 구조체 변수에 접근할 때

연습2-1: 포인터 sp를 사용하여 구조체 멤버 각각에 값 설정	연습2-2: 포인터 sp를 사용하여 구조체 멤버의 값을 입력 printf("학번 입력:"); scanf("%d",); printf("중간고사 입력:"); scanf("%d",); printf("학기말고사 성적 입력:"); scanf("%d",);
연습2-3: 포인터 sp를 사용하여 구조체 멤버의 값을 출력한다. printf("학번: %d 중간: %d 학기말: %d\n",)	
연습2-4: 중간고사와 학기말고사 성적을 더해 출력한다. printf("학번 %d의 총점은 %d이다\",)	

□ **-> 대신 *와 .(dot)를 사용할 수있다.**

sp->id은 (*sp).id와 같다.

여기서 괄호를 사용하는 이유는? 답:_____

■ **구조체 배열과 구조체 포인터**

```
struct student sList[3]; // sList[0], sList[1], sList[2]는 각각 구조체 변수

// sp가 sList 배열의 처음(sList[0])을 가르치게 하려면


// sp를 사용하여 첫번째 배열 원소(sList[0])의 멤버들을 출력
printf("%d ",                    )
printf("%d ",                    )
printf("%d ",                    );

// sp가 다음 원소(즉, sList[1])를 가르키게한다.
sp++;
// sp를 사용하여 배열 원소(sList[1])의 멤버들을 출력
printf("%d ",                    )
printf("%d ",                    )
printf("%d ",                    );

// sp가 다음 원소(즉, sList[2])를 가르키게한다.
sp++;
// sp를 사용하여 배열 원소(sList[2])의 멤버들을 출력
printf("%d ",                    )
printf("%d ",                    )
printf("%d ",                    );
```

● **구조체 변수를 선언하는 2가지 방법**

방법1 구조체 명(지난 시간에 배운)	방법2: **typedef**를 사용하여 구조체 타입

■ **구조체명 선언(정의)**
struct person {// person은 구조체명.
char name[10];
int age;
};
 	
■ **구조체 변수 선언**
struct person p; // 구조체 변수 p 선언
	■ **구조체타입 선언(정의)**
	typedef struct person { //person은 구조체명(생략가능)
	char name[10];
	int age;
	} Person; // Person은 구조체타입.
	
	■ **구조체 변수 선언**
	Person p; // 구조체 변수 p 선언

Q: 어느것이 더 better?

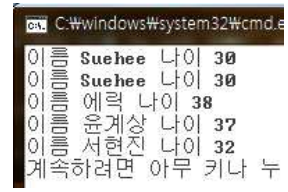
방법1을 지난 시간에 익혔으니 **오늘은 방법2를 이용하자.(이것이 더 편리)**

■ **구조체를 함수 매개변수로 전달**

- 구조체: 구조체를 전달(복사)(call by value)
- 구조체 포인터: 구조체의 주소를 전달(call by reference)

□ main함수에서 함수가 호출될 때의 인수를 잘 살펴보라. 아래의 세 함수의 정의부분의 매개변수를 살펴보고 적절하게 함수의 body를 완성하라.

□ printPerson1과 printPerson2는 어떤 것이 더 효율적인가?




```
#include <stdio.h>

typedef struct {
   char name[10];
   int age;
} Person; // Person은 구조체타입.

void printPerson1(           ) // call by value
{
   printf("이름 %s 나이 %d\n",                    );
}

void printPerson2(Person *p) // call by reference :  p는 포인터
{
   printf("이름 %s 나이 %d\n",                    );
}

void printPeople(Person *p, int size)  // call by reference: p는 포인터 : Person p[]라고 써도 됨
{
  int i;


}

int main(void)
{
   Person aPerson = {"Suehee", 30};
   Person people[3] = {{"에릭", 38}, {"윤계상", 37}, {"서현진", 32}};

   //아래의 호출에서 인수들을 잘 살펴보자.
   printPerson1(aPerson);
   printPerson2(&aPerson);

   printPeople(people, 3);
}
```

□ **연습 #1**

위에서 아래와 같은 함수를 생각하자. 출력 결과는? 답: _____

tenYearsAfter함수 호출후 aPerson의 나이를 40으로 하려면 어떻게 하면 되는가?

```
void tenYearsAfter(Person p)
{
  p.age += 10;
}
int main(void)
{
   Person aPerson = {"Suehee", 30};

   tenYearsAfter(aPerson);
   printPerson1(aPerson); // printPerson2(&aPerson);도 결과는 같다
}
```

구조체를 함수의 매개변수로 넘길 때 구조체포인터를 사용하는 2가지 이유

1. 매개변수로 값을 복사하지 않고(시간이 걸리므로) 주소값을 전달(포인터 사용)(이전 페이지의 printPerson1과 printPerson2의 비교)
2. 함수안에서 인수로 전달되는 구조체의 내용이 바뀌어야 할 때(위의 연습 #1)

➔ 그러므로 구조체는 주소값을 넘기고 포인터로 받는 것이 좋다.

□ **연습 #3**(이제 매개변수에 구조체 포인터를 사용해서 함수를 정의하자). totalScore1과 totalScore2는 같은 일을 하는 함수이다. 둘의 차이(구현에 있어서의)를 잘 살펴보라.

```
typedef struct {
   int midterm;
   int final;
} Score;

void printScore(Score *p)
{
    printf("중간고사 성적은 %d\n",                );
    printf("학기말고사 성적은 %d\n",                );
}

Score totalScore1(Score *p1, Score *p2)
{




}

void totalScore2(Score *p1, Score *p2, Score *total)
{


}

int main(void) {
    Score s1, s2, tempScore;

    s1.midterm = 50; s1.final = 100;
    s2.midterm = 70; s2.final = 70;

    // totalScore1과 totalScore2의 호출 및 함수 정의를 잘 살펴보라.
    tempScore = totalScore1(&s1, &s2); // 중간고사, 학기말고사를 각각 더해서 구조체 tempScore를 만든다
    printScore(&tempScore);

    totalScore2(&s1, &s2, &tempScore);
    printScore (&tempScore);
}
```

LAB 11 구조체 고급

■ LAB10_0 revisited

□ LAB10_0_2(구조체 변수와 구조체 포인터)

LAB10_0_0 의 프로그램에 아래와 같이 student 타입의 변수에 대한 포인터를 선언하고, 이 포인터를 이용하여 가)와 같은 일을 하도록 프로그램을 수정하라.(*와 .(dot)연산자를 사용하라))

```
struct student *sp = &aStudent;
```

□ LAB10_0_3(구조체 배열과 구조체 포인터)

LAB10_0_1 의 프로그램에 아래와 같이 student 타입의 변수에 대한 포인터를 선언하고, 이 포인터를 이용하여 나)와 같은 일을 하도록 프로그램을 수정하라. (*와 .(dot)연산자를 사용하라))

```
struct student *sp = s;
```

■ LAB10_1(revisited)

라) (구조체 포인터와 구조체변수)구조체 customer를 가르키는 포인터 변수 cp를 선언하고 이 cp에 나)의 aCustomer 변수가 저장된 메모리의 (첫째) 주소값을 가르키게 하자. 그런후 cp를 이용하여 나)의 실행결과와 같이 출력을 하라. 이때 -> 연산자를 사용하라.

마) (구조체 포인터와 구조체배열)위의 cp에 라)의 cArray가 저장된 메모리의 (첫째) 주소값을 가르키게 하자. 그런 후 cp를 이용하여 위의 실행결과와 같이 출력을 하라. 이때 -> 연산자를 사용하라

■ LAB11_0 아래의 프로그램을 수행시키면서 물음에 답해보자.

```
#include <stdio.h>
int main(void)
{
	struct foo_t {
		int x;
		int y;
		char c;
		char c2;
	};

	struct foo_t data;
	struct foo_t *ptr = &data, *ptr2;

	ptr2 = ptr;
	ptr2++;

	// 나)
	printf("sizeof data is = %d\n", sizeof(data));   //a)


	printf("data의 첫 주소값은 %d\n", &data);   //b)

	printf("data.x 첫 주소값은 %d\n", &data.x);  //c)


	printf("ptr이 가르키는 주소값은 %d\n", ptr);  //d)

	printf("ptr2가 가르키는 주소값은 %d\n", ptr2);//e)

	return;
}
```

가) 변수 data의 메모리 크기를 예상하고 메모리 저장상태를 그려보시오. 그리고, a)에서 sizeof 연산자를 사용하여 출력한 결과가 예상한 것과 일치하는지 확인 해 보시오. b)c)의 결과가 같은 이유를 생각해보시오. d)e)의 결과보고 그 차이를 이해하시오.

나) 구조체 foo_t를 아래와 같이 수정한 후 가)에서 수행한 작업을 다시 해 보시오. 어떤 변화가 있는지, 왜 그런 변화가 있었는지 밝혀보시오.

```
struct foo_t {
	char c;
	int x;
	char c2;
	int y;
};
```

■ LAB11_1(구조체변수와 함수)

□ LAB11_1_1(구조체 변수를 함수의 매개변수로 전달)

아래의 순서로 단계적으로 프로그램을 작성하라.

1) 중간고사성적(midterm)과 학기말고사성적(final)을 멤버로 갖는 구조체 타입 Score를 정의하고 중간고사 성적과 학기말고사 성적을 출력하는 printScore함수를 정의하고 아래의 main함수를 실행
2) 두 성적을 비교하여 성적이 좋은(중간고사와 학기말고사를 더한 총점이 큰)수를 반환하는 biggerScore함수를 정의하고 main함수를 실행
3) 두 성적의 중간고사를 더한 값, 학기말고사를 더한 값을 멤버로 갖는 구조체 변수를 반환하는 totalScore함수를 정의하고 main함수를 실행
4) 매개변수로 주어진 중간고사와 학기말고사의 성적을 갖는 구조체 변수를 반환하는 createScore함수를 정의하고 main함수를 실행

```
#include <stdio.h>
#include <stdlib.h>
// 1) typedef으로 구조체 타입 Score 정의하라.



void printScore(Score s) //2)
{
    printf("중간고사 성적은 %d\n",           );
    printf("학기말고사 성적은 %d\n",           );
}
Score biggerScore(Score s1, Score s2) //3)
{



}
Score totalScore(Score s1, Score s2) //4)
{



}
Score createScore(int m, int f) //5)
{



}
int main(void) {

    Score s1, s2, s3;
    s1.midterm = 50; s1.final = 100;
    s2.midterm = 70; s2.final = 70;

    printf("1)----------------------------\n");
    printScore(s1);
    printScore(s2);

    printf("2)----------------------------\n");
    printf("둘 중 성적이 좋은 점수:\n");
    printScore(biggerScore(s1, s2)           );

    printf("3)----------------------------\n");
    printf("두 성적의 총 합:\n");
    printScore(totalScore(s1, s2)             );
    printf("4)----------------------------\n");
    s3 = createScore(99, 99);
    printScore(s3);
}
```

```
C:₩windows₩system32₩cmd.exe
1)----------------------------
중간고사 성적은 50
학기말고사 성적은 100
중간고사 성적은 70
학기말고사 성적은 70
2)----------------------------
둘중 성적이 좋은 점수:
중간고사 성적은 50
학기말고사 성적은 100
3)----------------------------
두 성적의 총 합:
중간고사 성적은 120
학기말고사 성적은 170
4)----------------------------
중간고사 성적은 99
학기말고사 성적은 99
계속하려면 아무 키나 누르십시오 . .
```

□ LAB11_1_2(구조체 포인터를 함수의 매개변수로 전달)

구조체의 경우 그 크기가 클 때 데이터 복사에 비용이 많이 들 수있으므로, 매개변수 전달시 포인터를 사용하는 것이 좋다!

위의 LAB11_1_1에서 함수의 매개변수로 사용된 구조체 변수를 모두 구조체 포인터로 바꾸어 프로그램을 다시 작성하라.

변경되는 함수의 원수는 아래와 같다. createScore함수는 변경되지 않는다.

.

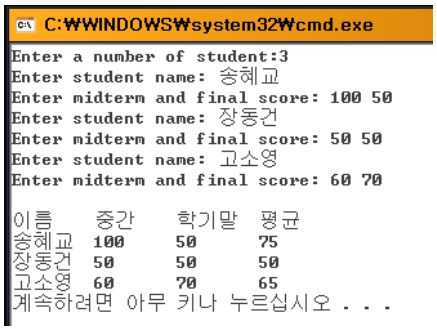
```
void printScore(Score *p) {…} // 매개변수 변경
Score *biggerScore(Score *p1, Score *p2) {…} // 반환타입 변경, 매개변수 변경
void totalScore(Score *p1, Score *p2, Score *tp) {…} // 반환타입 변경, 매개변수 한 개 추가
Score createScore(int m, int f) {…} // 변경되지 않음!
```

// main함수는 아래와 같이 변경된다.

```
int main(void) {

    Score s1, s2, temp;

    s1.midterm = 50;    s1.final = 100;
    s2.midterm = 70;    s2.final = 70;

    printf("1)----------------------------\n");
    printScore(&s1);
    printScore(&s2);

    printf("2)----------------------------\n");
    printf("둘중 성적이 좋은 점수:\n");
    printScore(biggerScore(&s1, &s2));

    printf("3)----------------------------\n");
    printf("두 성적의 총 합:\n");
    totalScore(&s1, &s2, &temp);
    printScore(&temp);

    printf("4)----------------------------\n");
    temp = createScore(99, 99);
    printScore(&temp);
}
```

■ **LAB11_2 아래 지시에 따라 프로그램을 작성해 본다.**

아래와 같이 student 구조체를 정의하고 typedef 를 이용하여 Student 라는 새로운 타입을 정의하는 부분을 채운후 프로그램을 실행시켜보면 아래와 같다.




```
#include <stdio.h>
struct student {
	char name[20];
	int midterm;
	int final;
	int average;
};
//  typedef 를 사용하여 Student 를 정의

void printStudent(Student aStudent)
{
	printf("%s\t", aStudent.name);
	printf("%d\t%d\t%d\n", aStudent.midterm, aStudent.final, aStudent.average);
}

int main(void)
{
	Student s[40];
	int num, i;

	printf("Enter a number of student:");
	scanf("%d", &num);

	for (i = 0; i < num; i++)
	{
		printf("Enter student name: ");		//a)
		scanf("%s", s[i].name);			//b)
		printf("Enter midterm and final score: ");	//c)
		scanf("%d %d", &s[i]. midterm, &s[i].final);	//d)
	}
	for (i = 0; i < num; i++)
		s[i].average = (s[i].midterm + s[i].final) / e)

	printf("\n 이름\t 중간\t 학기말\t 평균\n");
	for (i = 0; i < num; i++)
		printStudent(s[i]);
}
```

□ **LAB11_2_0(구조체 포인터를 매개변수로)**위의 printStudent 함수는 구조체변수를 매개변수로 받았다. 이를 고려해서, 구조체포인터를 매개변수로 하도록 printStudent 함수를 수정하라. 물론 main 도 수정하라.

□ **LAB11_2_1 아래의 지시대로 프로그램을 수정하라. 함수의 매개변수로 구조체 포인터를 사용하라.**

- 위의 LAB11_2_0 코드에서 a)b)c)d)의 네 문장을 아래의 함수 호출로 대치하고, 이 함수의 정의를 추가하라.
  readStudentScore(...)
- e)부분을 아래의 함수 호출로 대치하고, 이 함수의 정의를 추가하라.
  calculateStudentAverage(...)

HW 11 구조체 고급

■ **HW11_1 (구조체 포인터) 아래와 같은 구조체포인터 sp 를 사용하여 HW10_1 의 프로그램을 다시 작성하라.**

```
struct student s[3];
struct student *sp = s;
```

주의사항 및 힌트

- 여기서는 s[..]의 표현은 더 이상 사용하지 않고, sp 와 -> 연산자만을 사용하라.
- 그러나 중간중간에 sp = s;는 계속 사용할 수도(혹은 사용 해야할 수도) 있다(이것은 힌트이기도 하다)

LABHW 11 구조체 고급(추가)

■ **HW13_1 (구조체 일반)**(난이도 중상)
아래와 같이 student와 cClass 구조체를 정의하고 typedef를 이용하여, 각각 Student, CClass라는 새로운 타입을 정의하고 아래의 프로그램을 실행시켜보라. 어떤 결과가 나오는가? 프로그램을 이해하여야 이 숙제를 할수있다. 주어진 프로그램은 입력받은 학생의 중간, 학기말성적과 그 평균을 구하여 프린트하는 프로그램이다.

```
#include <stdio.h>
struct student {
        char name[20];
        int midterm;
        int final;
        int average;
};
// typedef사용하여 Student 정의

struct cClass {
        int num;
        Student s[40];
};
// typedef사용하여 CClass 정의

void printStudent(Student *sp)
{
        printf("%sWt", sp->name);
        printf("%dWt%dWt%dWn", sp->midterm, sp->final, sp->average);
}

int main(void)
{
        CClass cp;
        int i;
        Student all = {"Total", 0, 0, 0};

        printf("Enter a number of student:");
        scanf("%d", &cp.num);

        //a)
        for (i = 0; i < cp.num; i++)
        {
                printf("Enter student name: ");
                scanf("%s", cp.s[i].name);
                printf("Enter midterm and final score: ");
                scanf("%d %d", &cp.s[i].midterm, &cp.s[i].final);
        }

        //b)
        for (i = 0; i < cp.num; i++)
                cp.s[i].average = (cp.s[i].midterm + cp.s[i].final) / 2;

        printf("Wn이름Wt 중간Wt 학기말Wt 평균Wn");
        for (i = 0; i < cp.num; i++)
                printStudent(&cp.s[i]);
}
```

```
C:₩WINDOWS₩system32₩cmd.exe
Enter a number of student:3
Enter student name: 박수희
Enter midterm and final score: 100 50
Enter student name: 장동건
Enter midterm and final score: 50 100
Enter student name: 고소영
Enter midterm and final score: 50 60

이름    중간    학기말  평균
박수희  100     50      75
장동건  50      100     75
고소영  50      60      55
계속하려면 아무 키나 누르십시오 . . .
```

위의 프로그램에서 a)부분과 b)부분을 readStudentScore 와 calculateStudentAverage 함수를 각각 사용하여 다시 작성하라. 각 함수의 원형 및 설명은 다음과 같다.

```
// Student 구조체 변수에 name, midterm, final을 읽는다.
void readStudentScore(Student *sp);

// Student 구조체 변수에 average 값을 계산한다. average = (midterm + final) / 2
void calculateStudentAverage(Student *sp)
```

■ **HW11_2** 위의 HW13_1과 같은 일을 하는 프로그램을 아래의 main함수로 대치하여 다시 작성하라.
즉, 새로운 함수 readClass, calculateClassAverage, printClass를 정의해야한다.

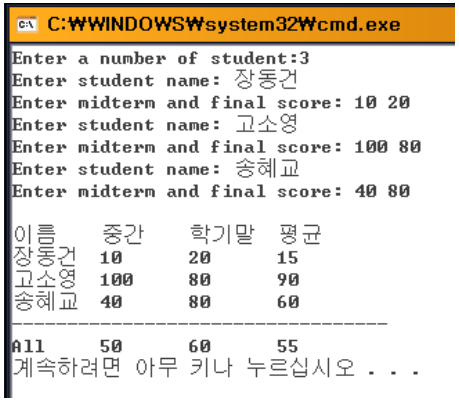
```
int main(void)
{
        CClass cp;

        Student all = {"Total", 0, 0, 0};

        printf("Enter a number of student:");
        scanf("%d", &cp.num);

        readClass(&cp); // 앞 코드의 a)부분 대치

        calculateClassAverage(&cp); // 앞 코드의 b)부분 대치

        printf("Wn이름Wt중간Wt학기말Wt평균Wn");
        printClass(&cp);
}
```

■ **Challenge11_1(구조체 일반)** (난이도 중상)

위의 HW13_2에 다음을 추가하라.
각 학생들의 중간, 학기말, 평균들에 대한 각각의 평균을 출력하려한다.
아래와 같은 실행결과를 갖는 프로그램을 작성하라.

- calculateAll()은 cp가 가리키는 CClass 구조체변수 내의 Student타입 변수들의 midterm, final, 그리고 average의 평균을 구하여 pAll이 가리키는 Student 구조체변수에 저장하는 함수이다.
  함수의 프로토타입은 다음과 같다.

  void calculateAll(CClass *cp, Student *pAll);




```
int main(void)
{
	CClass cp;

	Student all = {"All", 0, 0, 0};

	printf("Enter a number of student:");
	scanf("%d", &cp.num);

	readClass(&cp);

	calculateClassAverage(&cp);

	printf("\n 이름\t 중간\t 학기말\t 평균\n");
	printClass(&cp);

	calculateAll(&cp, &all);
	printf("---------------------------------\n");
	printStudent(&all);
}
```

■ **Challenge11_2(구조체 일반)** (난이도 중상)

위의 HW13_1에서 함수 calculateAll()을 아래와 같이 수정하여 같은 결과가 나오도록 프로그램하라.

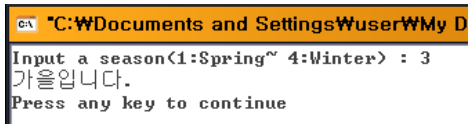
```
Student calculateAll2(CClass *cp)
{
    Student pAll = {"All", 0, 0, 0};

    ...




    return pAll;
}
```

main함수의 추가부분

```
    all = calculateAll2(&cp)
    printStudent(&all)
```

■ **HW11_3** (enum) (난이도 하)

enum을 이용하여 아래의 실행결과가 나오도록 프로그램을 작성하라. 이는 교재 p530의 enum_usable2.c와 거의 비슷한 문제입니다. 교재의 코드를 참조시오.



PROJECT 03(2) 비디오 관리 프로그램(구조체 고급)

■ Project3_1(구조체, 구조체 배열과 구조체 포인터)

앞의 문제를 발전시켜서 비디오 관리 프로그램에 **대여 기능을 추가**하려 한다.
대여정보는 고객 id와 Video 제목을 포함한다.

즉, Video 관리 프로그램은 다음의 기능을 갖는다.

1) 보유 Video들을 출력
2) Video 구입
3) title로 Video 찾기
4) **Video 대여(추가)**
5) **대여 정보들을 출력(추가)**

Note:
- Video 대여만 구현해본다. (Video 반납은 구현하지 않는다)
- 아래처럼 단계적으로 구현해본다.
  - 단계1: 1)을 구현하여 실행
  - 단계2: 2)를 구현하여 실행 -> 1)로 확인
  - 단계3: 3)을 구현하여 실행
  - 단계4: 4)를 구현하여 실행 -> 5)를 구현하여 실행하여 확인

이번에는 각 기능들을 **함수를 이용하여 구현**한다.
각 기능에 대한 설명 및 해당 함수의 원형은 다음과 같다.

typedef를 이용하여 VideoInfo와 RentInfo를 다시 타입정의하고 이를 구조체 변수 선언시 사용하였다.

1) 보유 Video들을 출력.
- 재고대장(videoList)에 있는 Video 정보를 출력한다.

```
void printAllVideo(VideoInfo videoList[], int videoCount);
```

2) Video 구입
- 제목, 수량을 입력 받아서 재고대장(videoList)에 저장한다.
- Video 개수(videoCount)을 한 개 증가시킨다.
- 여기서 videoCount을 가르키는 포인터가 매개변수임에 주목하자. 왜인가?

```
void purchaseVideo(VideoInfo videoList[], int *videoCountPtr, char *title, int qty);
```

3) title로 Video 찾기
- 제목을 입력 받은 후 아래의 함수를 이용하여
  재고대장(videoList)내에 이 Video가 있으면 그 Video가 있는 인덱스를 반환하고
  없으면 -1을 반환한다.

```
int searchVideoByTitle(VideoInfo videoList[], int videoCount, char *title);
```

4) Video 대여(*가장 어렵다!*)
- 이 함수를 간단히 하기 위해서 항상 대여 가능한 Video에 대한 대여가 시도 된다고 가정한다.
- Video 제목과 고객 id를 입력받고
  SearchVideoByTitle 함수를 사용하여 해당 Video의 인덱스를 찾는다.
  고객 id와 Video 제목을 대출대장(rentList)에 저장한다.
  rentCount를 한 개 증가시키고, 해당 video의 수량은 한 개 감소시킨다

```
void rentVideo(_____________________________________________________________);
```

5) 대여 정보들을 출력
- 현재 대여중인 정보(고객 id와 video 제목)들을 출력한다.

```
void printAllRent(_____________________________________________________________);
```

```
#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)들을 저장
   int custId ; // 고객 id : 1, 2, 3
   char title[MAX_CHAR] ; // Video 제목
} RentInfo;

void printAllVideo(VideoInfo videoList[], int videoCount) {//구현}
void purchaseVideo(VideoInfo videoList[], int *videoCountPtr, char *title, int qty) {//구현}
int searchVideoByTitle(VideoInfo *videoList, int videoCount, char *title) {//구현}
void rentVideo(…) {//구현}
void printAllRent(…) {//구현}

int main(void)
{
    int videoCount = 5;
    VideoInfo videoList[MAX_VIDEO] = {{"BeforeSunrise", 1}, {"BeforeSunset", 3},
                        {"BeforeMidnight", 5}, {"Casablanca", 7}, {"EdgeOfTomorrow", 9} };
    int rentCount = 0; // 현재 대출 건수는 0임
    RentInfo rentList[MAX_CUST];

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

    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);
    }
}
```

```
C:₩windows₩system32₩cmd.exe
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 1
Video제목       수량
----------------------
BeforeSunrise   1
BeforeSunset    3
BeforeMidnight  5
Casablanca      7
EdgeOfTomorrow  9
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 2
Enter video 제목: BeginAgain
Enter video 수량: 10
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 1
Video제목       수량
----------------------
BeforeSunrise   1
BeforeSunset    3
BeforeMidnight  5
Casablanca      7
EdgeOfTomorrow  9
BeginAgain      10
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 3
Enter video 제목: BeforeSun
그런 비디오는 없습니다.
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 3
Enter video 제목: BeforeSunrise
대여 가능합니다.
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 4
Enter video 제목: BeforeSunrise
Enter 고객 id: 2
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 5
 고객id  Video제목
----------------------
2       BeforeSunrise
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 1
Video제목       수량
----------------------
BeforeSunrise   0
BeforeSunset    3
BeforeMidnight  5
Casablanca      7
EdgeOfTomorrow  9
BeginAgain      10
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): 3
Enter video 제목: BeforeSunrise
다 대여중입니다.
1(All Video 출력), 2(구입), 3(검색), 4(대여), 5(All 대여정보 출력), 6(종료): _
}
```

■ Project3_2(구조체 포인터)(Challenge)

위의 문제에서 아래의 함수를 다음과 같이 수정하고. 이에 관련된 부분들(rentVideo함수, main함수)을 수정하여 같은 실행결과를 내도록 하라.

3) title로 Video 찾기
- 함수 내에서 제목을 입력받아서 아래의 함수를 이용하여

재고대장(videoList)내에 이 Video가 있으면 그 Video를 가르키는 포인터를 반환하고 없으면 NULL을 반환한다.

```
VideoInfo *searchVideoByTitle(VideoInfo videoList[], int videoCount, char *title);
```

질문: 위의 수정된 함수에서 Video에 대한 포인터, 즉 구조체에 대한 포인터를 반환하는 이유는 무엇인가? 구조체 타입을 반환하여서도 문제를 풀 수있는가?