2018년도 1학기 소프트웨어입문설계 기말고사 (1 / 2교시)

반: (월수반 / 화목반) 학번: _____ 이름: _____

- 1교시는 여러분이 특정 목표를 달성하기 위한 적절한 C 코드 요소를 작성할 수 있는지를 주로 묻습니다. 여러분이 작성한 답안은 우리 수업 표준 환경을 기준으로 채점됩니다.
- 답안 칸 밖에 적은 글씨들은 절대 채점에 반영되지 않습니다(답안은 마련된 칸 안에 적어 주세요).
  답안 칸 안에 적은 모든 글씨들은 채점에 반영됩니다(답안 칸에 이상한거 적지 말아 주세요).
  답안 칸에 있는 글씨를 지울 수 없다면 그 글씨에 ~~가로줄을 두 개~~ 그어 주세요.
- 여러분이 적은 글자는 사람인 강사가 읽습니다. 모호한 글자를 적는 경우 다른 곳에 적은 동일한 글자들의 필체를 감안하여 인식을 수행할 예정이나, 여러분의 의도가 명백히 강사에게 전달되지 않을 가능성이 있습니다. 이 점을 감안하여 몇몇 혼동하기 쉬운 글자들은 꼭 또박또박 적어 주세요.
- 일부 문제는 답안 적을 곳을 두 개 마련해 두었습니다. 위 칸을 쓰다가 다시 적으려는 경우 줄 긋고 아래 칸을 사용해 주세요. 만약 두 칸 모두 글씨가 적혀 있다면 무조건 아래 칸의 내용을 답안으로 간주합니다.
- 1교시 시험은 50분 동안 진행되며 만점은 50점입니다.

(1 ~ 3) 다음 **선언**들을 먼저 읽고, 문항별 목표들을 달성할 수 있는 수식을 각각 적어 주세요.
해당 목표를 달성하는 것이 불가능하다면 '불가능'이라 적어 주세요.

번호	내용
1	`int number;`
2	`int arr[10];`
3	`int func();`

1. number에 담긴 값을 number 자리에 담는 수식 (2점, 1분)

2. number에 담긴 값을 arr의 0번째 칸에 담는 수식 (3점, 2분)

3. number에 담긴 값을 func 자리에 담는 수식 (4점, 3분)

(다음 페이지에서 계속됩니다)

(4 ~ 8) 다음 **선언**들을 먼저 읽고, 문항별 목표들을 달성하기에 가장 적합한 **선언**을 각각 적어 주세요. 해당 목표를 달성하는 것이 불가능하다면 '불가능'이라 적어 주세요.

번호	내용
1	`int number;`
2	`double *ptr;`
3	`void func();`

주의사항:

- 시험 쉬울 줄 알았지요? 여기서부터는 꼭 문제 내용을 차근차근 읽어 봐야 해요.
- 새 **선언**을 적을 때 새 이름은 자유롭게 정해도 좋아요.
- 몇몇 문항에 적어 둔 표현 '동일한 형식'은 수식 측면이 아닌 **선언** 측면에서 해석해야 해요.
  (예: **배열**과 **포인터** 변수는 둘 다 [ ]연산자 써서 다룰 수 있지만 서로 다른 형식이에요)
- 여기서는 initializer는 신경쓰지 않아도 좋아요.

4. 위의 1번 **선언**에 의해 **정의**된 한 칸에 대한 **포인터** 값을 담을 수 있는 변수 **선언** (3점, 2분)

5. 위의 2번 **선언**에 의해 **정의**된 한 칸에 대한 **포인터** 값을 각각 담을 수 있는 세 칸짜리 **배열 선언** (4점, 3분)

6. 위 **선언**이 적절히 존재할 때 수식 `func()`를 계산한 결과값을 담을 수 있는 변수 **선언** (3점, 2분)

7. 4번 문항의 답으로 적은 이름과 동일한 형식을 return값의 형식으로 갖는 함수 **선언** (3점, 4분)

8. 5번 문항의 답으로 적은 이름과 동일한 형식을 return값의 형식으로 갖는 함수 **선언** (3점, 7분!)

(다음 페이지에서 계속됩니다)

(9 ~ 10) 아래의 예시 코드 구조를 잘 보고, 문항별 주어진 목표를 달성하기 위한 함수 정의를 각각 적어 주세요.

```
// 여러분이 적는 함수 정의는 이 부분에 위치하게 될 예정

int main()
{
    int arr[2];

    arr[0] = put_five();
    fill_with_three(&arr[1]);

    return 0;
}
```

주의사항:

- 이번에는 약간 쉬운 문항이 나왔어요. 푼 다음 꼼꼼히 점검해 보고 다음 문제로 넘어가면 좋을 듯 해요.
- 함수 정의 밖에는 아무 것도 적으면 안 되고, 한 문항에 대해 함수 정의를 여러 개 적어도 안 돼요.
- 함수 정의 안에 **선언**도 문장도 아닌 다른 것(예: `#include` 등)을 적으면 0점으로 간주돼요.
- 제약 조건이 있음에도 성공적으로 목표를 달성할 수 있는지를 묻는 것이므로 위 내용을 꼭 지켜 주세요.

9. 목표: 5를 return하는 함수 `put_five()` 정의하기 (4점, 3분)

10. 목표: 인수로 받은 `int` **포인터** 값을 통해 그 칸(**포인터**를 '따라가면' 있는 int 칸)에 3을 담아 주는 함수 `fill_with_three()` 정의하기(인수 이름은 자유롭게 정해요) (4점, 4분)

다음 페이지는 조금 어려우니 마음의 준비를 해 두는 게 좋겠어요.

(다음 페이지에서 계속됩니다)

11. 아래와 같이 두 파일을 작성하여 하나의 프로그램을 만들 예정입니다. 여러분은 이들 중 data.c(왼쪽 것)을 먼저 만들어 두는 역할을 담당하고 있습니다. 여러분의 목표는, main.c를 작성할 다른 프로그래머가 data.c에 **선언 / 정의**된 int 변수 number에 담긴 값을 읽을 수는 있되 수정(변경)할 수는 없도록 만드는 것입니다. 우리가 배운 범위 내에서, 이러한 목표를 달성하는 방법은 크게 두 가지가 있었습니다. 두 방법 모두를 각각 사용하여 data.c의 내용을 적절하게 구성해 주세요. (총 17점, 15분)

주의사항:

- (매우 주의)동일한 목표에 대한 서로 다른 답을 두 번 적어야 해요.
- (매우 주의)문제에서 요구하는 '서로 다른 답'은, 완전히 다른 specifier를 각각 사용하여 접근하는 것을 말해요.
  - 아래에 적혀 있는 **선언** int number;에 서로 다른 specifier를 하나 붙이는 것
- 어떤 방법은 단순히 **선언**을 고치는 것뿐만 아니라 추가적인 **선언** 및 **정의**(함수 정의 포함)를 필요로 해요.
- main.c를 작성하는 프로그래머가 절대 정상적인 방법으로 int 변수 number에 담긴 값을 변경할 수 없어야 해요.
- main.c를 작성하는 프로그래머는 자신이 목표를 달성(값 얻기)하기 위해 필요한 **선언**을 적을 능력을 갖추고 있어요. (여러분은 그 프로그래머를 trust해도 좋아요)

방법 1)

data.c

```
// 목표: 아래의 빈 칸에 적절한 specifier 적기
_____ int number;

// 목표: 필요하다면 추가적인 선언 / 정의 적기
```

main.c

```
// 주의: 여기엔 아무 것도 적으면 안 됨!

// 적절한 선언을 적을 예정

int main()
{
   // 적절한 문장(수식)을 적을 예정
}
```

방법 2)

data.c

```
// 목표: 아래의 빈 칸에 적절한 specifier 적기
_____ int number;

// 목표: 필요하다면 추가적인 선언 / 정의 적기
```

main.c

```
// 주의: 여기엔 아무 것도 적으면 안 됨!

// 적절한 선언을 적을 예정

int main()
{
   // 적절한 문장(수식)을 적을 예정
}
```

(1교시 마지막 페이지예요. 여유가 조금 있다면 내가 작성한 코드를 구경하며 복습을 해 두는 것을 권장해요)