「Korea Polytechnic University」
프로그래밍 과제 노트

2019-01학기

교과목 담당교수	컴퓨터공학부 박정민
학과	컴퓨터공학과
학번	2017152049
이름	정하림

프로그래밍 과제 노트 목차

1. 온라인 강의 요약정리

1-1 Part1-7장. 함수 (02페이지 ~ 04페이지)

2. 심화문제 분석 및 풀이

2-1 심화문제 (05페이지 ~ 08페이지)

3. 자기성찰 및 평가

3-1 6주차-자기성찰 및 평가 (9페이지)

1. 온라인 강의 요약정리

1-1 Part1-7장. 함수

■ 함수와 함수의 형태

- 특정 작업을 수행하는 코드의 집합
- 기능을 구현해놓고 원하는 시점에서 호출하여 사용할수 있다.
- 표준 라이브러리가 제공하는 함수와 사용자가 만든 사용자 정의 라이브러리 함수가 있다.
- 코드에 안정성, 재사용성, 응집력이 향상되고 복잡성이 낮아지며 에러의 수정이 쉬워진다.
- 함수의 입, 출력여부에 따라 있으면 1 없으면(void) 0으로 총 4가지 형태가 존재
- main문 이전에 정의하여 호출할 수 있고 또는 main문 이전에 선언만 해놓고 이후에 정의하여 호출시킬 수 있다. 함수 목록을 직관적으로 볼 수 있기 때문에 후자가 일반적이다.
- 함수의 이름은 함수의 주소이다.
-

```
//함수의 선언
반환형 함수명 (매게변수);
예시) int sum(int num1, num2); // 11 형태


//함수의 정의
반환형 함수명 (매게변수)
{
        실행문;
        return 반환할 값;
}
예시) int sum(int num1, num2)
{
        return num1 + num2;
}
```

■ 변수의 종류와 범위

- 지역변수
  - 함수의 내부에서 사용하는 변수로 함수 내부에서만 사용이 가능하다.
  - 자신이 만들어진 {}중괄호 블록(지역)을 벗어나거나 함수가 종료되면 사라진다.
  - 초기화 작업을 하지 않으면 쓰레기값이 들어가게 된다.
  - 조건문이나 반복문의 중괄호 내부나 매개변수(입력변수)에서도 주로 사용한다.
  - C언어에서는 포인터를 통해 지역 밖에서도 주소로 참조하여 사용할 수 있다.

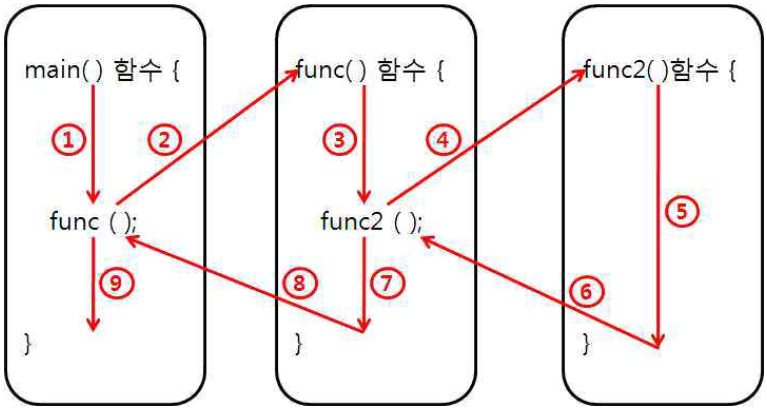
- 전역변수
  - 함수 밖에 선언된 변수로 해당 파일에 모든 함수에서 사용이 가능하다.
  - 초기화 작업을 하지 않아도 0으로 초기화된다.
  - 한번 만들면 프로그램 종료 시까지 사라지지 않아 메모리 낭비가 심하다.
  - 여러 군데에서 값을 참조할 수 없도록 static을 통해 정적 전역변수를 만들 수 있다

- 정적변수
  - 변수 앞에 static키워드를 붙이면 정적변수가 된다.
  - 전역변수처럼 자동으로 0으로 초기화되고 프로그램 종료시 까지 사라지지 않는다.
  - 초기화를 딱 1번만 수행하므로 선언과 동시에 값을 할당해 주어야한다.

- 외부변수
  - 변수 앞에 extern키워드를 붙이면 외부변수가 된다.
  - 외부파일에 선언된 전역변수를 참조하는 변수이다.
  - 외부변수로 선언하면 자동으로 다른 파일에서 같은 이름의 변수를 찾아 값을 가져온다.

- 레지스터 변수
  - 변수 앞에 register키워드를 붙여 만들 수 있다.
  - 레지스터 변수는 변수에 대한 접근속도를 높이기 위해 CPU 안의 레지스터에 변수를 생성한다.
  - 지역변수로만 선언이 가능하다.
  - 메모리를 거치지 않기 때문에 처리속도가 빠르다.
  - cpu 내부 레지스터 개수의 한계에 따라 코드 최적화가 발생하면 자동으로 지역변수로 할당된다.

■ 함수의 작동원리

- 함수는 수행중인 main문과 별도로 위치해 있는 코드의 집합이기 때문에 호출되면 함수코드가 있는 부분으로 이동하여 실행된다. 따라서 함수가 반환되면 원위치로 복귀된다.
- 그림같은 순서로 함수가 진행된다.
- 

■ 재귀함수

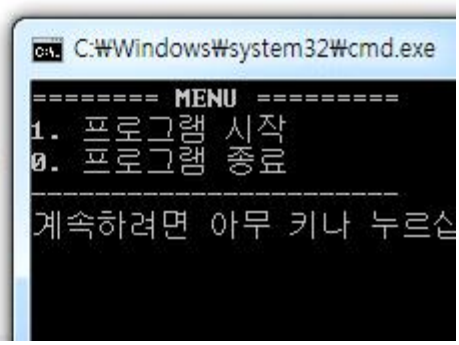
- 자기 자신을 호출하는 함수이며 실행도중 직접, 간접적으로 자기자신을 호출하는 것을 재귀호출이라고 한다.
- 시간과 메모리공간의 효율이 저하될수 있어 신중히 개발해야 한다.
- 선언을 1번만 하는 정적변수의 특성을 이용하여 같이 쓰이기도 한다.
- 무한대로 반복시키면 의미가 없으므로 재귀호출 함수를 만들 때 반드시 조건문을 활용하여 탈출구를 만들어주어야 한다.

2. 심화문제 중점 문제 분석, 디버깅

2-1 심화 분석

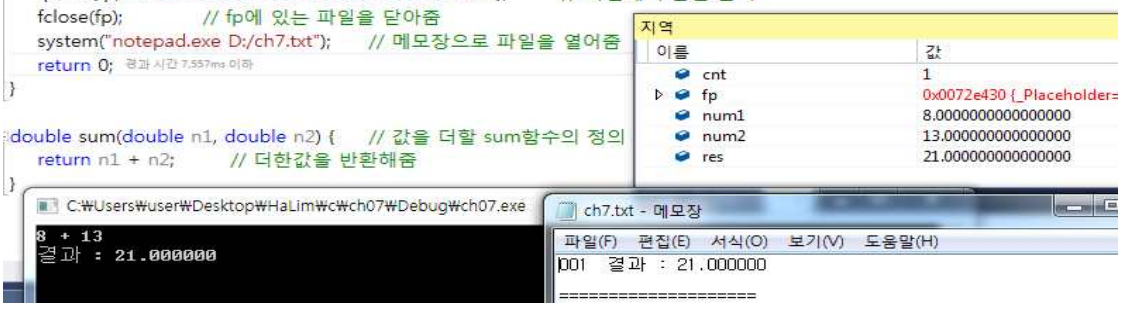
■ **printf 함수를 사용해 메뉴를 출력해주는 함수 void menu()를 만들어 메뉴를 출력하세요.**

```c
#include <stdio.h>

void menu(void);      // 00형태로 메뉴 함수를 선언

int main() {
menu();               // 메뉴함수를 호출
return 0;
}

void menu() {         // 메뉴함수의 정의부분
    // 프로그램 메뉴를 출력
    printf("========= MENU ==========₩n");
    printf("1. 프로그램 시작₩n");
    printf("0. 프로그램 종료₩n");
    printf("----------------------₩n");
}
```



■ **두 수를 입력받아 더한 값을 리턴하는 함수 double sum(double, double)을 정의하고 결과를 출력하는 프로그램을 작성하세요.**

```c
#include <stdio.h>
#pragma warning(disable:4996)

double sum(double n1, double n2); // 11형태의 double형 함수 sum 선언

int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch7.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    double num1 = 0, num2 = 0, res;     // 숫자를 입력받을 num1, num2 결과를 입력받을 res 선언
    int cnt = 0;                        // 레코드 번호 cnt 선언
    scanf("%lf + %lf", &num1, &num2);   // 합산식 입력
    res = sum(num1, num2);    // 결과를 res에 넣어줌
    fprintf(stdout, "결과 : %lf₩n", res);      // 결과를 모니터에 출력
    fprintf(fp, "%03d  ", ++cnt);    // 파일에 레코드번호 출력
    fprintf(fp, "결과 : %lf₩n", res);          // 결과를 파일에 출력


    fprintf(fp, "₩n====================₩n₩n");      // 파일에 구분선 출력
    fclose(fp);          // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch7.txt");     // 메모장으로 파일을 열어줌
    return 0;
}

double sum(double n1, double n2) {    // 값을 더할 sum함수의 정의
    return n1 + n2;       // 더한값을 반환해줌
}
```



■ **사칙연산에 관한 함수를 모두 정의하고 메뉴 함수와 결합해 사칙연산을 수행하는 프로그램을 작성하세요.**

```c
#include <stdio.h>
#pragma warning(disable:4996)
// 함수를 미리 선언해놓는 선언부
void menu(void);
double sum(double n1, double n2);
double min(double n1, double n2);
double mult(double n1, double n2);
double div(double n1, double n2);

int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch7.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    double num1 = 0, num2 = 0, res;         // 숫자를 입력받을 num1, num2 결과를 입력받을 res 선언
    int cnt = 0;                            // 레코드 번호 cnt 선언
    char op, num = 1;                   // 연산자를 받을 op와 num

    while (1) {         // 무한반복문
        menu();      // 메뉴 출력함수 호출
        fprintf(stdout, "메뉴번호를 입력하세요 : ");         // 메뉴 입력 안내메시지
        scanf("%d", &num);                                  // 메뉴 번호 입력
        if (num == 1) {                                     // 메뉴 번호를 확인하여 프로그램 시작
            fprintf(stdout, "식을 입력하세요 : ");           // 식 입력 안내메시지
            scanf("%lf %c %lf", &num1, &op, &num2);     // 식 입력
            switch (op)     // 연산자를 통해 분기시킴
            {
            case '+':     res = sum(num1, num2);    break;
            case '-':     res = min(num1, num2);    break;
            case '*':     res = mult(num1, num2);   break;
            case '/':     res = div(num1, num2);    break;
            default:
                fprintf(stdout, "수식오류!₩n");     // 수식이 잘못 입력되면 다시입력하도록 화면에 메시지 출력
                continue;                           // 반복문 아래를 생략시켜줌
            }
            // 결과 문구 출력
            fprintf(stdout, "결과 : %.1lf₩n₩n", res);

            fprintf(fp, "%03d  ", ++cnt);
            fprintf(fp, "결과 : %.1lf₩n₩n", res);
        }
        else if (num == 0) {                        // 메뉴 번호를 확인하여 프로그램 종료
            // 종료 문구 출력
            fprintf(stdout, "프로그램을 종료합니다.₩n₩n");
            fprintf(fp, "%03d  ", ++cnt);
            fprintf(fp, "프로그램을 종료합니다.₩n");
            break;
        }
        else                                        // 메뉴에 원치 않는 입력이 들어왔을 때
        {
            fprintf(stdout, "다시 입력해주세요₩n₩n");
        }

    }

    fprintf(fp, "₩n====================₩n₩n");       // 파일에 구분선 출력
    fclose(fp);             // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch7.txt");     // 메모장으로 파일을 열어줌
    return 0;

}
void menu() {           // 메뉴를 출력해주는 함수 정의
    printf("========= MENU =========₩n");
    printf("1. 프로그램 시작₩n");
    printf("0. 프로그램 종료₩n");
    printf("----------------------₩n");
}

// 각 연산을 담당하는 함수들의 정의
double sum(double n1, double n2)  { return n1 + n2; }
double min(double n1, double n2)  { return n1 - n2; }
double mult(double n1, double n2) { return n1 * n2; }
double div(double n1, double n2)  { return n1 / n2; }
```

지역

이름	값
cnt	3
fp	0x004ce430 {_Placeholder=0xdddddddd }
num	0 '\0'
num1	3.0000000000000000
num2	7.0000000000000000
op	47 '/'
res	0.42857142857142855

자동 조사식 1 지역

C:₩Users₩user₩Desktop₩HaLim₩c₩ch07₩Debug₩ch07.exe

```
======== MENU =========
1. 프로그램 시작
0. 프로그램 종료
----------------------
메뉴번호를 입력하세요 : 1
식을 입력하세요 : 8 * 4
결과 : 32.0

======== MENU =========
1. 프로그램 시작
0. 프로그램 종료
----------------------
메뉴번호를 입력하세요 : 1
식을 입력하세요 : 3 / 7
결과 : 0.4

======== MENU =========
1. 프로그램 시작
0. 프로그램 종료
----------------------
메뉴번호를 입력하세요 : 0
프로그램을 종료합니다.
```

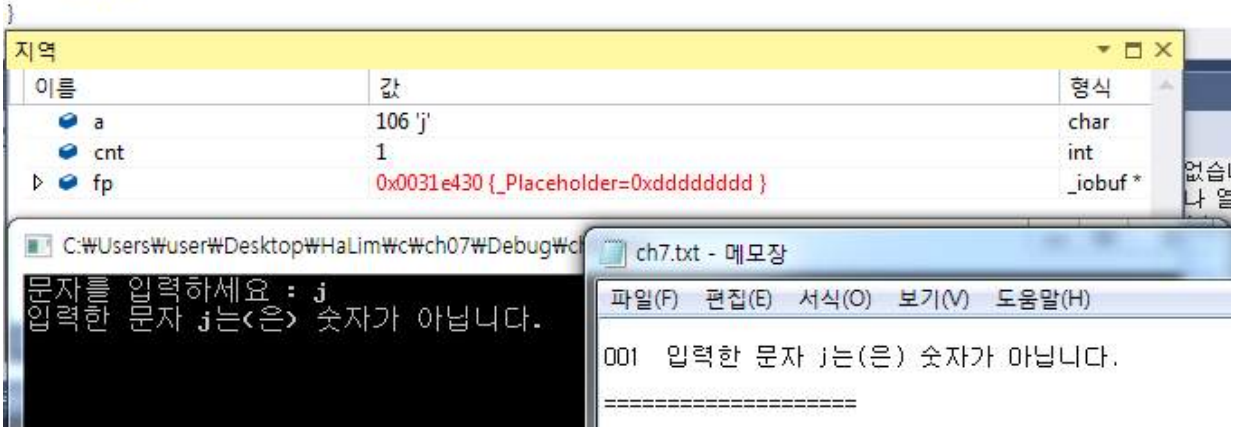
ch7.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
001  결과 : 32.0
002  결과 : 0.4
003  프로그램을 종료합니다.
====================
```

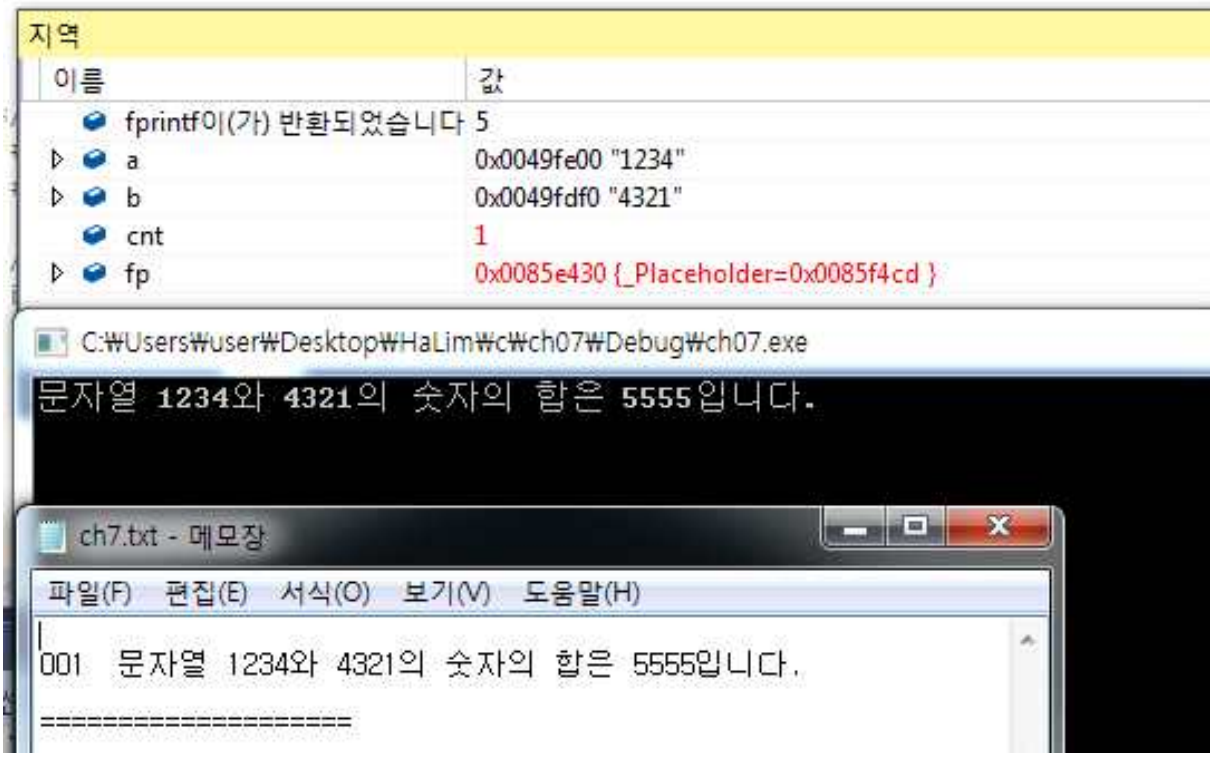
■ **문자 하나를 입력받아 숫자인지 아닌지 판별하는 함수 int checkInt(char)를 정의하고 입력한 문자가 숫자인지 아닌지 판별하는 프로그램을 작성하세요.**

```c
#include <stdio.h>
#pragma warning(disable:4996)
// 함수를 미리 선언해놓는 선언부
int checkInt(char a);

int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch7.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    char a = 0;                         // 입력받을 문자
    int cnt = 0;                        // 레코드 번호 cnt 선언
    fprintf(stdout, "문자를 입력하세요 : ");          // 문자 입력 안내메시지
    scanf("%c", &a);                                  // 문자 입력

    if (checkInt(a)) {                                // 문자를 매개변수로 chekInt값을 반환받아 조건으로 사용
        // 문자 a가 숫자일 때 1을 반환 받아 실행됨
        fprintf(stdout, "입력한 문자 %c는(은) 숫자입니다.₩n", a);
        fprintf(fp, "%03d  ", ++cnt);    // 파일에 레코드번호 출력
        fprintf(fp, "입력한 문자 %c는(은) 숫자입니다.₩n", a);
    }
    else {
        // 문자 a가 숫자가 아닐때 0을 반환 받아 실행됨
        fprintf(stdout, "입력한 문자 %c는(은) 숫자가 아닙니다.₩n", a);
        fprintf(fp, "%03d  ", ++cnt);    // 파일에 레코드번호 출력
        fprintf(fp, "입력한 문자 %c는(은) 숫자가 아닙니다.₩n", a);
    }
    fprintf(fp, "₩n====================₩n₩n");       // 파일에 구분선 출력
    fclose(fp);             // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch7.txt");     // 메모장으로 파일을 열어줌
    return 0;
}
int checkInt(char a) {
    if (a > 47 && a < 58)       // 48 ~ 57까지 아스키코드로 문자 0 에서 9
        return 1;
    else
        return 0;
}
```

- 

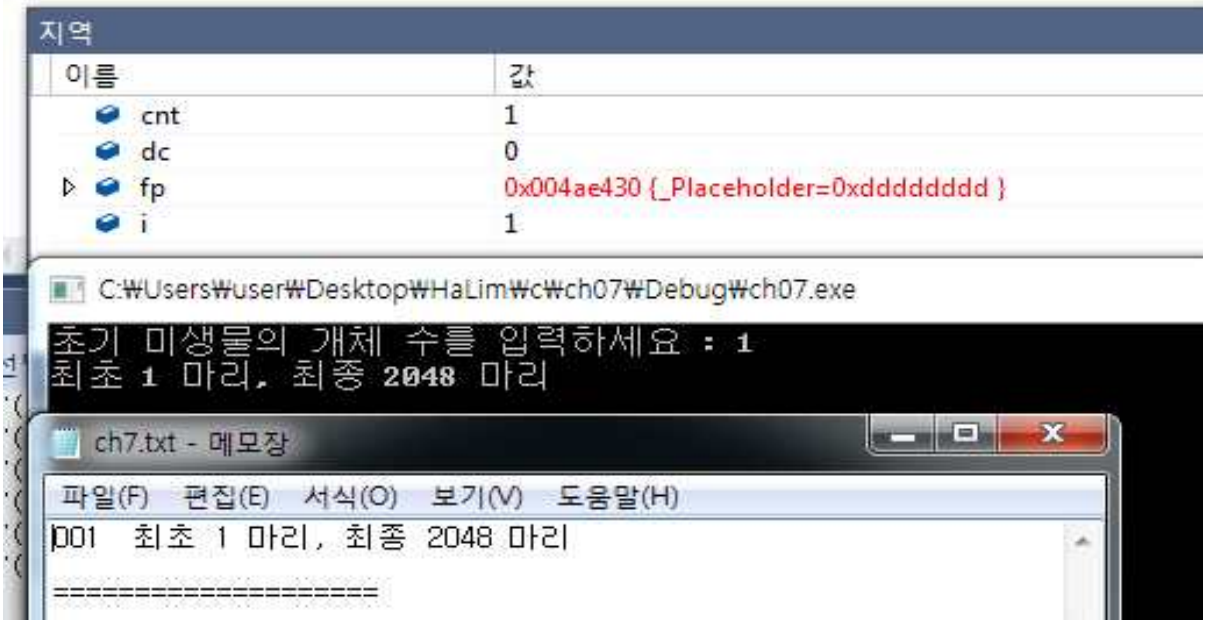
■ **stdlib.h 헤더에 정의되어 있는 atoi 함수는 사용자가 입력한 문자형 정수를 실제 정수형으로 변환해주는 함수입니다. atoi 함수를 사용해 문자열 상수 “1234” 와 “4321”을 정의하고 두 문자열을 숫자로 변환해 더한 수를 출력하는 프로그램을 작성하세요.**

```c
#include <stdio.h>
#include <stdlib.h>
#pragma warning(disable:4996)

int main() {
FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
fp = fopen("D:/ch7.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
const char a[5] = "1234", b[5] = "4321";                    // 문자열 상수 a와 b 정의
int cnt = 0;                        // 레코드 번호 cnt 선언

// atoi함수를 통해 두 문자열의 숫자 합을 출력
fprintf(stdout, "문자열 %s와 %s의 숫자의 합은 %d입니다.₩n", a, b, atoi(a)+ atoi(b));
fprintf(fp, "%03d  ", ++cnt);     // 파일에 레코드번호 출력
fprintf(fp, "문자열 %s와 %s의 숫자의 합은 %d입니다.₩n", a, b, atoi(a) + atoi(b));

fprintf(fp, "₩n====================₩n₩n");        // 파일에 구분선 출력
fclose(fp);             // fp에 있는 파일을 닫아줌
system("notepad.exe D:/ch7.txt");      // 메모장으로 파일을 열어줌
return 0;
}
```

- 

■ **A라는 미생물은 한 달에 한번 세포분열하여 개체 수가 2배로 증가합니다. 이러한 미생물A의 최초 개수를 입력받고, 미생물이 1월부터 12월까지 세포분열한다 했을 때 최종 개체수는 몇 개인지를 구하는 프로그램을 작성하세요. (단,재귀함수로 구현하며 전역변수 및 정적(static)변수 사용 금지)**

```c
#include <stdio.h>
#pragma warning(disable:4996)

// 함수를 미리 선언해놓는 선언부
int div(int i, int dc);

int main() {
    FILE * fp;                          // 파일 입출력을 위해 파일형 포인터 fp 선언
    fp = fopen("D:/ch7.txt", "at+");    // fp에 파일의 위치와 이어쓰기옵션을 넣어 파일을 열어줌
    int i, cnt = 0, dc = 0;                     // 레코드 번호 cnt 선언

    fprintf(stdout, "초기 미생물의 개체 수를 입력하세요 : ");       // 메뉴 입력 안내메시지
    scanf("%i", &i);                                    // 메뉴 번호 입력
    fprintf(stdout, "최초 %d 마리, 최종 %d 마리₩n", i, div(i, dc));          //
    fprintf(fp, "%03d  ", ++cnt);    // 파일에 레코드번호 출력
    fprintf(fp, "최초 %d 마리, 최종 %d 마리₩n", i, div(i, dc));       //

    fprintf(fp, "₩n======================₩n₩n");       // 파일에 구분선 출력
    fclose(fp);             // fp에 있는 파일을 닫아줌
    system("notepad.exe D:/ch7.txt");     // 메모장으로 파일을 열어줌
    return 0;
}

int div(int i, int dc)
{
    if (++dc == 12)
        return i;
    div(i * 2, dc);
}
```

- 

3. 자기성찰 및 평가

3-1 6주차-자기성찰 및 평가

■ 수업 및 실습을 통해서 배운 내용

- 함수가 무엇이고 어떤 형태로 사용하는 지
- 변수의 범위와 각 상황에 맞춰 사용할 수 있는 변수의 종류
- 함수의 작동원리
- 재귀 함수의 이해와 사용법

■ 느낀점 (자유롭게 기술)

- 자연스럽게 써왔던 main()문도 하나의 함수라는 것을 깨달았고 메인문에는 어떤 매게변수가 들어갈 수 있을지 궁금해졌다.

- 함수의 사용으로 메인함수에 모든 코드를 나열하는 것이 아니라 재사용 가능한 함수를통해 따로 빼서 사용하면서 앞으로의 코딩을 더 보기 좋게 할 수 있을 것 같다.

- 재귀함수는 반복문 보다 동작과정이 너무 복잡해서 많이 연습하여 익숙해 져야 할 것 같다.

- 시험기간이라 과제작성을 할 시간이 빠듯해서 심화문제를 여러 가지 하지 못했습니다.
  다음 과제부터는 책에 있는 심화문제나 인터넷 심화문제, 그리고 이해가 어려운 코드는 메모리구조까지 꼼꼼히 그려서 제출하겠습니다.