

기초프로그래밍 강의 (C언어)

Workbook

김 광

(kkim0691@gmail.com)

서 문

이 워크북은 컴퓨터 프로그래밍을 배우는 데에 있어서 기초가 되는 논리들을 이해하고 이를 활용한 프로그램을 제작할 수 있도록 훈련하기 위해 제작된 교재이다. 이를 위해 기초부터 응용까지 단계별로 영역을 나누어 각 단계마다 필요한 논리를 간단한 예제를 통해 설명한다. 그리고 각 단계마다 반복적인 실습문제들을 제시하여 풀게 함으로써 프로그래밍 논리를 차근차근 습득해 나갈 수 있도록 돋고 있다.

본 교재에서는 C 언어를 기준으로 설명하고 있지만, 여기에서 다루는 프로그래밍 논리 연습은 컴퓨터 프로그래밍을 공부하는 학생들에게 특정 프로그래밍 언어에 국한되지 않는 공통적인 프로그래밍 논리를 스스로 제작할 수 있도록 하는데 도움이 될 것이다. 따라서 앞으로 어떤 언어를 사용 하더라도 본 교재에서 습득한 프로그램 해결 능력을 활용하면 빠르게 개발 능력을 키워 갈 수 있을 것이다.

프로그래밍 논리 제작 수준에 따라 영역 1부터 영역 4까지 나누었으며, 각 영역에는 4개의 단계로 나누어 각 단계마다 필요한 프로그래밍 논리 문제들을 예제와 함께 간단히 설명한다. 그리고 각 단계마다 제시되는 실습 문제들은 주어진 문제와 조건, 사용해야 하는 변수나 함수의 이름과 형식을 준수하고, 제시된 실행 예시 화면을 참고하여 문제를 해결하기 바란다. 본 교재는 일반적인 C 언어 교재에서 다루는 구체적인 문법들을 설명하고 있지는 않다. 그렇기 때문에 본 교재의 실습 문제를 풀 때에는 다른 C 언어 교재나 인터넷의 관련 사이트 정보를 충분히 활용하기 바란다.

그리고 실습문제를 풀 때에는 제시된 변수만을 사용해서 풀어보기를 권한다. 임의로 변수를 추가 해서 만들기보다는 주어진 조건을 최대한 지키면서 문제를 해결할 수 있다면 훨씬 유익한 공부가 될 것으로 기대한다.

2016년 2월

저자.

목 차

영역 1. 프로그래밍 제어구조 기초

[Step A] 변수의 사용과 데이터 입출력	6
[Step B] 단순 조건문 사용하기	13
[Step C] 복합 조건문 사용하기	25
[Step D] 반복문 사용하기	39

영역 2 : 프로그래밍 제어구조 응용

[Step E] 복합 반복문 사용하기	54
[Step F] 배열 사용하기	63
[Step G] 조건과 반복을 활용하는 응용 예제 해결하기	75
[Step H] C언어의 주요 함수 사용하기	85

영역 3 : 함수 만들어 사용하기

[Step I] 리턴 값이 없는 함수 만들기	97
[Step J] 파라미터는 없고 리턴 값만 있는 함수 만들기	105
[Step K] 파라미터와 리턴 값이 모두 있는 함수 만들기	111
[Step L] 재귀 호출 함수 만들기	119

영역 4. 구조적 데이터 다루기

[Step M] 구조체 사용하기	128
[Step N] 포인터 사용하기	136
[Step O] 구조체와 배열 함께 사용하기	146
[Step P] 파일 사용하기	158

실습문제 소스 및 목록

영역 1. 프로그래밍 제어구조 기초

영역 1에서는 프로그래밍의 가장 기본이 되는 변수의 사용과 입출력 처리 방법, 조건문과 반복문을 통한 제어구조의 기초를 연습한다. 영역 1은 다음과 같이 4개의 단계로 구성된다.

Step A : 변수의 사용과 데이터 입출력

Step B : 단순 조건문 사용하기

Step C : 복합 조건문 사용하기

Step D : 반복문 사용하기

[Step A] 변수의 사용과 데이터 입출력

○ 변수의 사용

프로그램을 제작할 때 어떤 종류이건 데이터나 값을 다루기 위해서는 변수를 사용해야 한다. 변수를 사용할 때에는 다루어야 하는 값이 숫자(정수와 실수)인지 문자인지에 따라 변수의 형식(type)을 구별해서 결정해야 하므로 프로그래밍 언어에서 사용가능한 변수 형식을 미리 숙지하고 있어야 한다. C언어의 경우 주로 사용하는 변수 형식은 다음과 같다.

형식	데이터 종류	설명
int	정수 숫자	정수로 표현하는 숫자를 다룬다.
long	정수 숫자	int보다 자리수가 큰 정수로 표현하는 숫자를 다룬다.
float	실수 숫자	실수로 표현하는 숫자를 다룬다.
double	실수 숫자	float보다 자리수가 큰 실수로 표현하는 숫자를 다룬다.
char	문자	1글자(영문, 기호 등)의 문자를 다룬다.

그리고 변수의 이름을 정해주어야 하는데 영문과 숫자를 적당히 사용하되 변수의 용도를 쉽게 파악할 수 있도록 의미 있는 이름으로 정하는 것이 좋다. 다음과 같은 이름은 좋은 변수 이름이다.

```
int apple_count; // 사과 개수
char keyChar; // 입력받은 문자
```

프로그램에서 변수를 다루는 가장 기본적인 방법은 변수 선언과 값 할당이다. 예를 들어 1개에 1,000원씩 판매하는 사과 15개의 가격을 계산하라고 한다면 누구나 1,000원 곱하기 15개인 15,000 원을 계산할 수 있을 것이다. 그렇다면 이 계산을 컴퓨터에게 시켜본다면 어떻게 해야 할까? 먼저 컴퓨터에게 사과의 개수를 다루기 위한 변수와 전체 가격을 다루기 위한 변수를 사용한다고 선언해야 한다.

```
int apple_count; // 사과 개수
int price; // 전체 사과의 가격
```

그런 다음에 각각의 변수에게 값을 할당해준다. 즉 사과 개수에는 15를 할당하고, 전체 사과의 가격은 사과 개수에 1,000원을 곱한 값을 할당하는 것이다.

```
apple_count = 15; // 사과 개수에 15를 할당한다.
price = apple_count * 1000; // 전체 가격에는 사과 개수와 1000을 곱한 값을 할당한다.
```

하나만 더 생각해보자. 만일 사과 1개의 가격이 1,200원으로 변경된다면 어떻게 해야 할까? 다음과 같이 프로그램을 변경하면 쉽게 해결할 수 있어 보인다.

```
price = apple_count * 1200; // 전체 가격에는 사과 개수와 1200을 곱한 값을 할당한다.
```

하지만 만약 사과 1개의 가격이 앞으로도 변동될 가능성이 있을 때에는 다음과 같이 사과 1개당 단가를 별도의 변수로 선언하여 사용하는 것이 더 좋은 방법이다.

```
int unit_price; // 단가
unit_price = 1200; // 단가에 1200을 할당한다.
price = apple_count * unit_price; // 전체 가격에는 사과 개수와 단가를 곱한 값을 할당한다.
```

○ 데이터 입출력

프로그램에서 변수에 값을 할당하기 위해서 사용자로부터 입력을 받아야 하는 경우가 있다. 그리고 사용자에게 프로그램의 진행 과정이나 결과를 모니터 화면을 통해 출력해주어야 할 필요가 있다. 이런 경우에 데이터 입력, 출력 기능을 사용하게 된다.

먼저 화면에 원하는 문장을 출력하는 구문은 다음과 같다.

```
printf("안녕하세요?"); // 화면에 "안녕하세요?"라는 문장을 출력한다.
printf("\n사과의 개수는 몇 개입니까?"); // \n은 화면에서 다음 라인으로 이동하라는 표시이다.
```

특정 변수의 값을 화면에 출력하는 방법은 다음과 같다. 이 때 주의해야 할 것은 출력하려는 변수의 형식에 맞추어 주어야 한다.

```
printf("사과의 개수는 모두 %d개 입니다\n", apple_count); // %d에는 정수형 변수의 값이 들어간다.
```

사용자에게 특정 변수에 할당할 값을 입력받는 방법은 다음과 같다.

```
printf("사과의 개수는 모두 몇 개 입니까? ");
scanf("%d", &apple_count); // 사용자가 입력한 정수 값을 apple_count에 할당한다.
```

다음과 같이 한 번에 2개의 변수에 사용자로부터 입력된 값을 할당할 수도 있다.

```
printf("사과 1개의 가격과 사과의 갯수를 입력하세요. ");
scanf("%d %d", &unit_price, &apple_count);
```

이 때 입력 또는 출력하는 값의 형식을 결정하는 주요 표시자의 종류는 다음과 같다.

표시자	형식	설명
%d	정수형	정수형 변수에 대해 값을 출력하거나, 입력받는다.
%f	실수형	실수형 변수에 대해 값을 출력하거나, 입력받는다.
%c	문자형	문자형 변수 (영문, 기호 등)의 값을 출력하고나 입력받는다.
%s	문자배열	문자열(배열) 변수의 값을 출력하거나 입력받는다.

위 표시자를 `printf()`에서 사용할 때에는 출력하려는 데이터의 자릿수를 지정할 수 있다. 예를 들어 정수형 변수의 값을 5자리에 맞추어 출력하려면 `%5d`, 실수형 변수의 값을 소수점 2자리까지 맞추어 출력하려면 `%.1f`로 넣어주면 된다. 그리고 음수가 아닌 정수형 변수의 값을 출력할 경우 `%u`를 사용할 수도 있다.

여기서 한 가지 주의해야 할 것은 문자나 숫자(정수, 실수) 데이터를 입력받아 변수에 할당해야 하는 경우에 사용하는 `scanf()`에서는 반드시 변수이름 앞에 & 기호를 붙여주어야 한다. 이렇게 변수 앞에 & 기호가 붙어 있어야 C 프로그램에서는 사용자가 입력한 값을 제대로 할당시킬 수 있다.

그러면 이제 몇 개의 변수를 사용하여 간단한 입출력을 수행하는 프로그램을 만들어보자. 문제는 사과의 단가와 개수를 입력받아 전체 사과의 가격을 계산하는 것이다. 이를 해결하기 위해서 다음과 같은 순서대로 프로그램을 제작하여 보자.

1. 사과 1개의 가격을 입력받는다.
2. 사과 개수를 입력받는다.
3. 전체사과의 가격을 계산한다.
4. 계산한 가격을 출력한다.

위의 순서에 따라 프로그램 코드를 만들면 다음과 같다.

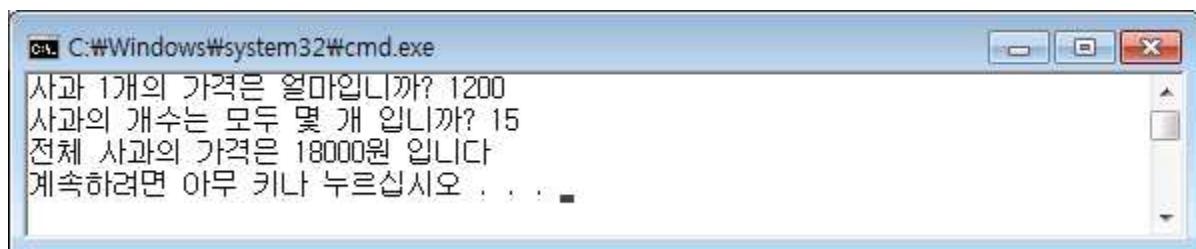
```
printf("사과 1개의 가격은 얼마입니까? ");
scanf("%d", &unit_price);           // 사용자가 입력한 정수 값을 unit_price에 할당한다.
printf("사과의 개수는 모두 몇 개 입니까? ");
scanf("%d", &apple_count);          // 사용자가 입력한 정수 값을 apple_count에 할당한다.
price = apple_count * unit_price;   // 전체 사과의 가격을 계산한다.
printf("전체 사과의 가격은 %d원 입니다\n", price); // %d에는 정수형 변수 price의 값이 들어간다.
```

이를 완전한 프로그램을 제작하기 위해서는 필요한 헤더파일을 지정하고, 메인 함수 내에 사용하여야 할 변수를 미리 선언해야 한다. 이렇게 만들어진 C 프로그램은 다음과 같다.

C Workbook

```
// 예제 ex_A.c
#include <stdio.h>
void main()
{
    int unit_price, apple_count, price;
    printf("사과 1개의 가격은 얼마입니까? ");
    scanf("%d", &unit_price);
    printf("사과의 개수는 모두 몇 개 입니까? ");
    scanf("%d", &apple_count);
    price = apple_count * unit_price;
    printf("전체 사과의 가격은 %d원 입니다\n", price);
}
```

다음 화면은 이 프로그램의 실행 결과 예시를 보인 것이다.



○ 실습 문제

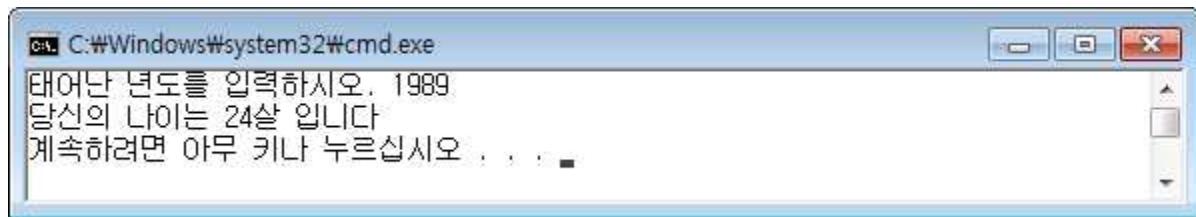
[A01] 나이 계산

태어난 년도를 입력받은 후, 이 값을 이용하여 나이를 계산하고 출력하라.

단, 나이 = $2012 - \text{태어난 년도} + 1$ 로 계산한다.

변수는 다음과 같이 사용하라.

```
int birth_year;      // 태어난 년도  
int age;            // 나이
```



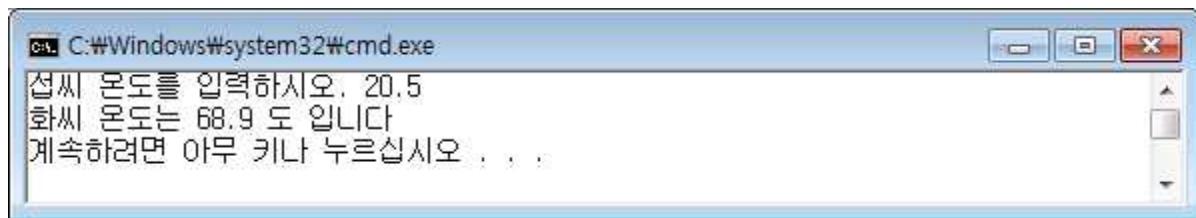
[A02] 온도 변환

섭씨 온도를 입력받아 이 값을 화씨온도로 변환하여 출력하라.

단, 화씨 온도 = 섭씨 온도 * 1.8 + 32 로 계산한다.

변수는 다음과 같이 사용하라.

```
float c_degree;    // 섭씨 온도  
float f_degree;    // 화씨 온도
```



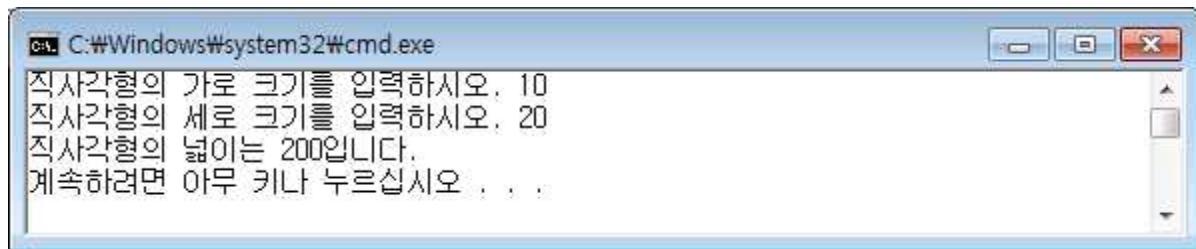
[A03] 직사각형 넓이 계산

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 넓이를 계산하고 출력하라.

단, 직사각형의 넓이 = 가로크기 * 세로크기 로 계산한다.

변수는 다음과 같이 사용하라.

```
int width, height; // 가로크기, 세로크기  
int area; // 직사각형의 넓이
```



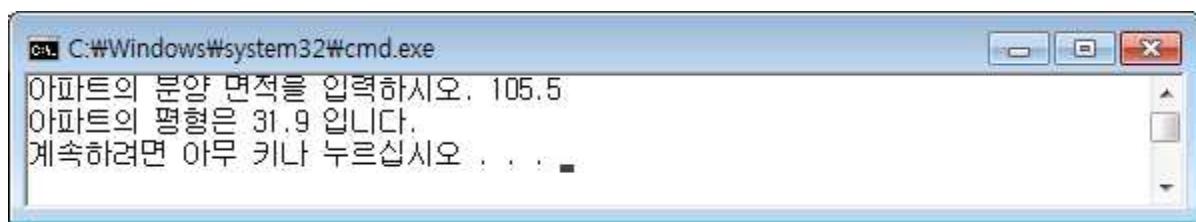
[A04] 아파트 평형 계산

아파트의 분양 면적을 제곱미터(m^2) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하여 출력하라.

단, 평형 수 = 제곱미터 / 3.305 로 계산한다.

변수는 다음과 같이 사용하라.

```
float m2_area; // 면적 (제곱미터)  
float pyung_area; // 면적 (평수)
```



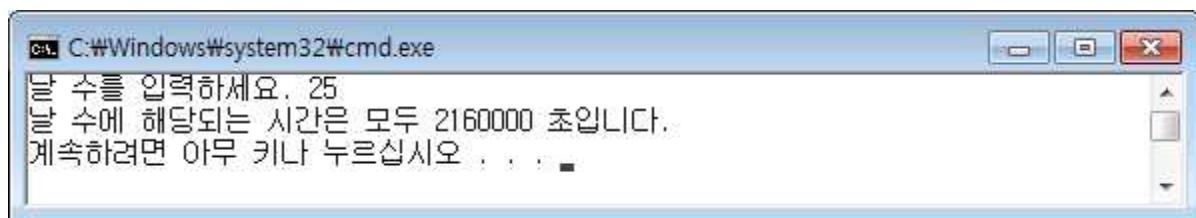
[A05] 날짜 계산

날 수를 입력받아 이 날 수에 해당되는 기간은 모두 몇 초인지 계산하여 출력하라.

단, 초 = 날 수 * 24 * 60 * 60 으로 계산한다.

변수는 다음과 같이 사용하라.

```
int days;           // 날 수
int seconds;        // 초 단위 시간
```



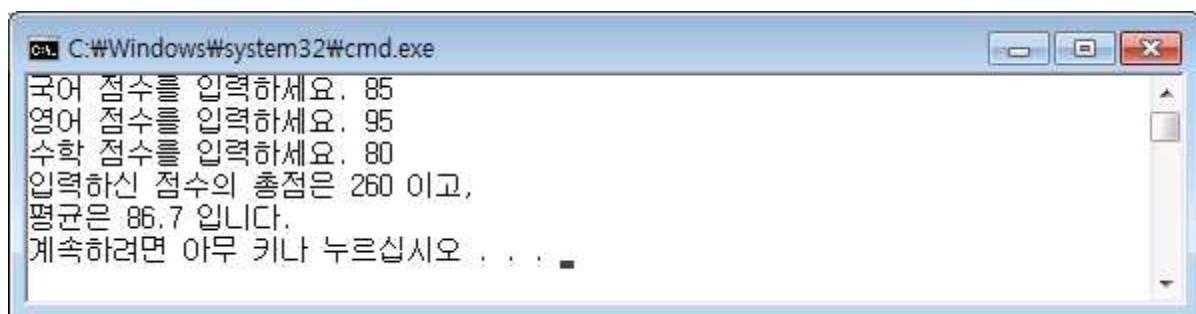
[A06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하여 출력하라.

단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하라.

변수는 다음과 같이 사용하라.

```
int kor, eng, math;      // 국어점수, 영어점수, 수학점수
int total;                // 총점
float average;            // 평균점수
```



[A07] 파일 용량 계산

파일 용량을 기가바이트 단위로 입력받아 이 값을 메가바이트, 킬로바이트, 바이트 단위로 계산하여 각각 출력하라.

단, 계산방법은 다음과 같다.

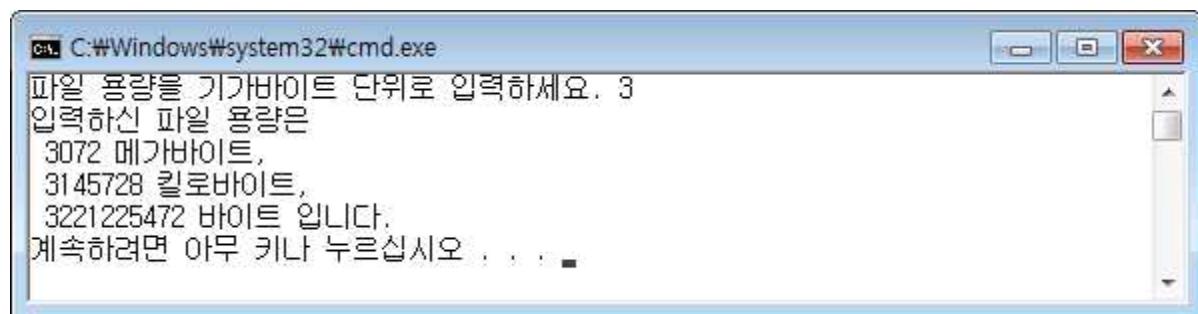
메가바이트 수 = 기가바이트 수 * 1024

킬로바이트 수 = 메가바이트 수 * 1024

바이트 수 = 킬로바이트 수 * 1024

변수는 다음과 같이 사용하라.

```
int gigabytes;           // 용량(기가바이트 단위)  
int megabytes;          // 용량(메가바이트 단위)  
int kilobytes;          // 용량(킬로바이트 단위)  
unsigned int bytes;      // 용량(바이트 단위)
```



[Step B] 단순 조건문 사용하기

프로그램에서는 입력된 값이나 계산한 값에 대해서 특정 조건에 부합하는지의 여부를 판정하여 그 결과에 따라 다른 작업을 수행하도록 해야 할 필요가 있다.这时候에 사용하는 구문이 조건문인데, 조건문의 가장 기본적인 모습은 다음과 같다.

```
if (조건문) {
    조건문이 참인 경우에 수행해야 하는 구문들
}
```

예를 들어 입력받은 사과의 개수가 30개가 넘는 경우에 “한 박스에 담을 수 없습니다.”라고 출력하려고 하면 다음과 같이 하면 된다.

```
int apple_count;
printf("사과의 개수는? ");
scanf("%d", &apple_count);
if (apple_count > 30) {
    printf("한 박스에 담을 수 없습니다.\n");
}
```

조건문이 참인 경우에 수행해야 하는 내용과 그 외의 경우 즉, 조건문이 거짓인 경우에 수행해야 하는 내용이 따로 있는 경우에는 다음과 같은 형식의 구문을 사용한다.

```
if (조건문) {
    조건문이 참인 경우에 수행해야 하는 구문들
}
else {
    조건문이 거짓인 경우에 수행해야 하는 구문들
}
```

위의 예제에서 사과의 개수가 30개가 넘지 않는 경우에는 “한 박스에 담을 수 있습니다.”라고 출력하도록 변경하면 다음과 같이 하면 된다.

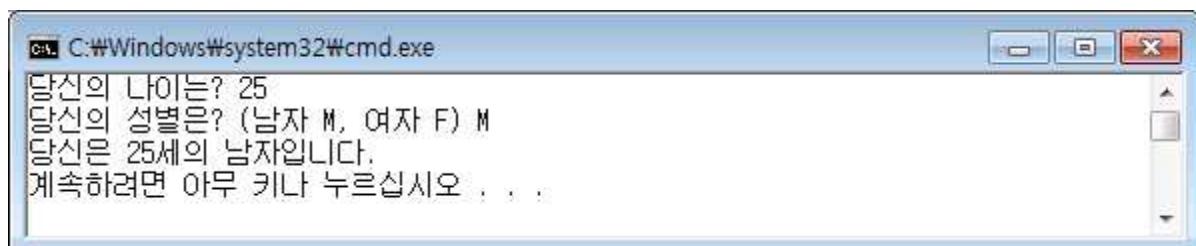
```
int apple_count;
printf("사과의 개수는? ");
scanf("%d", &apple_count);
if (apple_count > 30) {
    printf("한 박스에 담을 수 없습니다.\n");
}
```

```
else {
    printf("한 박스에 담을 수 있습니다.\n");
}
```

이번에는 나이와 성별을 입력받아 이를 출력하는 구문을 만들어 보자. 나이는 정수형 변수로 처리하고, 성별은 'M' 또는 'F'의 값을 갖는 문자형 변수로 처리하도록 한다.

```
int age;                                // 나이
char gender;                             // 성별(남자 'M', 여자 'F')
printf("당신의 나이는? ");
scanf("%d", &age);
printf("당신의 성별은? (남자 M, 여자 F) ");
fflush(stdin);                           // 문자입력 전에 입력버퍼를 비운다.
scanf("%c", &gender);
if (gender == 'M')                      // 성별이 남자인 경우
{
    printf("당신은 %d세의 남자입니다.\n", age);
}
else if (gender == 'F')                 // 성별이 여자인 경우
{
    printf("당신은 %d세의 여자입니다.\n", age);
}
else                                    // 성별이 남자도 여자도 아닌 경우
{
    printf("성별을 잘못 입력하였습니다.\n");
}
```

위 구문에서 `fflush(stdin);` 구문의 역할을 이해해야 할 필요가 있다. 보통 `scanf()` 를 사용해서 문자형 변수의 값을 입력받을 경우에 이보다 먼저 수행된 `scanf()` 구문이 있는 경우에는 문자형 변수 입력이 제대로 처리되지 않는다. 왜냐면 먼저 수행된 `scanf()` 구문에서 엔터키가 입력버터에 남아 있기 때문에 이를 문자형 변수의 값으로 인식해버리는 것이다. 따라서 이미 수행된 `scanf()` 구문 다음에 문자형 변수 입력을 위한 `scanf()` 구문이 있는 경우에는 반드시 `fflush(stdin);` 구문을 사용해서 입력버퍼를 비워주어야 한다.



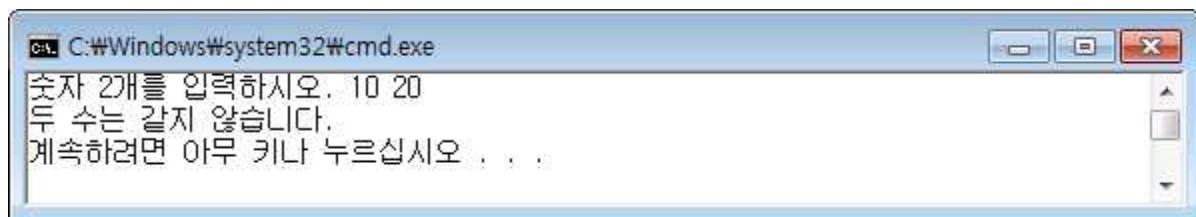
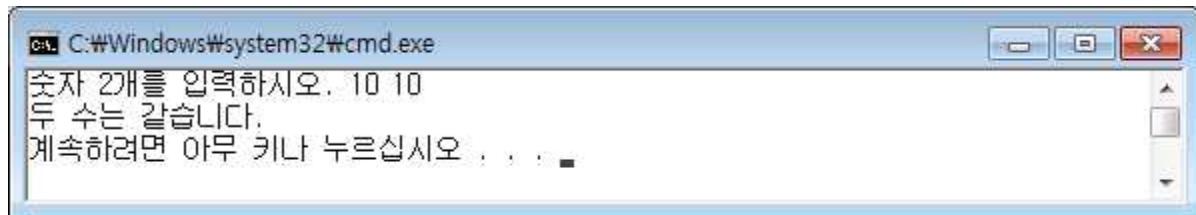
그리면 조건문을 이용하는 간단한 프로그램을 만들어보자. 문제는 2개의 숫자를 입력받아 이 숫자들이 같은 수인지를 판정하는 것이다. 이 문제를 해결하기 위해 다음과 같은 순서를 생각하자.

1. 숫자 2개의 값을 입력받는다.
2. 만일 첫 번째 숫자와 두 번째 숫자가 같으면 3번을, 그렇지 않으면 4번을 수행한다.
3. “두 수는 같습니다.”라고 출력한다.
4. “두 수는 같지 않습니다.”라고 출력한다.

이 문제를 해결하는 C 프로그램은 다음과 같다.

```
// 예제 ex_B.c
#include <stdio.h>
void main()
{
    int num1, num2;
    printf("숫자 2개를 입력하시오. ");
    scanf("%d %d", &num1, &num2);
    if (num1 == num2) {
        printf("두 수는 같습니다.\n");
    }
    else {
        printf("두 수는 같지 않습니다.\n");
    }
}
```

이 프로그램을 수행한 결과는 다음과 같다. 같은 수를 넣은 경우와 다른 수를 넣은 경우에 따라 출력 내용이 달라지는 것을 확인할 수 있다.

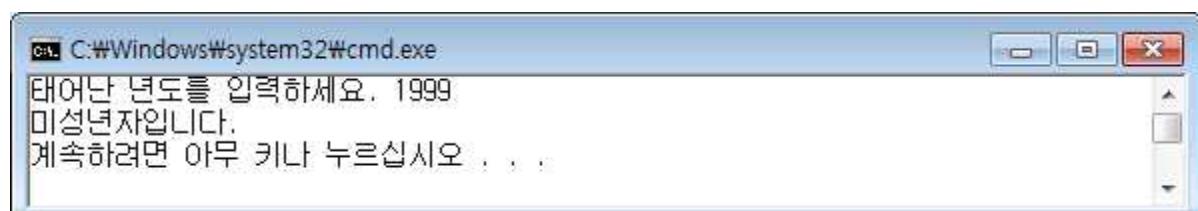
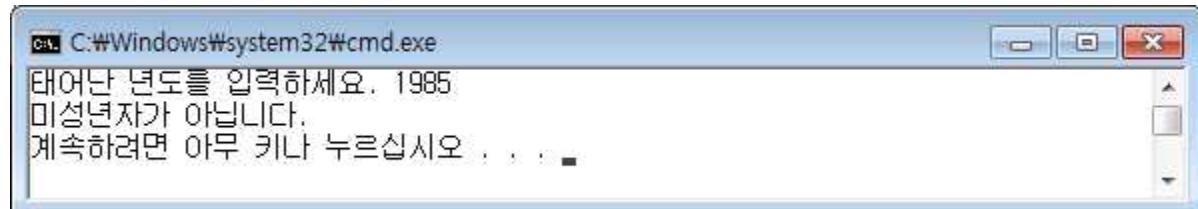


○ 실습 문제

[B01] 나이 계산 및 미성년자 판정

태어난 년도를 입력받아 나이를 계산한 후, 미성년자인지 여부를 판정하여 그 결과를 출력하라.
단, 나이 = $2012 - \text{태어난 년도} + 1$ 로 계산하고 나이가 20세 미만인 경우, 미성년자로 판정한다.
변수는 다음과 같이 사용하라.

```
int birth_year;      // 태어난 년도
int age;            // 나이
```



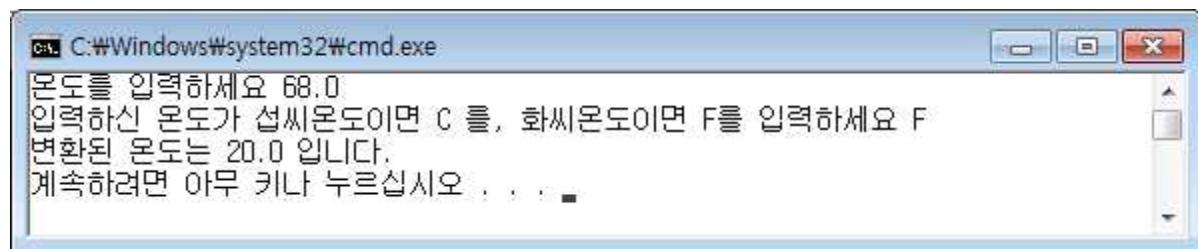
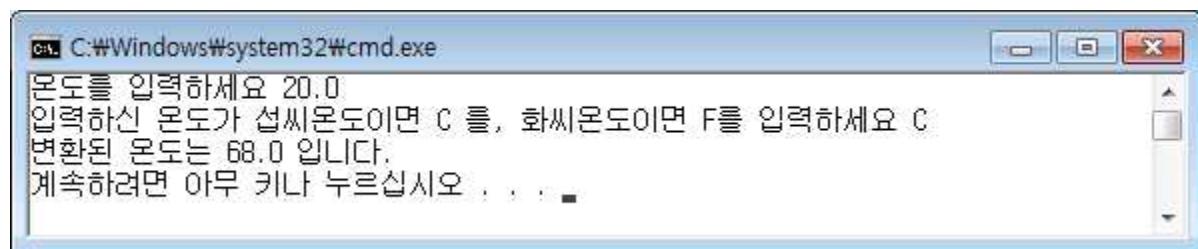
[B02] 온도 상호 변환

온도를 입력받은 후, 이 값이 섭씨온도인지 화씨온도인지 종류를 C또는 F로 입력받아 섭씨온도이면 화씨온도로 변환하고, 화씨온도이면 섭씨온도로 변환하여 그 값을 출력하라.

단, 화씨온도 = 섭씨온도 * 1.8 + 32, 섭씨온도 = (화씨온도 - 32)/1.8 로 계산한다.

변수는 다음과 같이 사용하라.

```
float input_degree;           // 입력받은 온도  
char kind;                   // 온도의 종류, 섭씨온도이면 'C', 화씨온도이면 'F'  
float output_degree;         // 변환한 온도
```



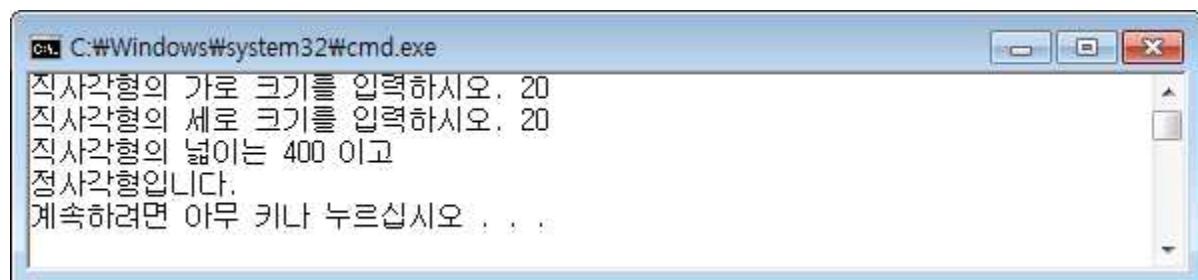
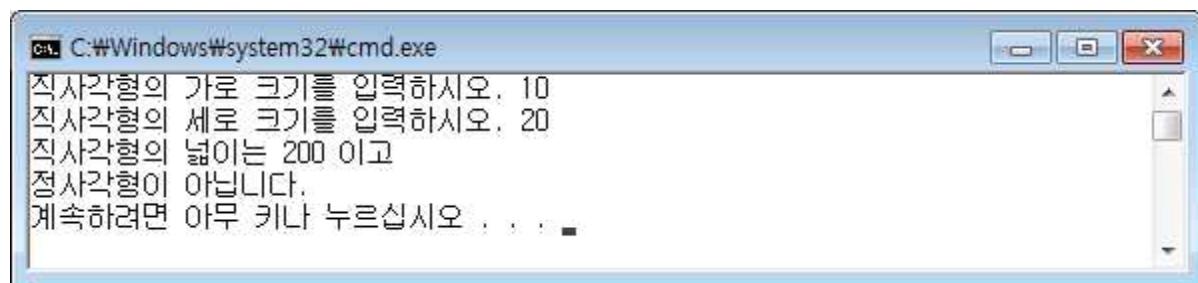
[B03] 직사각형 넓이 계산 및 정사각형 판정

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 넓이를 계산하고 정사각형인지의 여부를 판정하여 함께 출력하라.

단, 직사각형의 넓이 = 가로크기 * 세로크기 로 계산한다.

변수는 다음과 같이 사용하라.

```
int width, height; // 가로크기, 세로크기  
int area;           // 사각형의 넓이
```

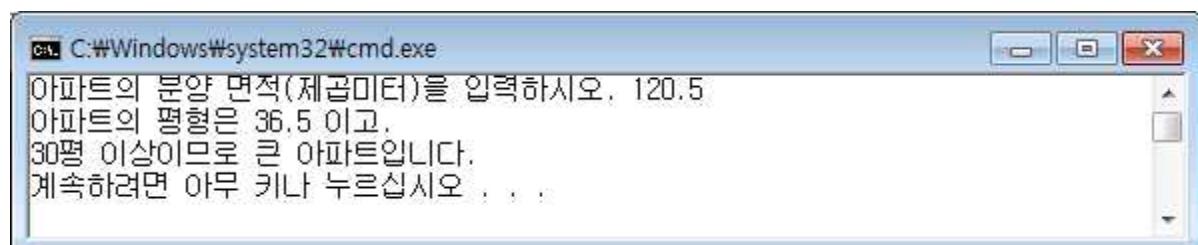
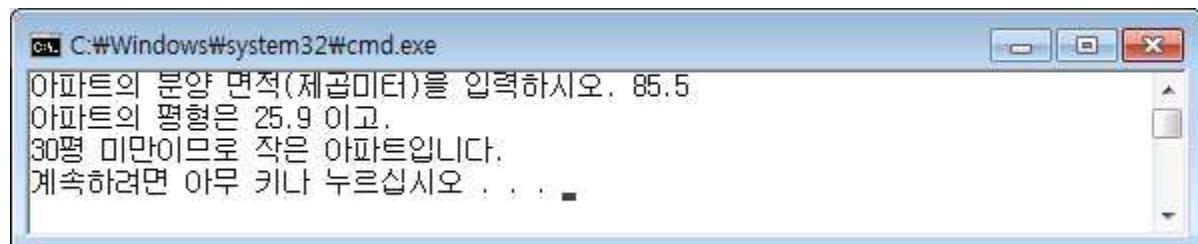


[B04] 아파트 평형 계산 및 종류 판정

아파트의 분양 면적을 제곱미터(m^2) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하라. 그리고 평형 수에 따라 아파트의 종류가 작은 아파트인지 큰 아파트인지 판정하여 판정 결과를 출력하라.
단, 평형 수 = 제곱미터 / 3.305로 계산하고, 30평 미만이면 작은 아파트 30평 이상이면 큰 아파트로 판정한다.

변수는 다음과 같이 사용하라.

```
float m2_area;      // 면적 (제곱미터)  
float pyung_area;  // 면적 (평수)
```



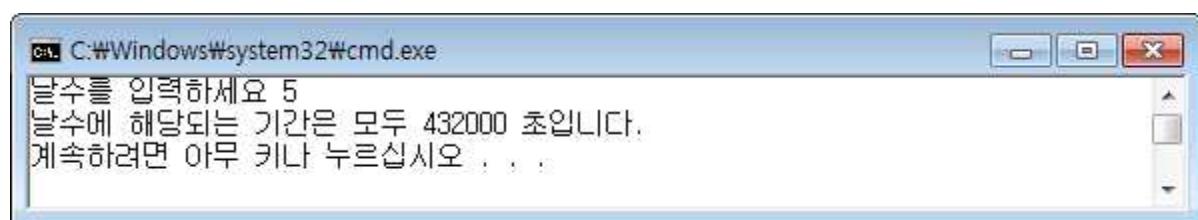
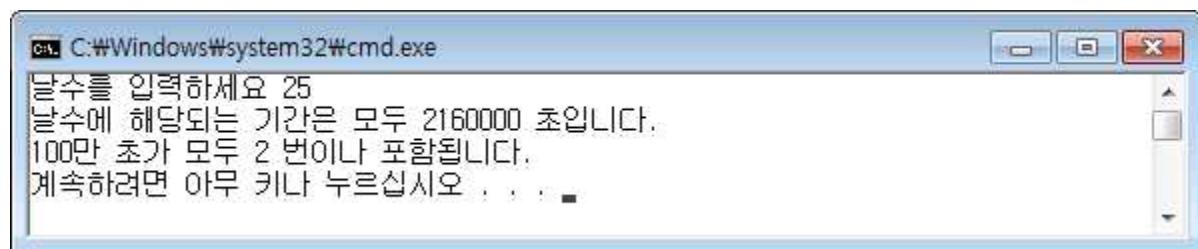
[B05] 날짜 계산

날 수를 입력받아 이 날 수에 해당되는 기간은 모두 몇 초인지 계산하고, 100만 초가 넘는 경우에는 100만 초가 모두 몇 번이나 포함되는 지 계산하여 출력하라.

단, 초 = 날 수 * 24 * 60 * 60 으로 계산한다.

변수는 다음과 같이 사용하라.

```
int days;           // 날 수  
int seconds;        // 초 단위 시간  
int m_count;        // 100만 초 포함 횟수
```



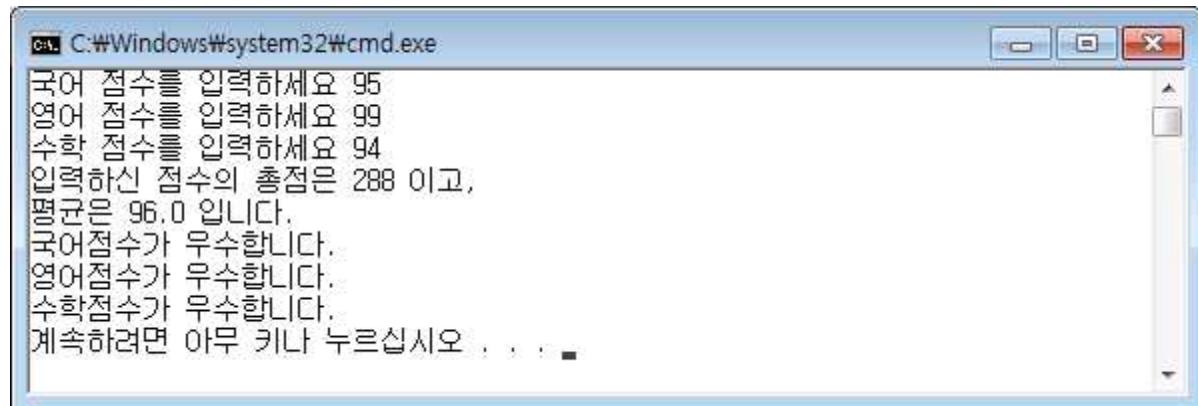
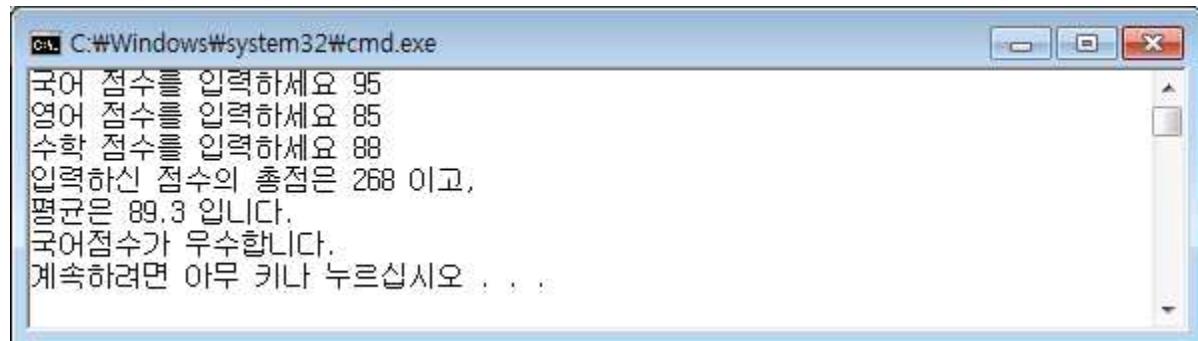
[B06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하고, 각 과목별로 90점 이상이면 성적우수로 표시하여 출력하라.

단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하라.

변수는 다음과 같이 사용하라.

```
int kor, eng, math;           // 국어점수, 영어점수, 수학점수
int total;                   // 총점
float average;               // 평균점수
```



[B07] 파일 전송 시간 계산

파일 용량을 메가바이트 단위로 입력받고, USB 포트가 2.0인지 아닌지를 'Y' 또는 'N'으로 입력받아 이에 따라 파일 전송 시간을 초 단위로 계산하여 출력하라.

단, 계산방법은 다음과 같다.

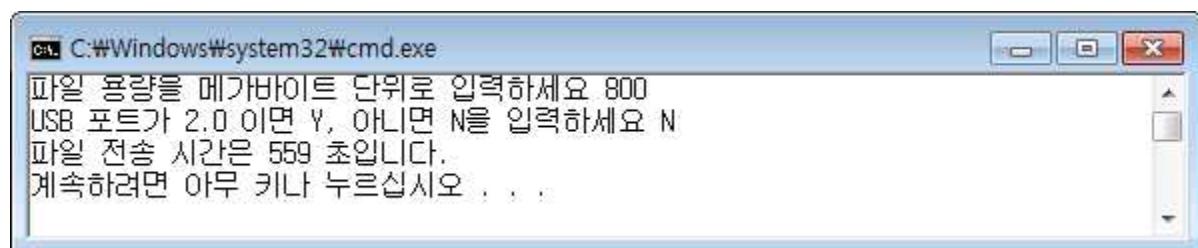
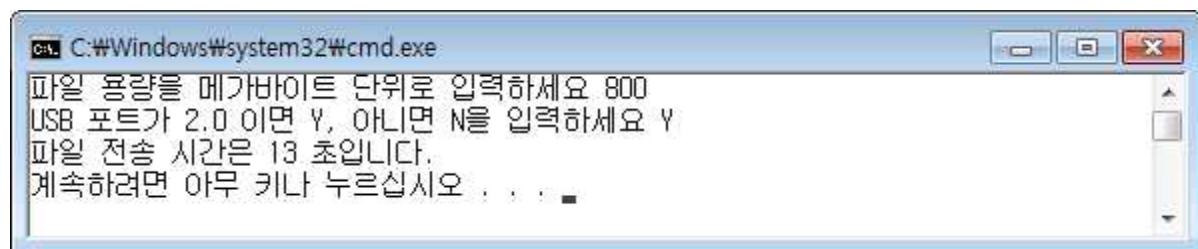
바이트 수 = 메가바이트 수 * 1024 * 1024

USB 1.1 전송 속도 = 1,500,000바이트 / 초

USB 2.0 전송 속도 = 60,000,000바이트 / 초

변수는 다음과 같이 사용하라.

```
int megabytes;           // 용량(메가바이트 단위)  
long bytes;             // 용량(바이트 단위)  
char usb2;               // USB 2.0 사용여부 (Y: 예, N: 아니요)  
int time;                // 전송시간(초 단위)
```



[B08] 다양한 조건 판정

3개의 정수를 입력받아 이 숫자들에 대해서 다음 조건 중에 만족시키는 번호들을 모두 출력하라.

- 1번. 3개의 숫자 중 적어도 두 수의 값이 같다.
- 2번. 3개의 숫자 중 적어도 두 수의 크기가 모두 50 보다 크다.
- 3번. 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.
- 4번. 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.

변수는 다음과 같이 사용하라.

```
int num1, num2, num3; // 첫 번째 숫자, 두 번째 숫자, 세 번째 숫자
```

C:\Windows\system32\cmd.exe

첫번째 숫자를 입력하세요 10
 두번째 숫자를 입력하세요 20
 세번째 숫자를 입력하세요 30
 3번 조건 만족 : 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.
 4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
 계속하려면 아무 키나 누르십시오 . . .

C:\Windows\system32\cmd.exe

첫번째 숫자를 입력하세요 10
 두번째 숫자를 입력하세요 10
 세번째 숫자를 입력하세요 20
 1번 조건 만족 : 3개의 숫자 중 적어도 두 수의 값이 같다.
 3번 조건 만족 : 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.
 4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
 계속하려면 아무 키나 누르십시오 . . .

C:\Windows\system32\cmd.exe

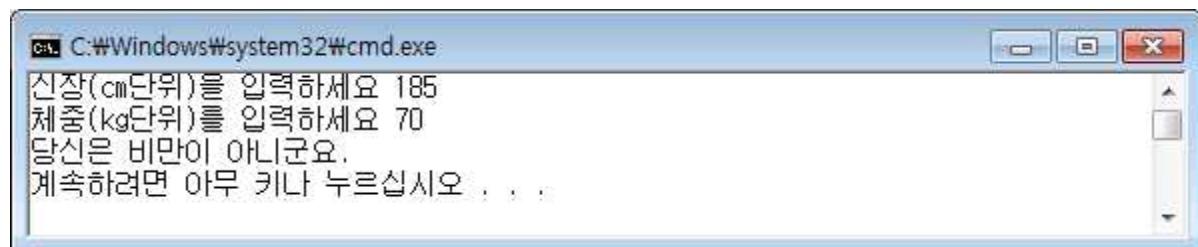
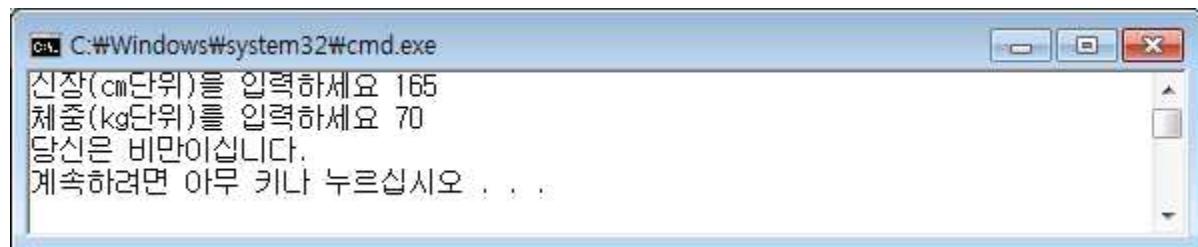
첫번째 숫자를 입력하세요 50
 두번째 숫자를 입력하세요 100
 세번째 숫자를 입력하세요 200
 2번 조건 만족 : 3개의 숫자 중 적어도 두 수의 크기가 모두 50 보다 크다.
 4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
 계속하려면 아무 키나 누르십시오 . . .

[B09] 비만 판정

신장(cm단위)과 체중(kg단위)를 입력받은 후, 비만 여부를 판정하여 출력하라.
단, 비만여부는 다음 비만도 수치가 25이상인 경우에 "비만"으로 판단한다.
비만도 수치 = 체중(kg) / (신장(m)의 제곱) 으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.

변수는 다음과 같이 사용하라.

```
int height, weight;           // 신장(cm), 체중(kg)  
float bmi;                  // 비만도 수치
```



[Step C] 복합 조건문 사용하기

조건문을 사용하여 프로그램을 만드는 경우에 단순하게 조건문이 참이냐 거짓이냐만으로 구분할 수 없는 복잡한 경우가 있다. 즉 조건문이 연속으로 필요한 경우에는 다음과 같은 형식의 조건문을 사용해야 한다.

```
if (첫 번째 조건문) {
    첫 번째 조건문이 참인 경우에 수행해야 하는 구문들
}
else if (두 번째 조건문) {
    첫 번째 조건문이 거짓이고 두 번째 조건문이 참인 경우에 수행해야 하는 구문들
}
else if (세 번째 조건문) {
    첫 번째 조건문과 두 번째 조건문 모두 거짓이고 세 번째 조건문이 참인 경우에 수행해야 하는 구문들
}
...
else{
    위에서의 모든 조건문이 전부 거짓인 경우에 수행해야 하는 구문들
}
```

예를 들어 숫자를 하나 입력받은 후에 이 숫자가 양수인지, 0인지, 음수인지를 판별하는 경우를 생각해보자. 이 때 생각해 볼 수 있는 조건문은 (숫자가 양수이다) 와 (숫자가 음수이다) 와 (숫자가 0이다)의 세 종류가 필요하다.

```
if (num1 > 0) { // 첫 번째 조건 : num1이 양수인가?
    printf("숫자는 양수입니다.\n");
}
else if (num1 < 0) { // 두 번째 조건 : num1이 양수는 아니고 음수인가?
    printf("숫자는 음수입니다.\n");
}
else { // 마지막 조건 : num1이 양수도 음수도 아닌가?
    printf("숫자는 0입니다.\n");
}
```

그리면 이런 복합 조건문을 이용하는 간단한 프로그램을 만들어보자. 문제는 2개의 숫자를 입력받아 이 숫자 중에 어떤 숫자가 더 큰 수인지를 판정하는 것이다. 이 문제를 해결하기 위해 다음과 같은 순서를 생각하자.

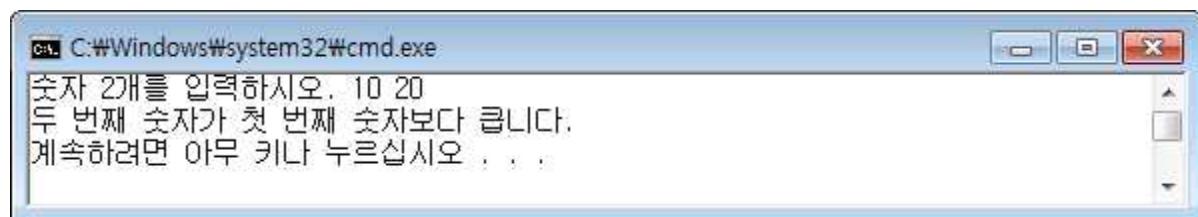
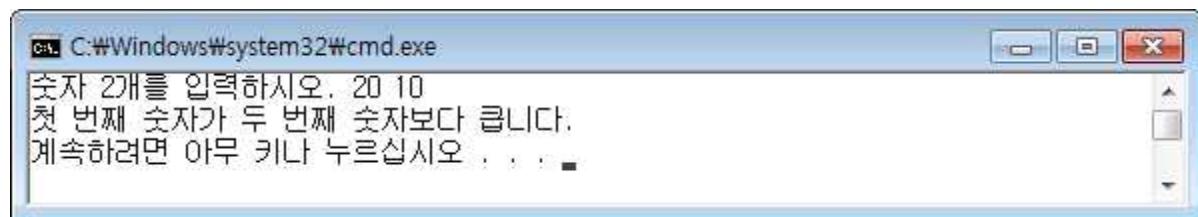
1. 숫자 2개의 값을 입력받는다.

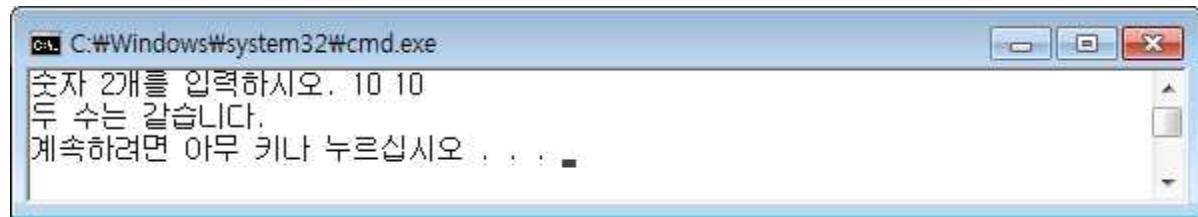
2. 만일 첫 번째 숫자가 두 번째 숫자보다 크면 “첫 번째 숫자가 두 번째 숫자보다 큽니다.”라고 출력한다.
3. 그렇지 않고 만일 두 번째 숫자가 첫 번째 숫자보다 크면 “두 번째 숫자가 첫 번째 숫자보다 큽니다.”라고 출력한다.
4. 위 2번과 3번 조건 모두에 해당되지 않는 경우에는 “두 수는 같습니다.”라고 출력한다.

이 문제를 해결하는 C 프로그램은 다음과 같다.

```
// 예제 ex_C.c
#include <stdio.h>
void main()
{
    int num1, num2;
    printf("숫자 2개를 입력하시오. ");
    scanf("%d %d", &num1, &num2);
    if (num1 > num2) {
        printf("첫 번째 숫자가 두 번째 숫자보다 큽니다.\n");
    }
    else if (num1 < num2) {
        printf("두 번째 숫자가 첫 번째 숫자보다 큽니다.\n");
    }
    else {
        printf("두 수는 같습니다.\n");
    }
}
```

이 프로그램을 수행한 결과는 다음과 같다. 첫 번째 숫자가 더 큰 경우와 두 번째 숫자가 더 큰 경우, 그리고 두 숫자가 같은 경우에 따라 출력 내용이 달라지는 것을 확인할 수 있다.





복합 조건문을 사용할 때에 여러 조건문이 모두 “한 변수의 값이 무엇인가?”로 정해지는 경우에는 이를 스위치(switch ~ case) 구문으로 처리할 수 있다. 예를 들어 등급을 나타내는 문자 변수 **grade**의 값이 각각 'A', 'B', 'C' 인 경우에 따라 출력내용이 달라지는 경우를 생각해보자. 먼저 이 문제를 복합 if 구문으로 처리하면 다음과 같다.

```
if (grade == 'A') {
    printf("참 잘했어요.");
}
else if (grade == 'B') {
    printf("잘했어요.");
}
else if (grade == 'C') {
    printf("좀 더 노력하세요.");
}
else {
    printf("없는 등급입니다.");
}
```

위의 내용과 동일하게 처리하는 스위치 구문은 다음과 같다.

```
switch (grade) {
    case 'A':     printf("참 잘했어요.");
                   break;
    case 'B':     printf("잘했어요.");
                   break;
    case 'C':     printf("좀 더 노력하세요.");
                   break;
    default :    printf("없는 등급입니다.");
}
```

복합 if 구문과 스위치 구문 중에 어떤 것을 사용하는 것이 좋을까? 정답은 자신에게 편한 구문을 사용하라는 것이다. 유의해야 할 점은 스위치 구문에서는 모든 조건문이 문자형 변수나 정수형 변수의 값이 정확하게 어떤 값을 갖느냐로 표현할 수 있을 때에만 사용 가능하다는 것이다. 그리고 각각의 case 구문이 끝나기 전에 break; 구문을 넣어주는 것을 잊지 않아야 한다.

○ 실습 문제

[C01] 나이 계산 및 연령대 판정

태어난 년도를 입력받아 나이를 계산한 후, 나이에 따라 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 그 결과를 출력하라.

단, 나이 = $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아,

7세 이상 ~ 13세미만 : 어린이,

13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년,

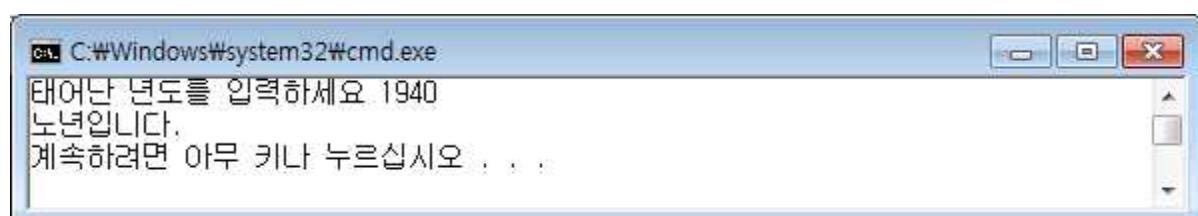
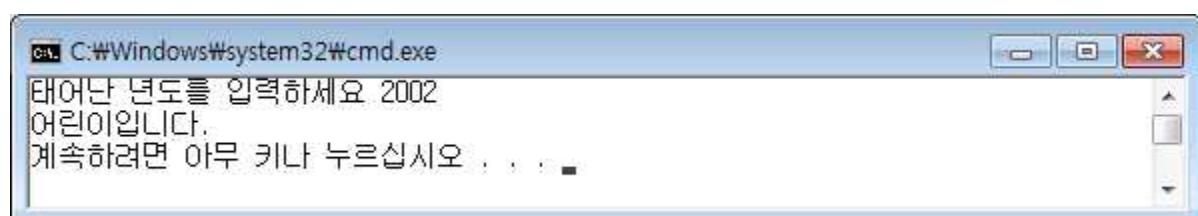
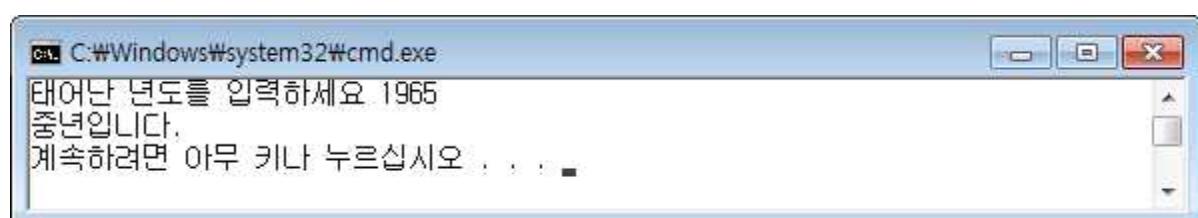
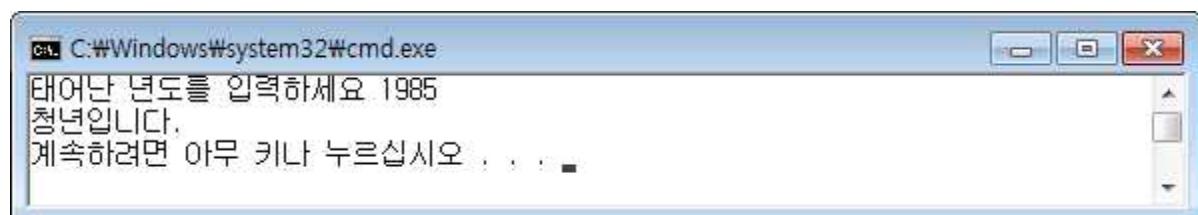
30세 이상 ~ 60세 미만 : 중년,

60세 이상 : 노년

변수는 다음과 같이 사용하라.

```
int birth_year; // 태어난 년도
```

```
int age; // 나이
```



[C02] 물의 온도 구간 판정

물의 온도를 입력받은 후, 이 물이 어느 정도의 온수인지 판정하여 그 결과를 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

음수값 (0미만) : 잘못입력

0도 이상 ~ 25도 미만 : 냉수

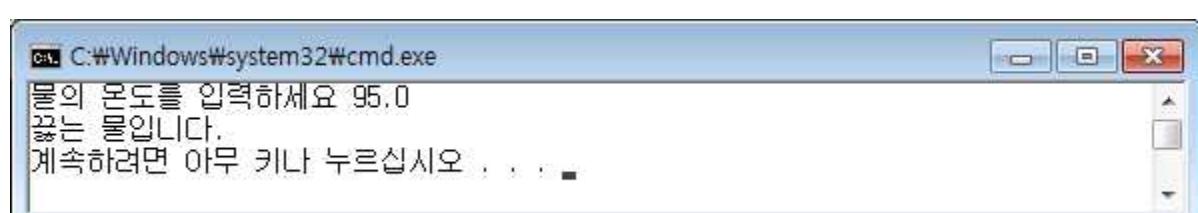
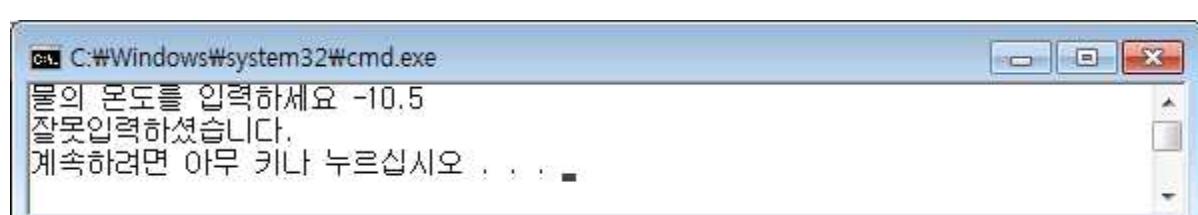
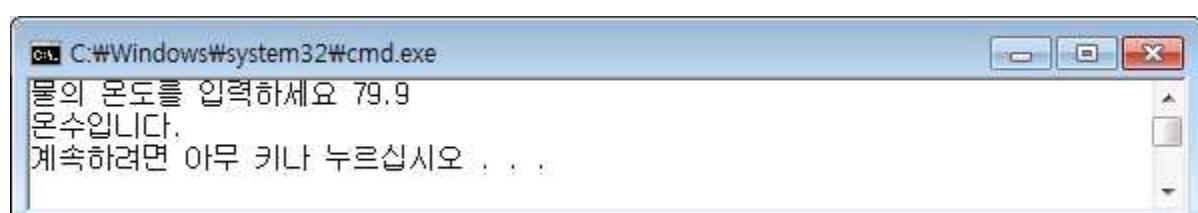
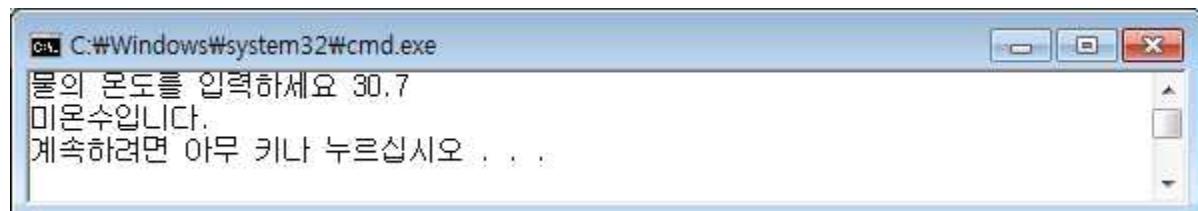
25도 이상 ~ 40도 미만 : 미온수

40도 이상 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

변수는 다음과 같이 사용하라.

```
float input_degree; // 입력받은 온도
```



[C03] 직사각형 형태 판정

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 모양에 대해 평가하는 내용을 출력하라.

단, 평가 내용은 다음 중 1가지 경우로 결정한다.

가로 크기와 세로크기가 동일 : "정사각형입니다."

가로 크기가 세로크기의 2배 이상 : "좌우로 길쭉한 직사각형입니다."

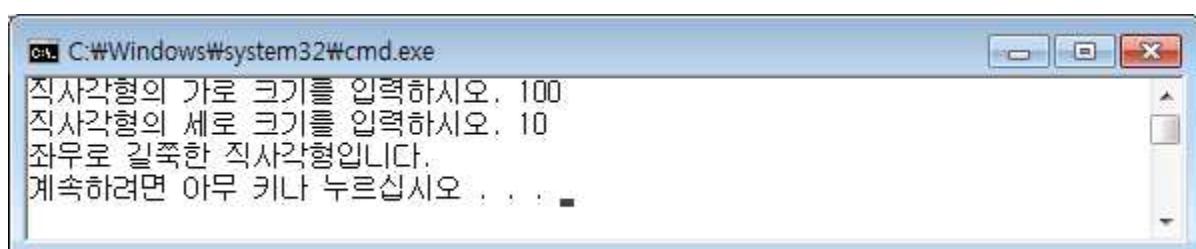
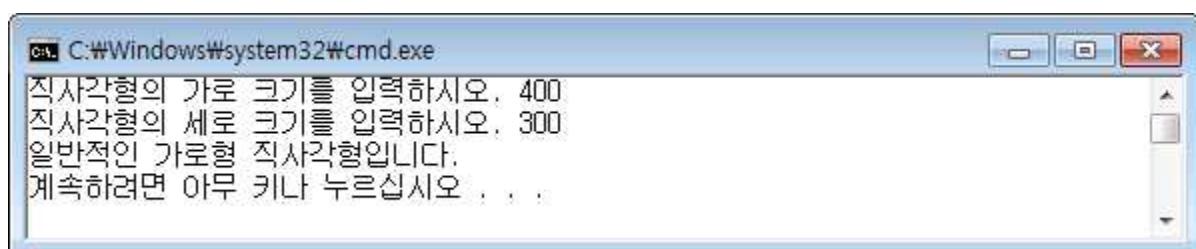
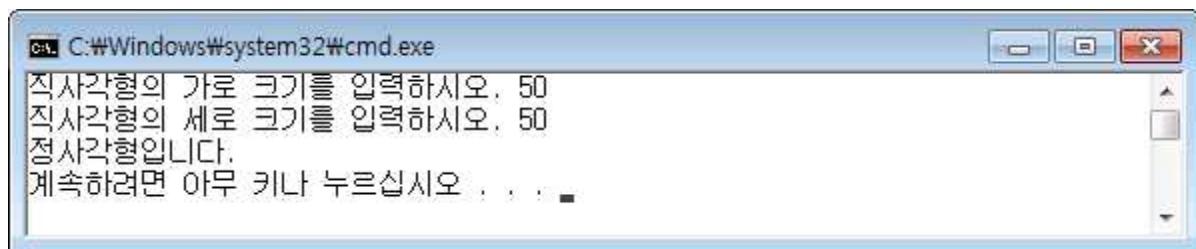
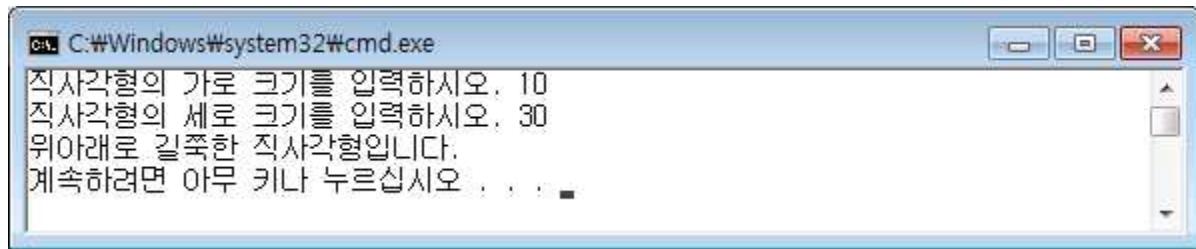
세로 크기가 가로크기의 2배 이상 : "위아래로 길쭉한 직사각형입니다."

가로 크기가 세로크기보다 크면 : "일반적인 가로형 직사각형입니다"

세로 크기가 가로크기보다 크면 : "일반적인 세로형 직사각형입니다"

변수는 다음과 같이 사용하라.

```
int width, height; // 가로크기, 세로크기
```



[C04] 아파트 평형 계산 및 종류 판정

아파트의 분양 면적을 제곱미터(m^2) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하라. 그리고 평형 수에 따라 아파트의 종류를 구분하여 그 결과를 출력하라.

단, 평형 수 = 제곱미터 / 3.305 로 계산하고, 크기에 따른 아파트 종류는 다음과 같이 판정한다.

15평 미만 : 소형

15평 이상 ~ 30평 미만 : 중소형

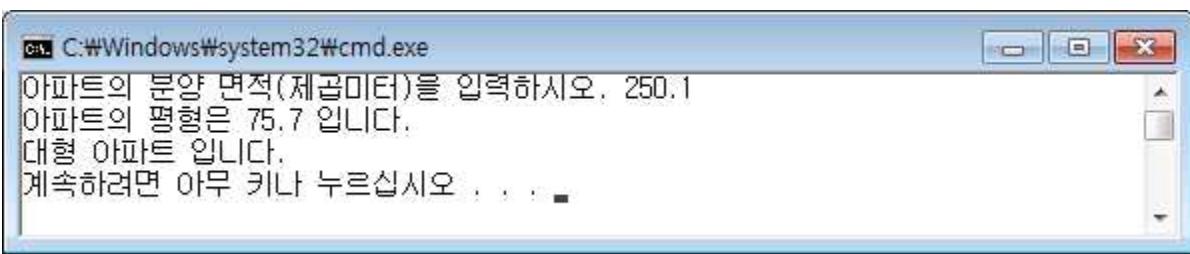
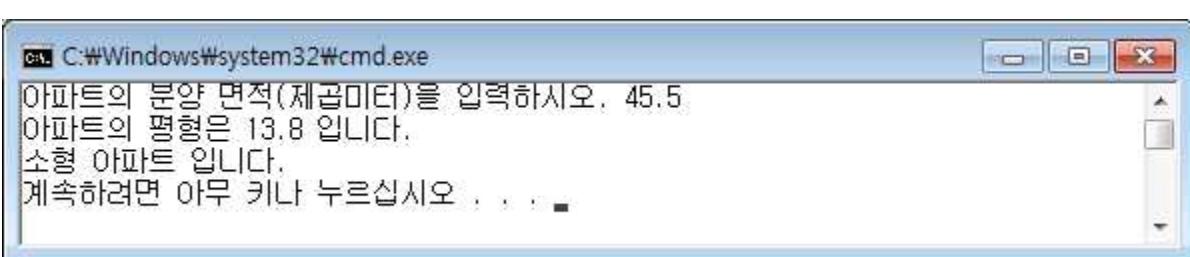
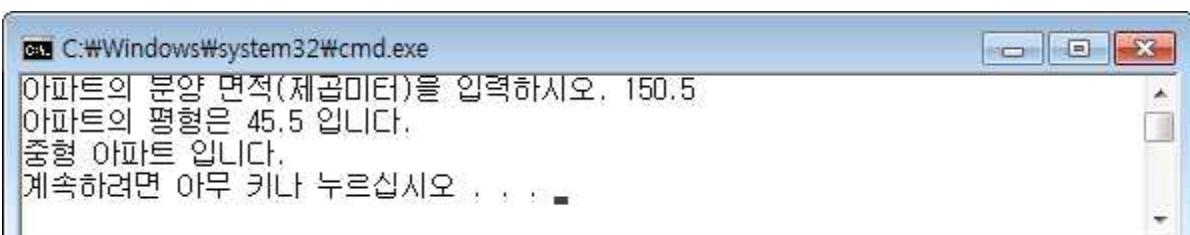
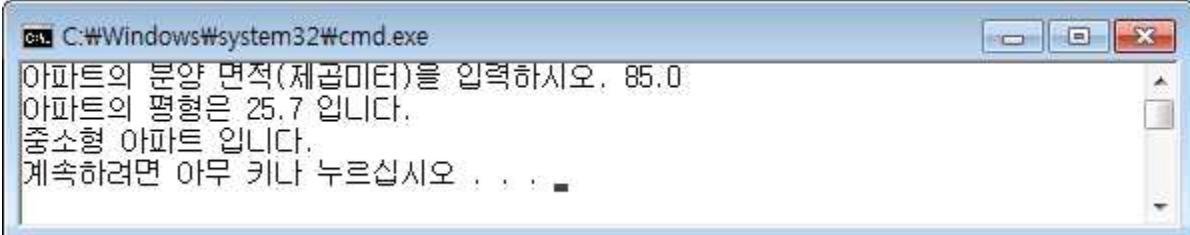
30평 이상 ~ 50평 미만 : 중형

50평 이상 : 대형

변수는 다음과 같이 사용하라.

```
float m2_area; // 면적 (제곱미터)
```

```
float pyung_area; // 면적 (평수)
```



[C05] 연중 날짜 계산

날짜를 월과 일로 입력받아 이 날짜는 1년 중 몇 번째 날에 해당되는지 계산하여 출력하라.

단, 매 월의 날 수는 다음과 같이 정한다.

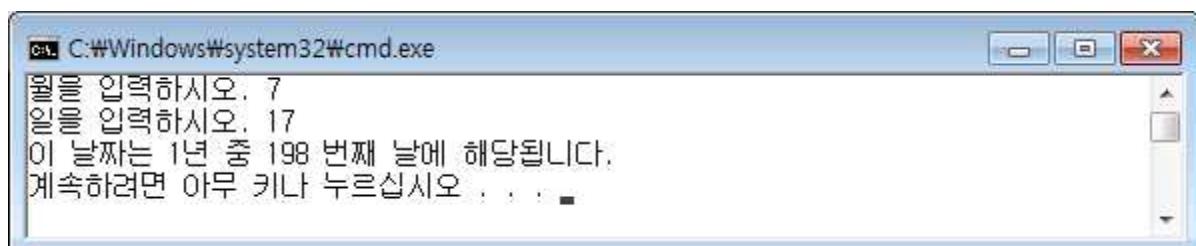
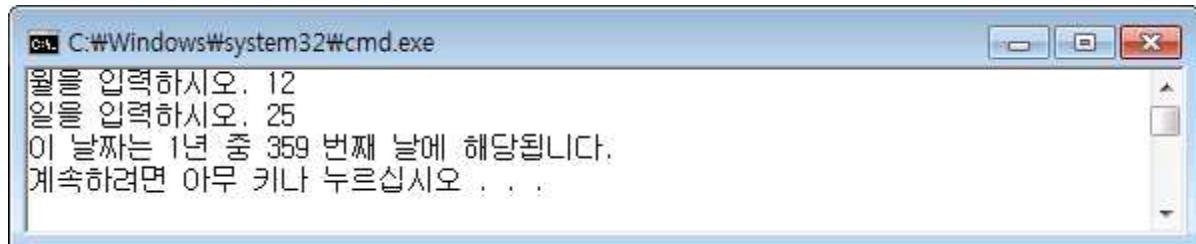
2월 : 28일

1, 3, 5, 7, 8, 10, 12월 : 31일

4, 6, 9, 11월 : 30일

변수는 다음과 같이 사용하라.

```
int month, day;           // 월, 일  
int day_count;           // 1년 중 날 수
```



[C06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하고, 평균에 따라 등급을 정하여 출력하라.
 단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하고 등급은 다음과 같은 기준으로 결정하라.

평균 90이상 : 수

평균 80이상 ~ 90미만 : 우

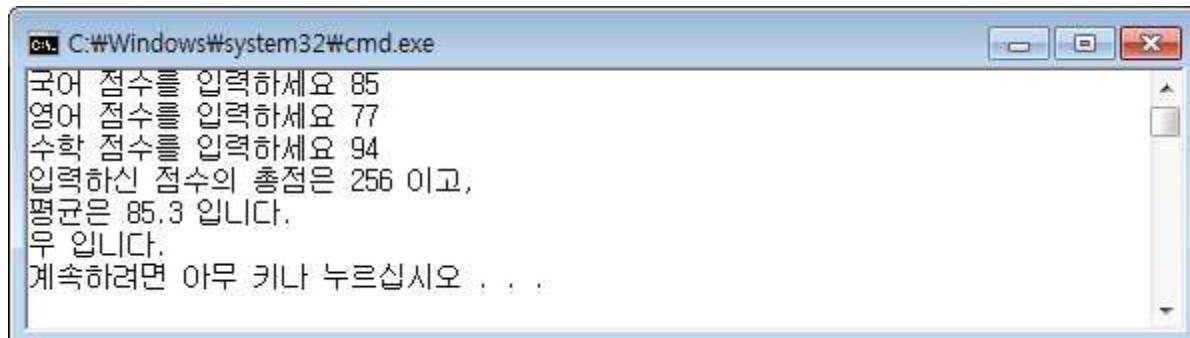
평균 70이상 ~ 80미만 : 미

평균 60이상 ~ 70미만 : 양

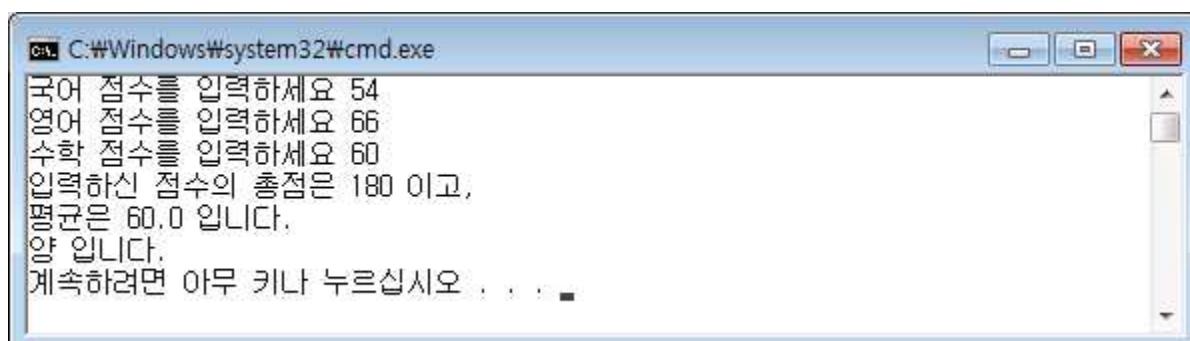
평균 60미만 : 가

변수는 다음과 같이 사용하라.

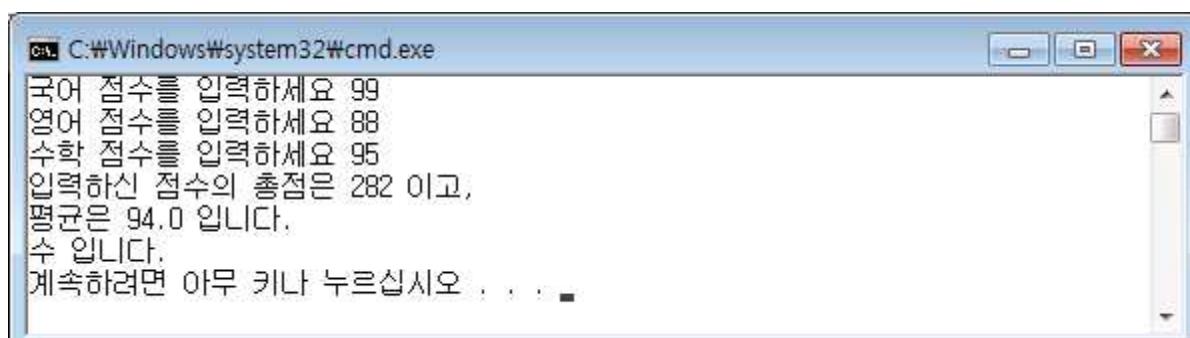
```
int kor, eng, math;           // 국어점수, 영어점수, 수학점수
int total;                   // 총점
float average;               // 평균점수
```



```
C:\Windows\system32\cmd.exe
국어 점수를 입력하세요 85
영어 점수를 입력하세요 77
수학 점수를 입력하세요 94
입력하신 점수의 총점은 256 이고,
평균은 85.3 입니다.
우 입니다.
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
국어 점수를 입력하세요 54
영어 점수를 입력하세요 66
수학 점수를 입력하세요 60
입력하신 점수의 총점은 180 이고,
평균은 60.0 입니다.
양 입니다.
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
국어 점수를 입력하세요 99
영어 점수를 입력하세요 88
수학 점수를 입력하세요 95
입력하신 점수의 총점은 282 이고,
평균은 94.0 입니다.
수 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[C07] 파일 전송 시간 계산

파일 용량을 메가바이트 단위로 입력받고, 전송 방식을 숫자로 입력받아 이에 따라 파일 전송 시간을 초 단위(소수점 1자리)로 계산하여 출력하라.

단, 계산방법은 다음과 같다.

바이트 수 = 메가바이트 수 * 1024 * 1024

Wi-Fi 전송 속도 = 1,500,000바이트 / 초

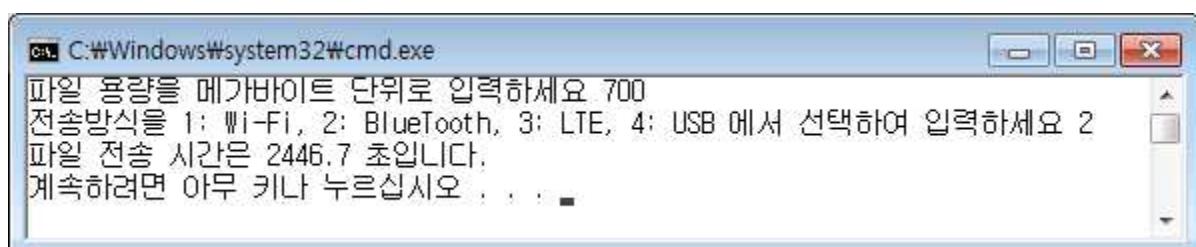
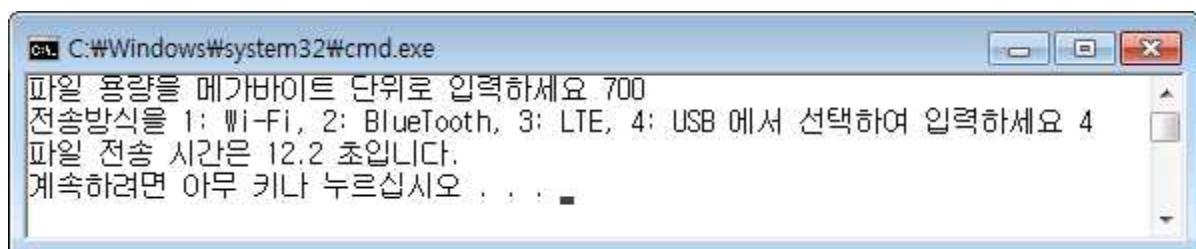
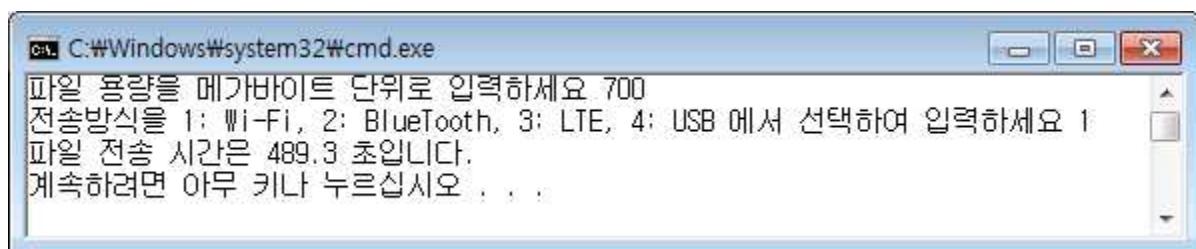
Bluetooth 전송 속도 = 300,000바이트 / 초

LTE 전송 속도 = 1,000,000바이트 / 초

USB 전송 속도 = 60,000,000바이트 / 초

변수는 다음과 같이 사용하라.

```
int megabytes;           // 용량(메가바이트 단위)
long bytes;              // 용량(바이트 단위)
int kind;                // 전송방식 (1: Wi-Fi, 2: BlueTooth, 3: LTE, 4: USB)
float time;               // 전송시간(초 단위)
```

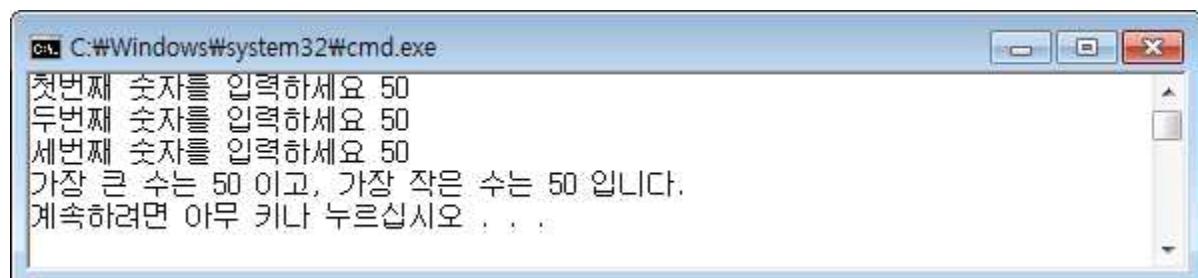
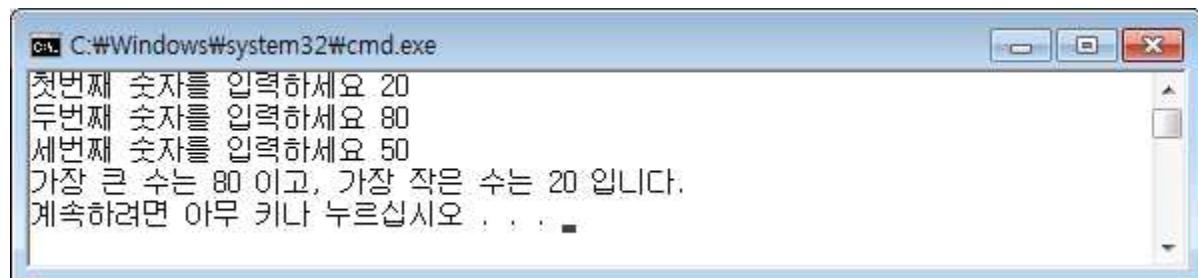


[C08] 3개의 수 중 최댓값과 최솟값 구하기

숫자를 3개 입력받은 후, 이 숫자 중에서 가장 큰 수, 가장 작은 수를 출력하라.

변수는 다음과 같이 사용하라.

```
int num1, num2, num3; // 첫 번째 숫자, 두 번째 숫자, 세 번째 숫자  
int max_num, min_num; // 가장 큰 숫자, 가장 작은 숫자
```



[C09] 소득세 계산

연봉(원 단위)을 숫자로 입력받은 후, 연봉 금액에 대한 소득세를 계산하여 출력하라.

단, 소득세의 금액은 다음과 같이 계산한다.

연봉 1천만 원 미만 : 연봉의 9.5%

연봉 1천만 원 이상 ~ 4천만원미만 : 연봉의 19%

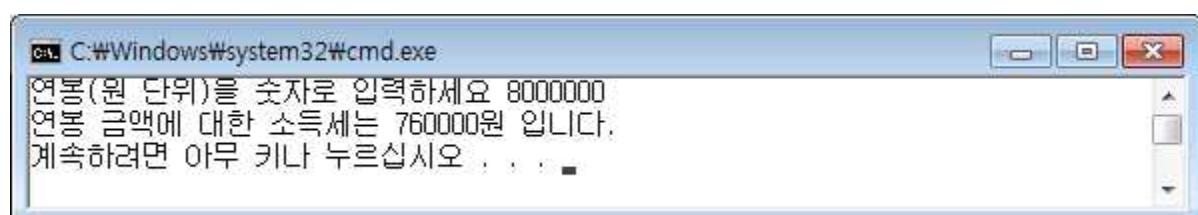
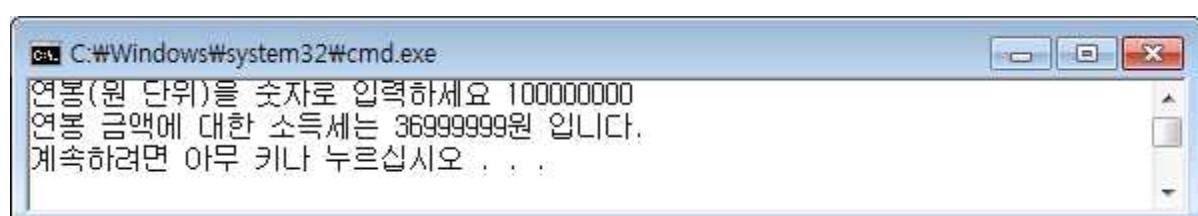
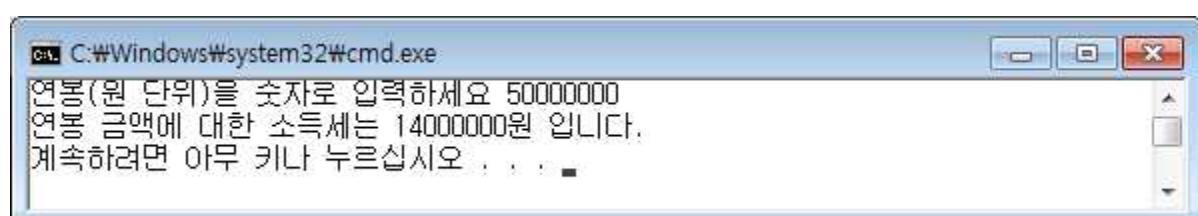
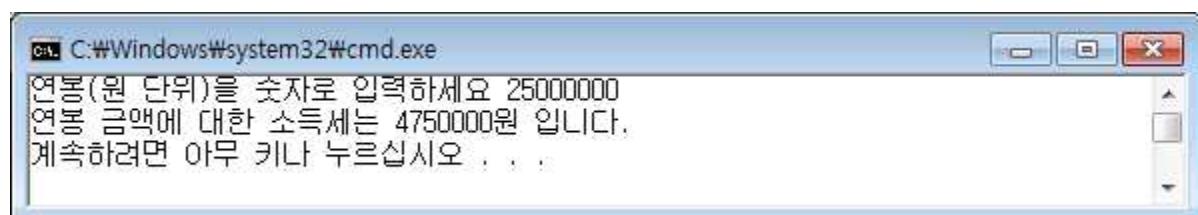
연봉 4천만 원 이상 ~ 8천만원미만 : 연봉의 28%

연봉 8천만 원 이상 : 연봉의 37%

변수는 다음과 같이 사용하라.

```
int income; // 연봉 (원 단위)
```

```
int tax; // 소득세 (원 단위)
```

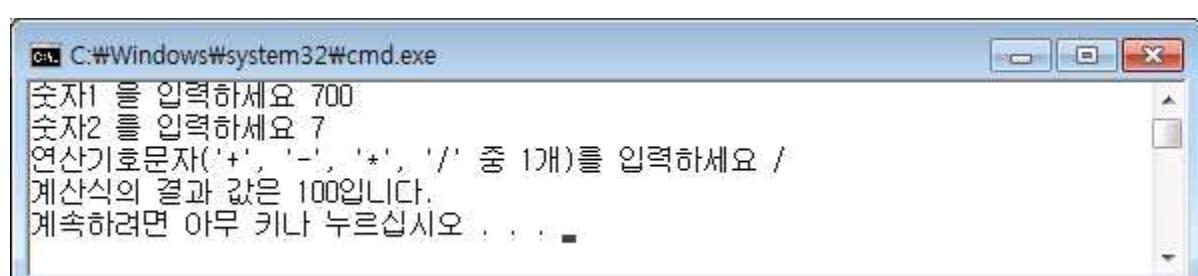
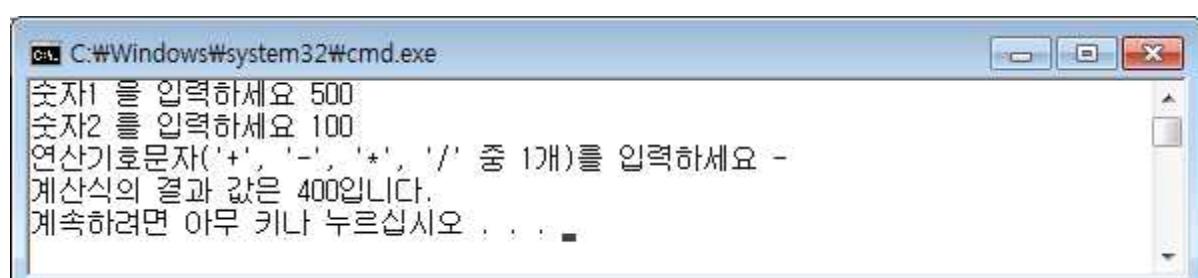
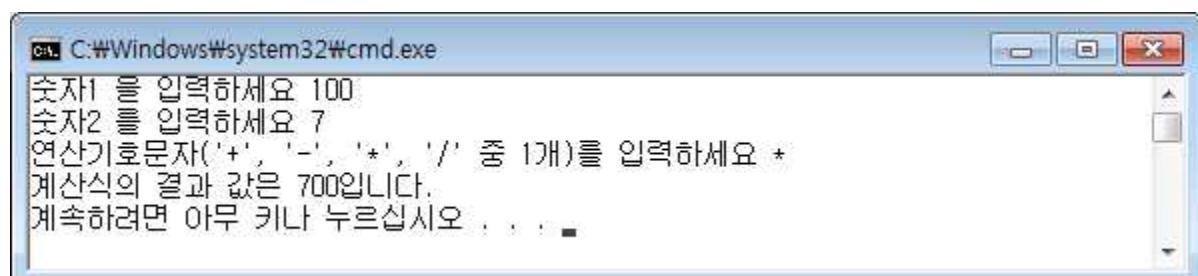
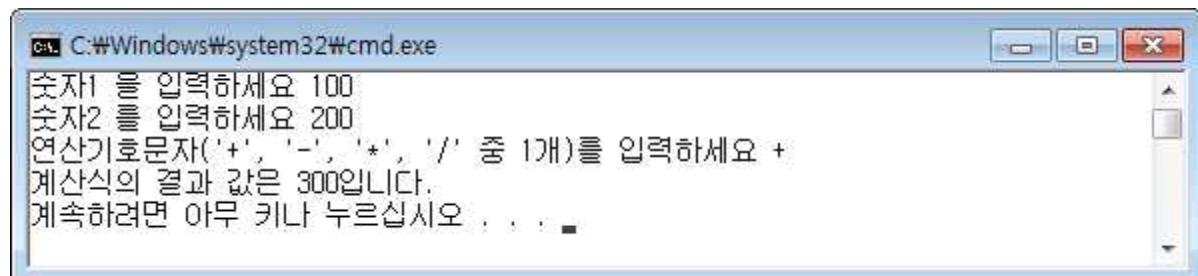


[C10] 간단한 사칙연산 계산기

숫자 2개와 하나의 연산기호문자('+', '-', '*', '/' 중 1개)를 입력받은 후, 첫 번째 숫자와 두 번째 숫자 사이에 연산기호를 넣은 계산식의 결과 값을 계산하여 출력하라.

스위치 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int num1, num2; // 첫 번째 숫자, 두 번째 숫자
char operator; // 연산기호문자('+', '-', '*', '/' 중 1개)
int result; // 연산 결과
```



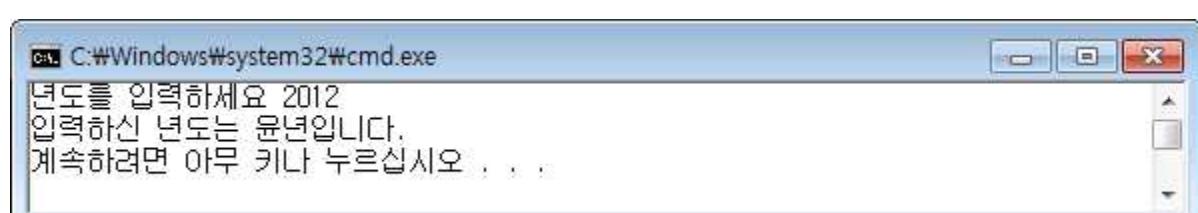
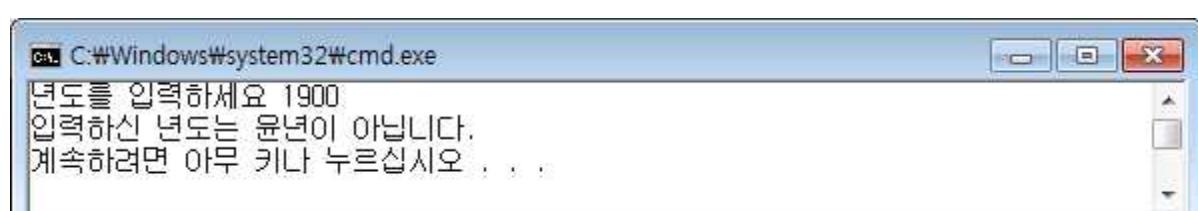
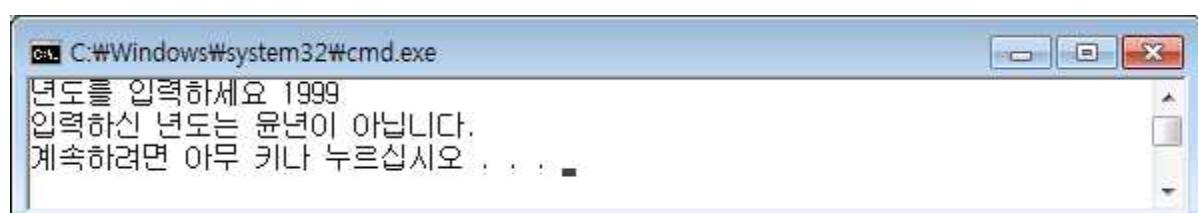
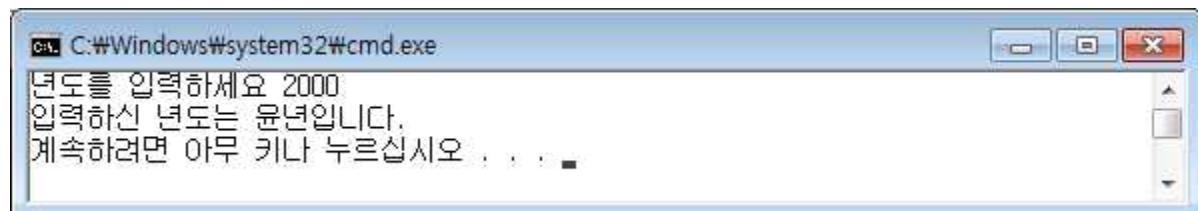
[C11] 윤년 판정하기

년도를 입력받은 후, 이 년도가 윤년이지 아닌지를 판정하여 그 결과를 출력하라. 윤년의 판정 기준은 다음과 같다.

- 1) 년도가 4로 나누어떨어지는 경우에 윤년이다.
- 2) 위 1)의 기준 중에 100으로 나누어떨어지는 년도는 윤년에서 제외한다.
- 3) 위 2)의 기준 중에 400으로 나누어떨어지는 년도는 윤년이다.

변수는 다음과 같이 사용하라.

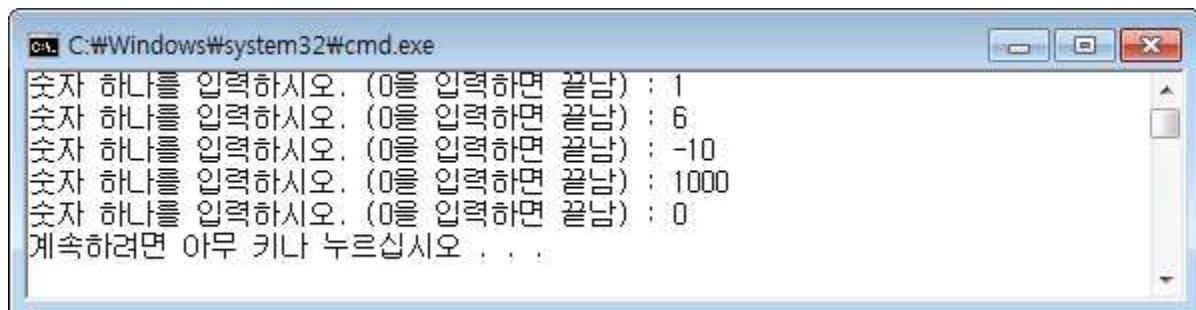
```
int year; // 입력받은 년도
```



[Step D] 반복문 사용하기

컴퓨터 프로그램에서는 동일하거나 비슷한 구문을 처리할 때에 반복문을 사용하면 효과적으로 프로그램을 수행할 수 있다. 가장 간단한 반복문은 `do ~ while` 구문으로 특정 조건이 참인 경우에 무한히 반복시키는 경우에 사용한다. 다음의 예는 0이 입력되기 전까지 계속해서 숫자를 입력받는 프로그램 구문이다. 먼저 숫자를 입력받은 후에 입력된 숫자가 0인지에 따라 반복문의 중단 여부를 결정하게 된다.

```
int number;
do {
    printf("숫자 하나를 입력하시오. (0을 입력하면 끝남) : ");
    scanf("%d", &number);
} while (number != 0); // number의 값이 0이 아닌 동안에 무한 반복
```

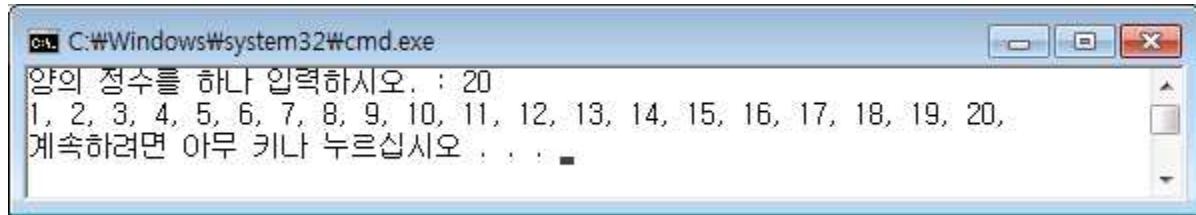


두 번째 반복문의 종류는 `while` 구문이다. 이 구문과 앞의 `do ~ while` 구문과의 차이점은 반복문의 실행 여부를 먼저 결정하느냐 나중에 결정하느냐의 차이이다. 양의 정수를 하나 입력받은 후에 1부터 입력받은 숫자까지의 모든 수를 화면에 출력시키는 문제를 해결하기 위해 `while` 구문을 사용해 보자.

```
int number;
int count=1;
printf("양의 정수를 하나 입력하시오. : ");
scanf("%d", &number);
while (number >= count) { // 1부터 시작하는 count값이 number를 넘지 않는 동안 무한 반복
    printf("%d, ", count);
    count++;
}
printf("\n");
```

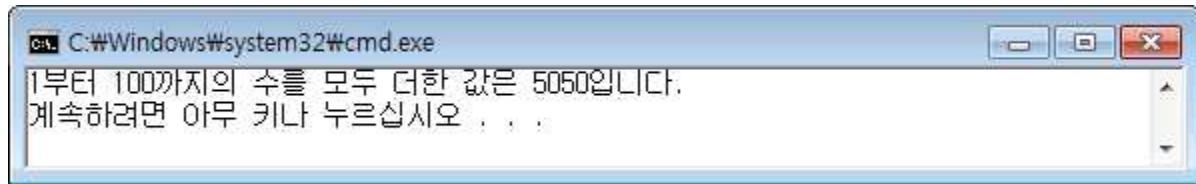
위 프로그램에서는 1보다 작은 값이 입력될 경우를 대비해서 반복문을 실행하기 전에 먼저 수행

여부를 따져야 하기 때문에 **while** 구문을 사용하는 것이 맞는 것이다.



세 번째 반복문의 종류는 **for** 구문으로서 일반적으로 반복 횟수를 미리 알고 있는 경우 또는 특정 범위에 해당되는 숫자만큼 반복해야 하는 경우에 사용한다. 예를 들어 1부터 100까지의 모든 수를 더한 값을 알아내려고 할 때 다음과 같이 프로그램 구문을 만들면 된다.

```
int i;
int sum=0;
for (i=1; i<=100; i++)
{
    sum = sum + i;
}
printf("1부터 100까지의 수를 모두 더한 값은 %d입니다.\n", sum);
```



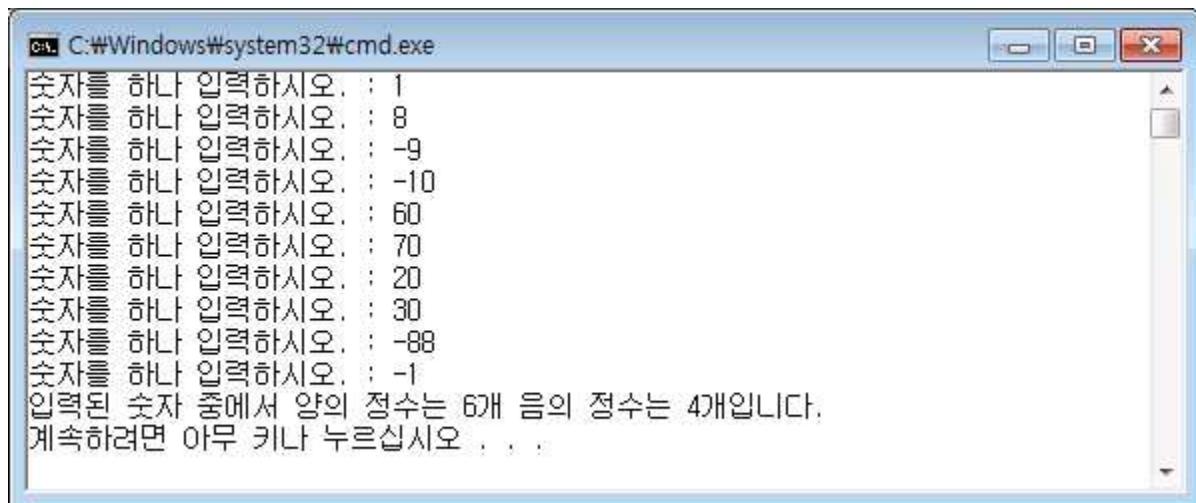
반복문과 조건문을 함께 사용해서 해결하는 문제를 생각해보자. 숫자 10개를 입력받은 후에 이 중에서 양의 정수는 몇 개이고, 음의 정수는 몇 개 인지 세는 문제를 풀어보자. 이 문제를 해결하기 위해서 다음과 같은 순서를 생각해 볼 수 있다.

1. 양의 정수 개수와 음의 정수 개수의 초기 값은 모두 0으로 정한다.
2. 숫자를 10번 입력 받으면서 3번~4번의 작업을 수행한다.
3. 만일 입력된 숫자가 양의 정수이면 양의 정수 개수를 하나 증가시킨다.
4. 그렇지 않고 만일 입력된 숫자가 음의 정수이면 음의 정수 개수를 하나 증가시킨다.
5. 반복문이 끝난 후에 양의 정수 개수와 음의 정수 개수를 출력한다.

위의 순서대로 프로그램을 제작하면 다음과 같다.

```
// 예제 ex_D.c
#include <stdio.h>
void main()
{
    int i;                      // 반복문을 위한 변수
    int number;                  // 입력된 숫자
    int count_plus=0;            // 양의 정수 개수 (초기값 0)
    int count_minus=0;           // 음의 정수 개수 (초기값 0)

    for (i=1; i<=10; i++)
    {
        printf("숫자를 하나 입력하시오. : ");
        scanf("%d", &number);
        if (number > 0)
        {
            count_plus++;
        }
        else if (number < 0)
        {
            count_minus++;
        }
    }
    printf("입력된 숫자 중에서 양의 정수는 %d개 음의 정수는 %d개입니다.\n", count_plus,
count_minus);
}
```



for 문에서는 인덱스로 사용하는 변수를 반복구문 안에서 잘 활용하는 것이 중요하다. 예를 들어

C Workbook

구구단의 5단을 출력하는 프로그램을 만들어보자. 먼저 출력될 모습을 미리 확인해보면서 반복문에 적용할 인덱스의 규칙을 찾아내야 한다.

```
5 × 1 = 5  
5 × 2 = 10  
5 × 3 = 15  
5 × 4 = 20  
5 × 5 = 25  
5 × 6 = 30  
5 × 7 = 35  
5 × 8 = 40  
5 × 9 = 45
```

위의 출력 내용에서 매 줄에서 보이는 숫자는 3부분인데, 첫 번째 5는 모두 동일하고, 두 번째 수는 1부터 9까지 변하는 값이다. 세 번째 수는 첫 번째 수와 두 번째 수를 곱한 값이다. 그렇다면 다음 구문처럼 `for` 구문을 만들 수 있다.

```
for (i=1; i<=9; i++){  
    printf("%d × %d = %d\n", 5, i, 5*i);  
}
```

○ 실습 문제

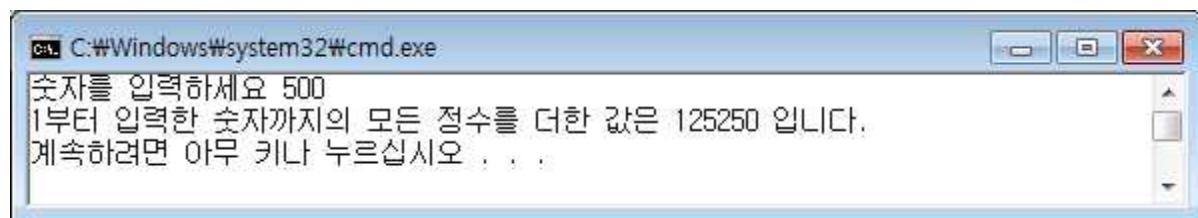
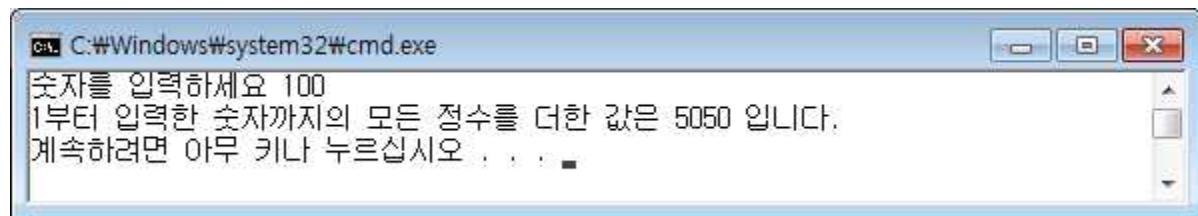
[D01] 1부터 숫자 더하기

숫자를 입력받은 후, 이 숫자가 1보다 큰 경우 1부터 이 숫자까지의 모든 정수를 더한 값을 출력하라.

단, 입력한 숫자가 1 이하이면 "잘못 입력하였습니다."라고 출력한다.

반복은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int number;           // 입력받은 수  
int totalsum;        // 1부터 더한 계산 결과 값  
int i;               // 반복문 사용을 위한 변수
```

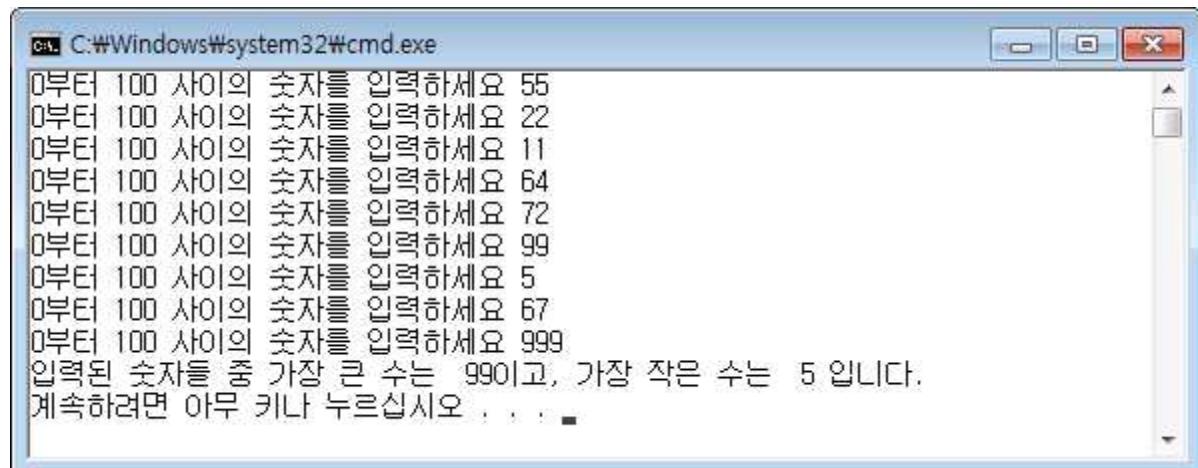


[D02] 입력 받은 숫자들 중에서 가장 큰 수와 가장 작은 수 구하기

반복해서 0부터 100 사이의 숫자를 입력받아서 지금까지 입력된 숫자들 중 가장 큰 수와 가장 작은 수가 무엇인지 출력하라. 0부터 100 사이의 숫자가 아닌 수가 입력되면 반복문이 중단되도록 하라.

반복은 do ~ while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int number;           // 입력받은 수  
int max_num, min_num; // 가장 큰 숫자, 가장 작은 숫자
```

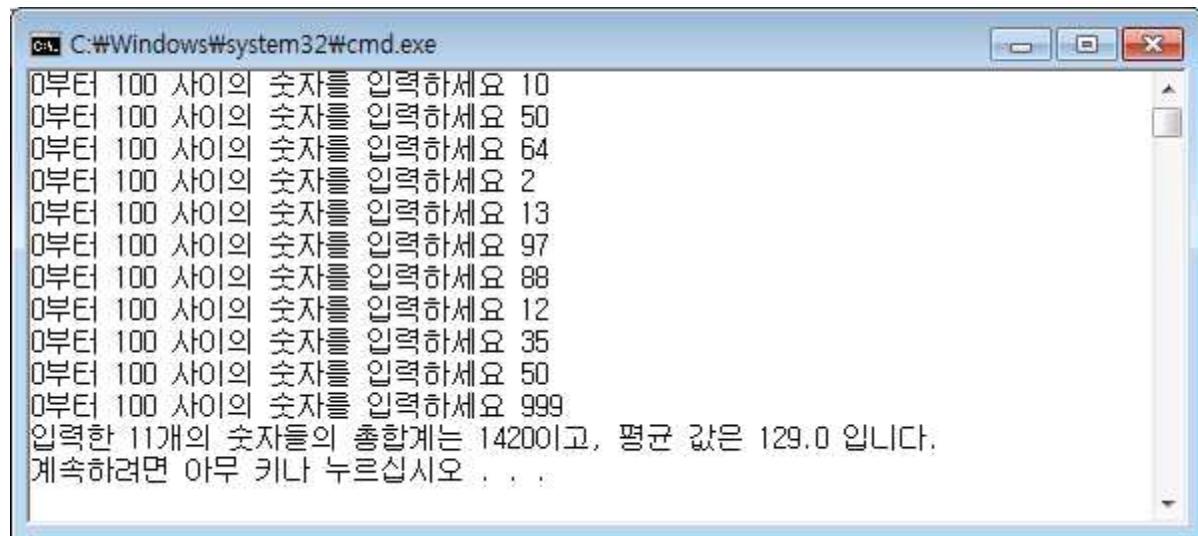


[D03] 입력 받은 숫자들의 총합계와 평균 값 구하기

반복해서 0부터 100 사이의 숫자를 입력받아서 지금까지 입력된 숫자들이 모두 몇 개이며, 이 숫자들의 총 합계와 평균 값을 계산하여 출력하라. 0부터 100 사이의 숫자가 아닌 수가 입력되면 반복문이 중단되도록 하라.

반복은 do ~ while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int number;           // 입력받은 수
int count=0;          // 입력받은 숫자의 개수
int totalsum;         // 총합계
float average;        // 평균 값
```



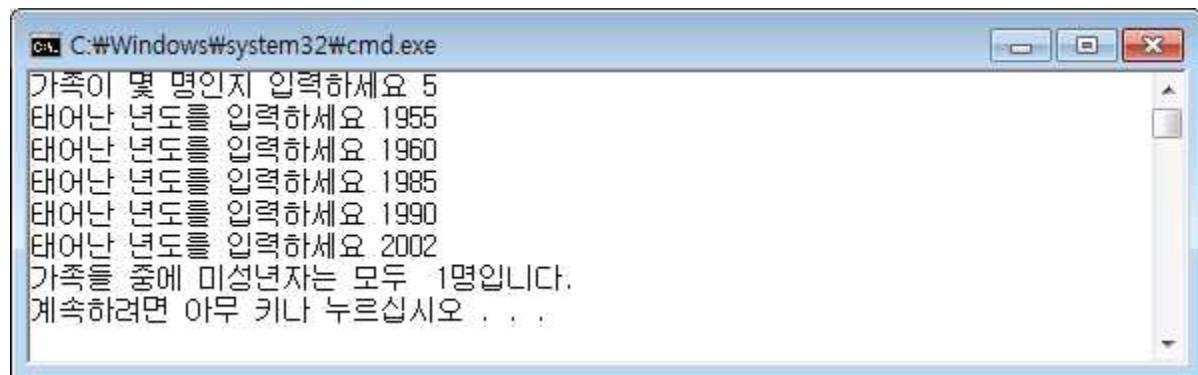
[D04] 미성년자 숫자 세기

가족이 몇 명인지 입력받은 후, 그 인원 수 만큼 태어난 연도를 입력받으면서, 나이를 계산하여 미성년자가 모두 몇 명인지 출력하라.

단, 나이 = $2012 - \text{태어난 연도} + 1$ 로 계산하고 나이가 20세 미만인 경우, 미성년자로 판정한다.

반복은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int count_all;      // 가족 인원수  
int count_young;    // 미성년자의 수  
int birth_year;     // 태어난 연도  
int age;            // 나이  
int i;              // 반복문을 위한 변수
```



[D05] 직사각형 형태 개수 세기

반복해서 직사각형의 가로크기와 세로크기를 입력받으면서 이 값을 이용하여 직사각형의 모양을 판정하여 각각의 종류별로 몇 개가 입력되었는지 결과를 출력하라. 가로 크기와 세로 크기 중 하나라도 0이하의 값이 입력되면 반복을 중단한다.

단, 평가 기준은 다음과 같다.

가로 크기와 세로크기가 동일 : "정사각형"

가로 크기가 세로크기의 2배 이상 : "좌우로 길쭉한 직사각형"

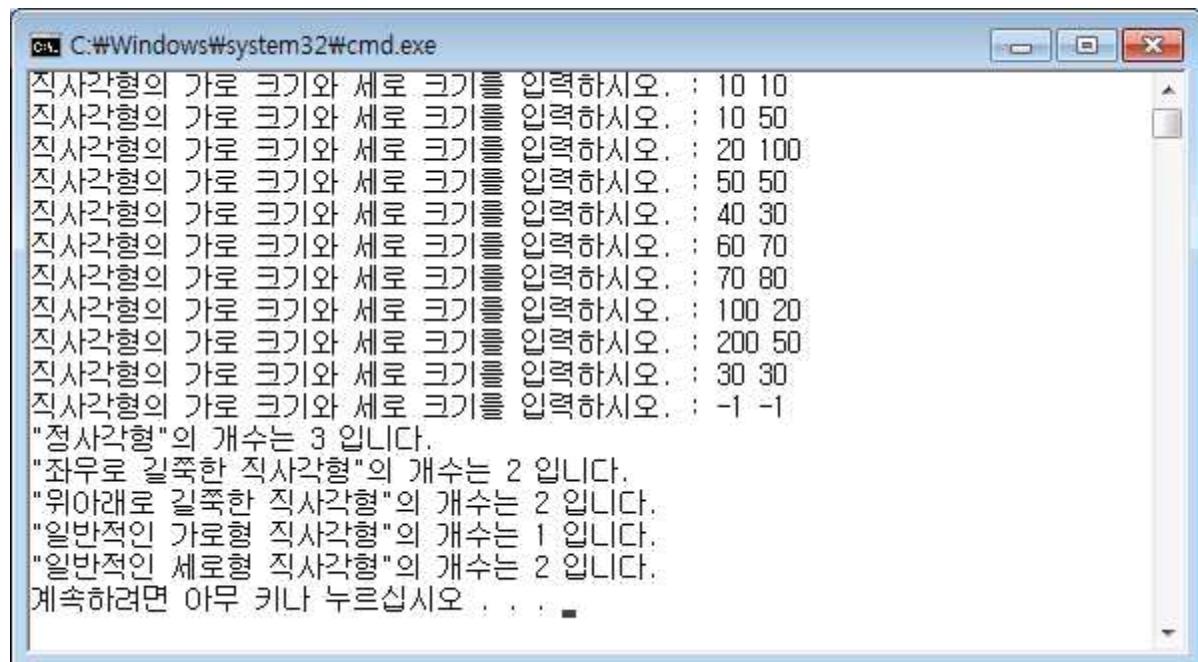
세로 크기가 가로크기의 2배 이상 : "위아래로 길쭉한 직사각형"

가로 크기가 세로크기보다 크면 : "일반적인 가로형 직사각형"

세로 크기가 가로크기보다 크면 : "일반적인 세로형 직사각형"

반복은 do ~ while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int width, height; // 가로크기, 세로크기
int count1; // "정사각형"의 개수
int count2; // "좌우로 길쭉한 직사각형"의 개수
int count3; // "위아래로 길쭉한 직사각형"의 개수
int count4; // "일반적인 가로형 직사각형"의 개수
int count5; // "일반적인 세로형 직사각형"의 개수
```



[D06] 아파트 평형 계산 및 종류 판정

아파트 10채의 분양 면적을 제곱미터(m^2) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하여 평형 수에 따라 아파트의 종류를 구분하여 종류별로 개수를 센 후, 그 결과를 출력하라.
단, 평형 수 = 제곱미터 / 3.305 로 계산하고, 크기에 따른 아파트 종류는 다음과 같이 판정한다.

15평 미만 : 소형

15평 이상 ~ 30평 미만 : 중소형

30평 이상 ~ 50평 미만 : 중형

50평 이상 : 대형

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
float m2_area;      // 면적 (제곱미터)  
float pyung_area;  // 면적 (평수)  
int count1;        // 소형 아파트 개수  
int count2;        // 중소형 아파트 개수  
int count3;        // 중형 아파트 개수  
int count4;        // 대형 아파트 개수  
int i;              // 반복문을 위한 변수
```

The screenshot shows a Windows Command Prompt window titled 'cmd.exe' running on a Windows system. The window displays a series of inputs and outputs related to apartment calculations. The user inputs various square meter values, and the program calculates the corresponding pyung value and the type of apartment (small, medium-small, medium, or large) based on the pyung value. The program also counts the number of each apartment type. The output ends with a prompt to continue input.

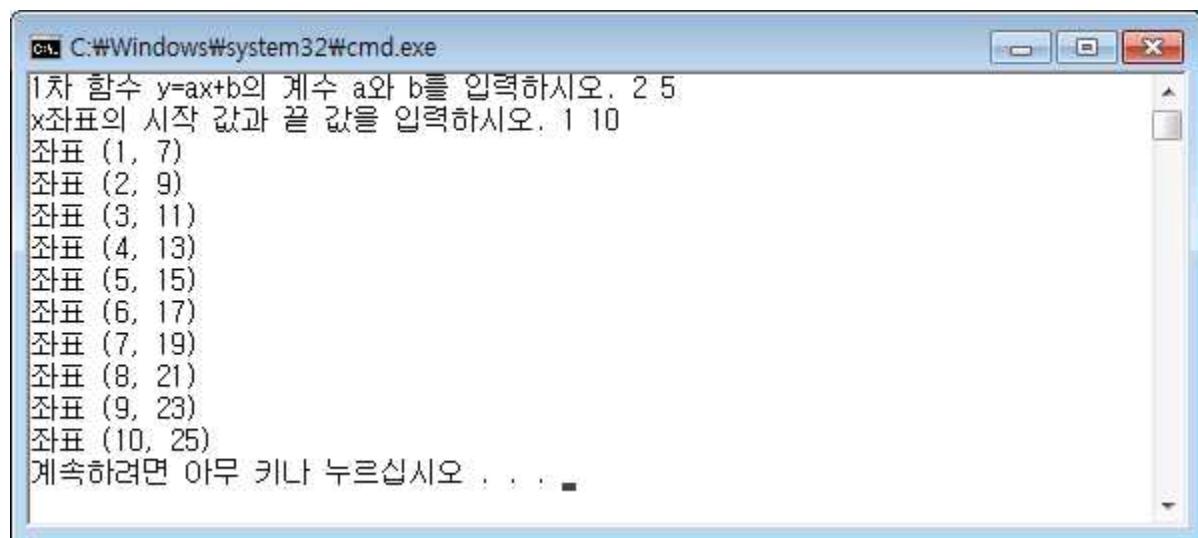
```
C:\Windows\system32\cmd.exe  
아파트의 분양 면적(제곱미터)을 입력하시오. 85.5  
→ 이 아파트의 평형은 25.9 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 120.4  
→ 이 아파트의 평형은 36.4 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 200.0  
→ 이 아파트의 평형은 60.5 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 40.5  
→ 이 아파트의 평형은 12.3 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 55  
→ 이 아파트의 평형은 16.6 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 155  
→ 이 아파트의 평형은 46.9 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 77.7  
→ 이 아파트의 평형은 23.5 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 99.9  
→ 이 아파트의 평형은 30.2 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 65.0  
→ 이 아파트의 평형은 19.7 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 250.1  
→ 이 아파트의 평형은 75.7 입니다.  
"소형 아파트"의 개수는 1 입니다.  
"중소형 아파트"의 개수는 4 입니다.  
"중형 아파트"의 개수는 3 입니다.  
"대형 아파트"의 개수는 2 입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

[D07] 1차 함수의 좌표 구하기

1차 함수 $y=ax+b$ 에 대해 계수 a 와 b 를 입력받은 후, x 값의 시작 값과 마지막 값을 입력받아 이 두 수 사이의 x 값에 대한 1차 함수의 (x, y) 좌표들을 출력하라.

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int a, b;           // 1차 함수의 계수 a, b  
int x_begin, x_end; // x좌표의 시작 값과 끝 값  
int x, y;          // x좌표, y좌표
```

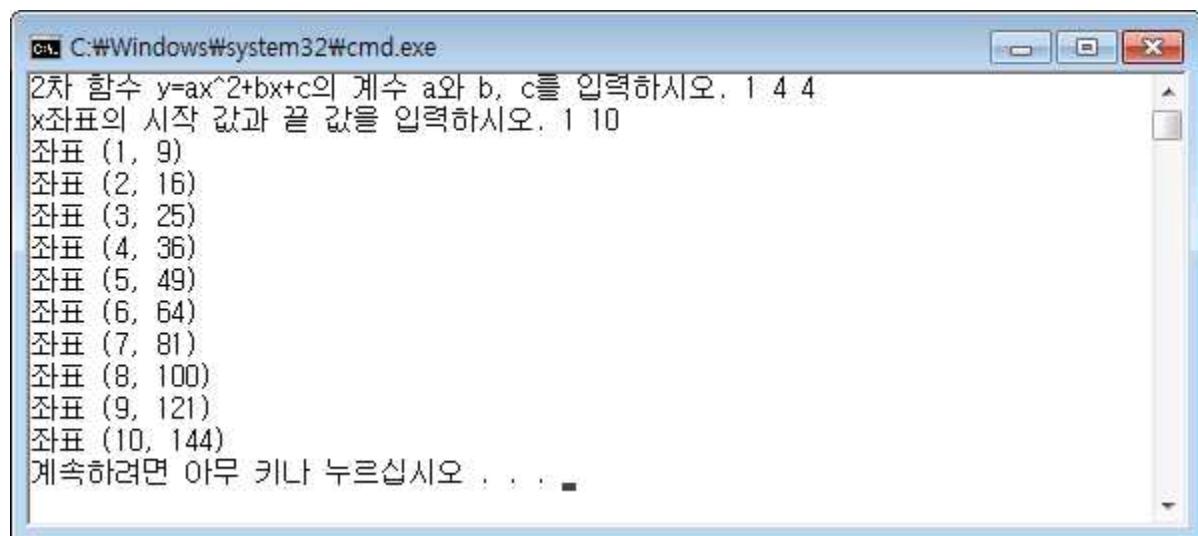


[D08] 2차 함수의 좌표 구하기

2차 함수 $y=ax^2 + bx + c$ 에 대해 계수 a 와 b 와 c 를 입력받은 후, x 값의 시작 값과 마지막 값을 입력받아 이 두 수 사이의 x 값에 대한 2차 함수의 (x, y) 좌표들을 출력하라.

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

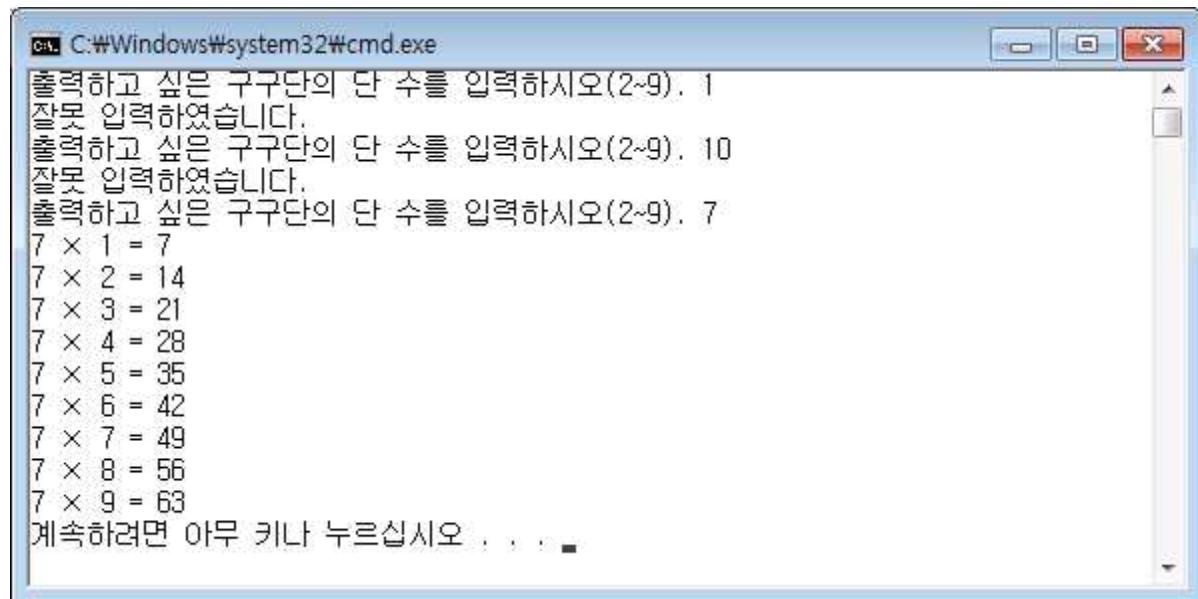
```
int a, b, c;           // 2차 함수의 계수 a, b, c  
int x_begin, x_end;   // x좌표의 시작 값과 끝 값  
int x, y;             // x좌표, y좌표
```



[D09] 원하는 구구단의 단 출력하기

2부터 9 사이의 숫자를 입력받아 이 숫자에 해당하는 구구단을 출력하라. 단, 2부터 9 사이의 숫자가 아닌 수를 입력하면 "잘못 입력하였습니다."라고 출력하고 바르게 입력할 때까지 다시 입력을 받도록 하라.
반복문은 적당하게 선택하고, 변수는 다음과 같이 사용하라.

```
int dan;           // 출력하려는 구구단의 단 수  
int i;             // 반복문을 위한 변수
```



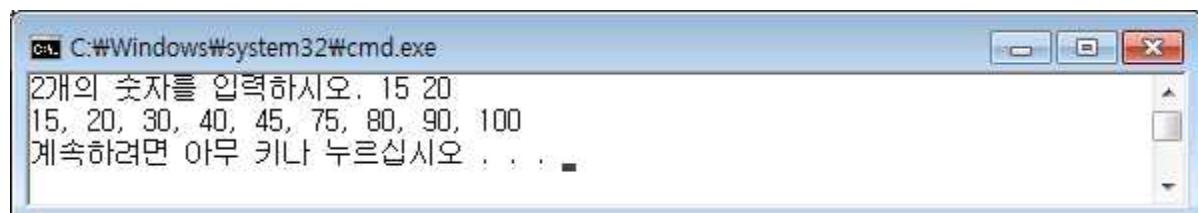
[D10] 두 수의 배타적 배수 출력하기

숫자 2개를 입력받은 후, 1부터 100까지의 숫자 중에 이 두 숫자 중 하나의 숫자에 대해서만 배수인 수를 모두 출력하라. 즉, 두 숫자의 공통 배수인 숫자는 출력하지 않아야 한다.

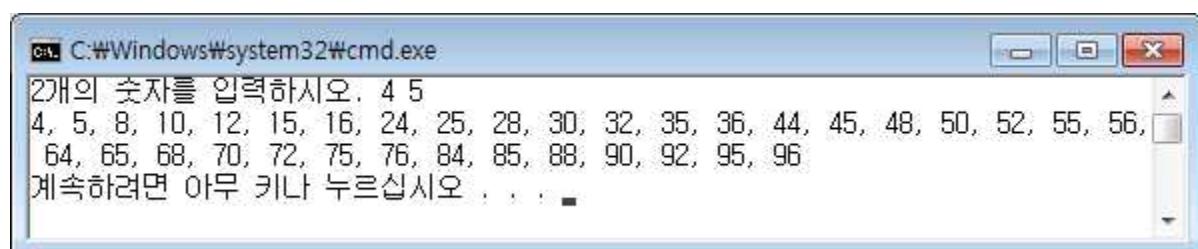
예를 들어 15와 20을 입력하게 되면 "15, 20, 30, 40, 45, 75, 80, 90, 100"이 출력된다.

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
int num1, num2;           // 입력받은 두 수  
int i;                   // 반복문을 위한 변수
```



```
C:\Windows\system32\cmd.exe  
2개의 숫자를 입력하시오. 15 20  
15, 20, 30, 40, 45, 75, 80, 90, 100  
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe  
2개의 숫자를 입력하시오. 4 5  
4, 5, 8, 10, 12, 15, 16, 24, 25, 28, 30, 32, 35, 36, 44, 45, 48, 50, 52, 55, 56,  
64, 65, 68, 70, 72, 75, 76, 84, 85, 88, 90, 92, 95, 96  
계속하려면 아무 키나 누르십시오 . . .
```

영역 2 : 프로그래밍 제어구조 응용

영역 2에서는 영역 1에서 습득한 프로그래밍의 기본 제어구조를 응용하는 방법을 연습한다. 먼저 2단계 이상이 중복된 복합 반복문을 사용하는 방법과 뒷자료 형식인 배열을 다루는 방법을 연습 한다. 또한 그동안 연습한 조건문과 반복문이 다양하게 적용되는 응용 예제들을 해결하는 연습을 하게 된다. 마지막으로 영역 3으로 진행하기 위한 준비단계로서 C언어에서 제공하는 주요 함수들을 사용하는 연습을 하게 된다. 영역 2는 다음과 같이 4개의 단계로 구성된다.

Step E : 복합 반복문 사용하기

Step F : 배열 사용하기

Step G : 조건과 반복을 활용하는 응용 예제 해결하기

Step H : C언어의 주요 함수 사용하기

[Step E] 복합 반복문 사용하기

이번 단계에서는 반복문을 2개 이상 겹쳐서 사용해야 하는 복합 반복문을 연습하려고 한다. 2개의 `for` 구문을 사용하는 이중 반복문의 사용법을 [Step D]에서 다룬 구구단 문제를 통해 알아보도록 하자. 구구단 중에서 5단을 출력하는 구문은 다음과 같았다.

```
for (i=1; i<=9; i++){
    printf("%d × %d = %d\n", 5, i, 5*i);
}
```

2단부터 9단까지의 구구단을 출력하는 첫번째 방법은 다음과 같이 위의 구문에서 5단을 출력하는 `for` 문장을 2단부터 9단까지 반복하는 것이다.

```
for (i=1; i<=9; i++){
    printf("%d × %d = %d\n", 2, i, 2*i);           // 2단 출력
}
for (i=1; i<=9; i++){
    printf("%d × %d = %d\n", 3, i, 3*i);           // 3단 출력
}
.... 중략 ...                                     // 4~7단 출력
for (i=1; i<=9; i++){
    printf("%d × %d = %d\n", 8, i, 8*i);           // 8단 출력
}
for (i=1; i<=9; i++){
    printf("%d × %d = %d\n", 9, i, 9*i);           // 9단 출력
}
```

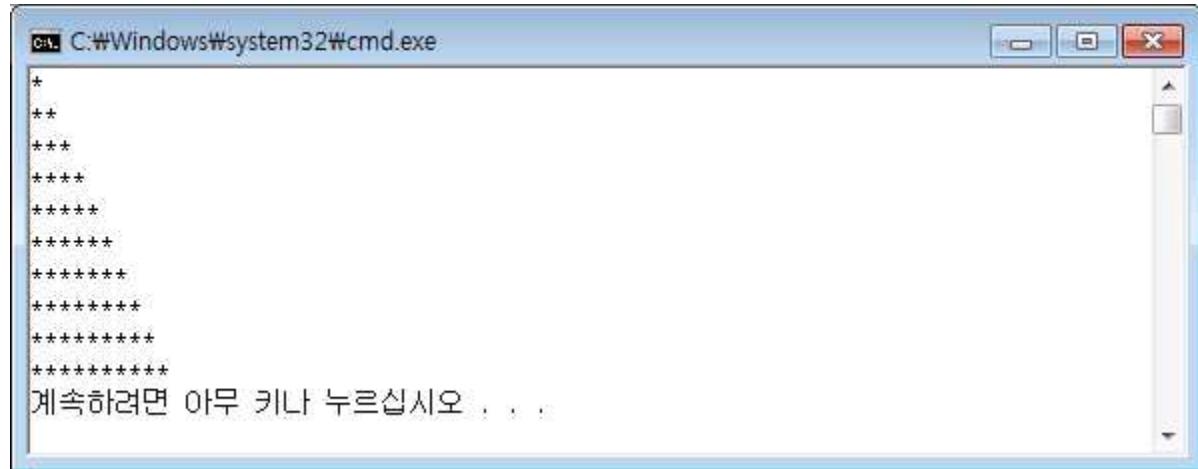
위의 구문들을 잘 살펴보면 각각의 `for` 구문에서 달라지는 부분은 2부터 9까지 변하는 두 군데이다. 이제는 위 9개의 다시 하나의 반복문으로 묶는 일만 남았다. 다음 구문을 잘 살펴보면서 이중 반복문을 이해하도록 하자.

```
for (dan=2; i<=9; dan++){
    for (i=1; i<=9; i++){
        printf("%d × %d = %d\n", dan, i, dan*i);   // dan 단 출력
    }
}
```

2개의 `for` 구문 중에서 바깥의 반복문은 2단부터 9단까지 증가하는 반복을 처리하고 있으며, 안쪽

의 반복문은 각각의 단을 출력하는데 있어서 곱해지는 수 1부터 9까지를 반복하고 있다. 이중 반복문에서는 바깥쪽의 반복문 인덱스와 안쪽의 반복문 인덱스를 바르게 결정하는 것이 가장 중요하다고 할 수 있다.

확실한 이해를 위해 한 문제를 더 풀어보자. 다음과 같이 삼각형 모양의 별을 화면에 출력하려면 어떻게 해야 할까?



이런 경우에 무엇을 바깥쪽 반복문으로 처리하고, 무엇을 안쪽 반복문으로 처리해야 할지를 찾아야 한다. 출력된 모양을 자세히 살펴보면 다음과 같은 규칙을 찾아낼 수 있다.

첫 줄에는 별 1개, 두 번째 줄에는 별 2개, 세 번째 줄에는 별 3개, ... 이렇게 해서 10번째 줄에는 별 10개가 출력된다. 이를 `for` 구문으로 표현하면 다음과 같다.

```
for (count=1;count <= 10 ;count++)  
{  
    별을 count개 출력한다.  
}
```

그렇다면 별을 count개 출력하려면 어떻게 해야 할까? 별을 여러 개 출력한 다음에는 반드시 다음 줄로 내리는 개행문자('\n')를 출력해 주어야 한다. 다음과 같이 하면 될 것이다.

```
for (star=1;star <= count ;star++)
{
    printf("*");
}
printf("\n");
```

이제는 위 2개의 for 구문을 하나의 프로그램으로 묶어주면 된다.

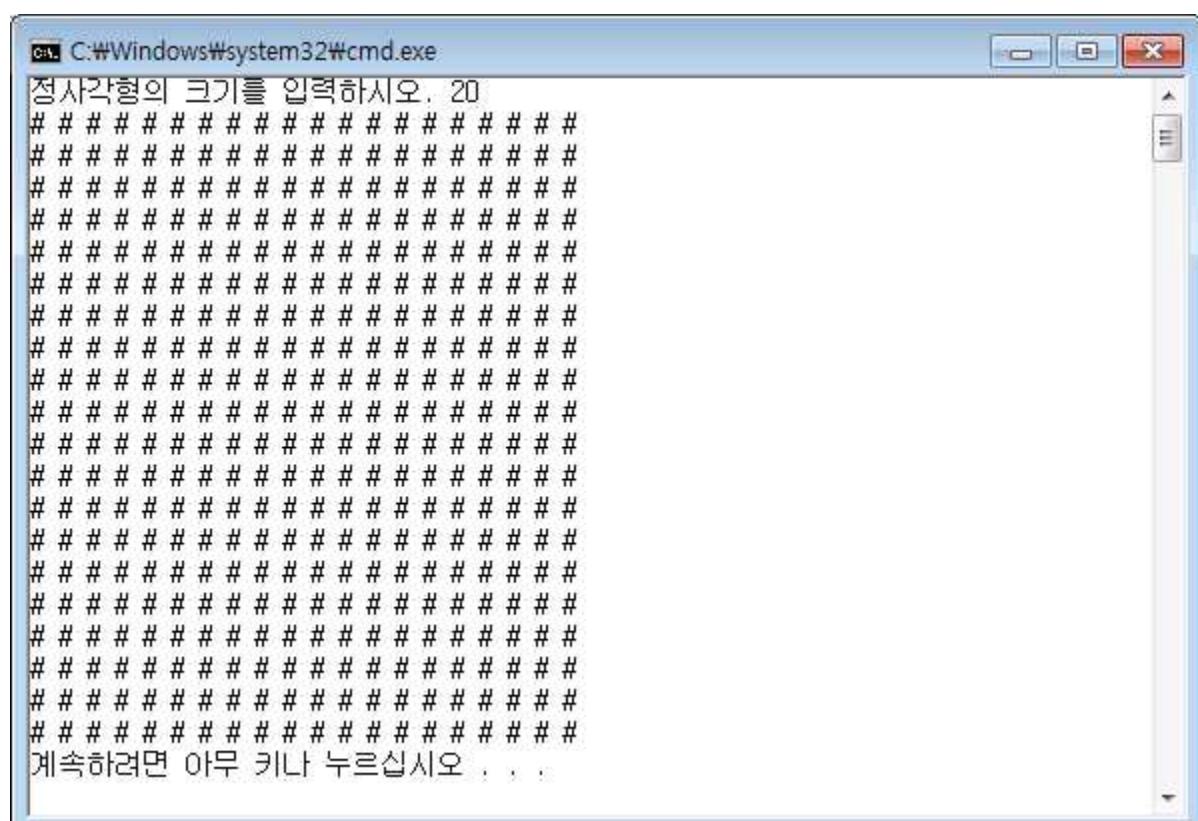
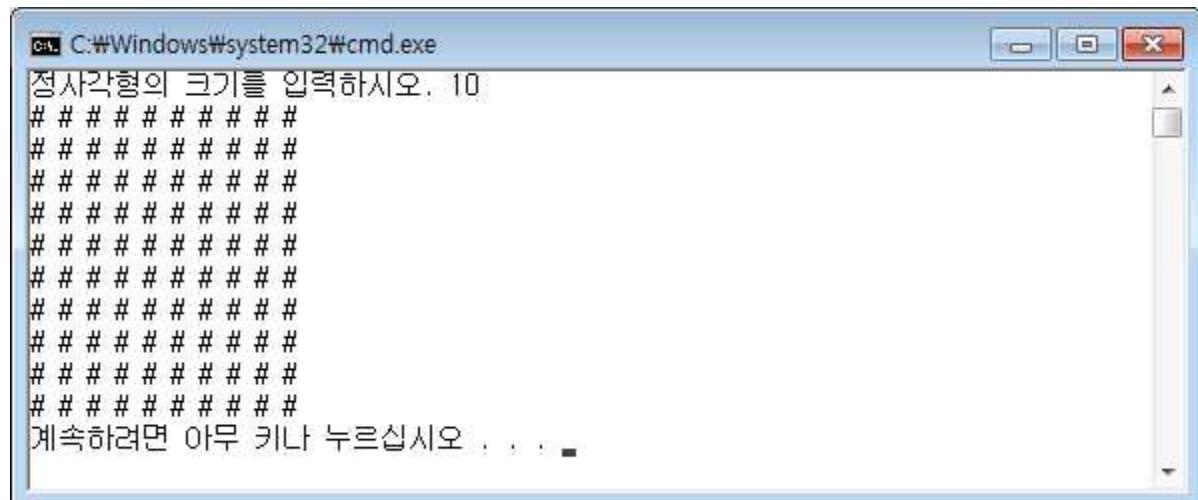
C Workbook

```
// 예제 ex_E.c
#include <stdio.h>
void main()
{
    int count, star;
    for (count=1;count <= 10 ;count++)
    {
        for (star=1;star <= count ;star++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

[E01] 입력한 숫자 크기의 정사각형 출력하기

숫자를 하나 입력받은 후에 이 숫자만큼의 크기를 갖는 정사각형을 '#' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 10개의 '#' 문자가 들어있는 라인 10개를 출력하는 것이다.
변수는 다음과 같이 사용하라.

```
int length;           // 입력받은 정사각형 한 변의 길이  
int i, j;           // 반복문 사용을 위한 변수
```

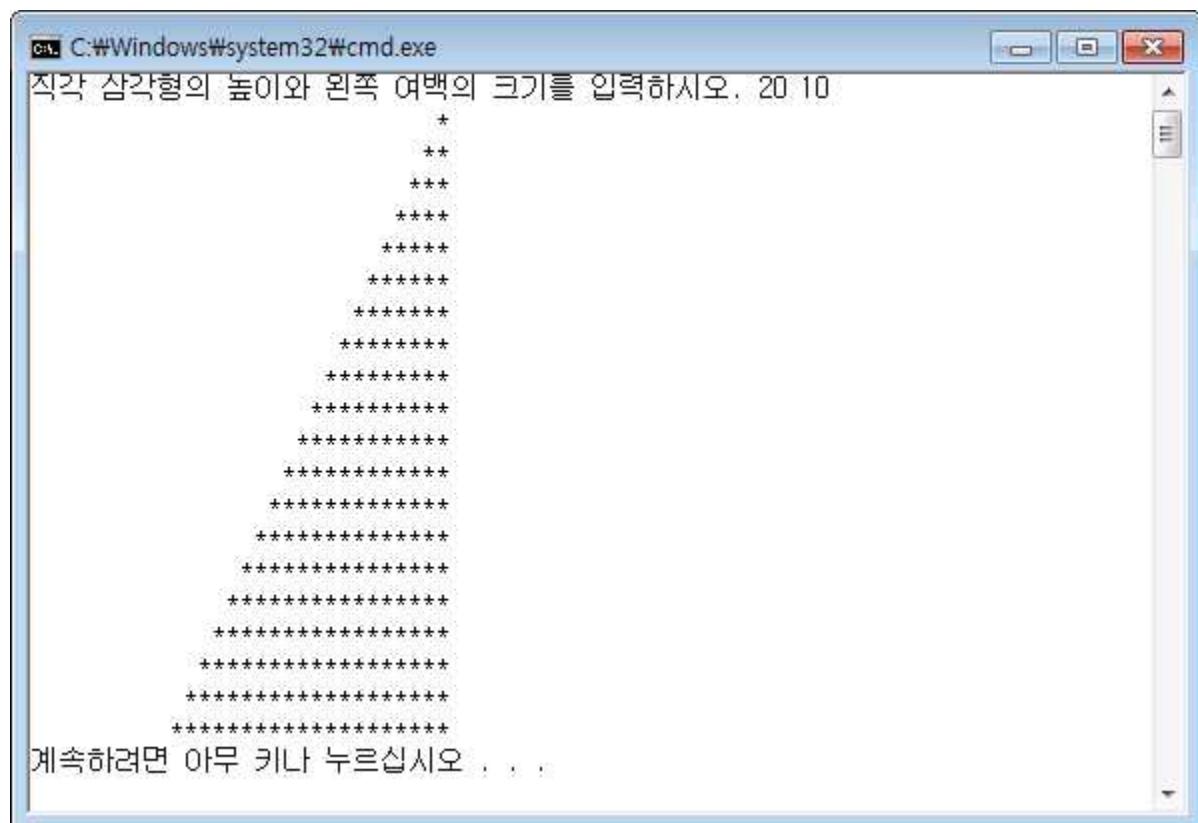


[E02] 입력한 숫자 크기의 높이를 갖는 우직각 삼각형 출력하기

높이와 여백을 정하는 숫자 두 개를 입력받은 후에 이 숫자만큼의 높이와 왼쪽 여백을 갖는 우직각 삼각형을 '*' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 첫 줄에는 1개, 2번째 줄에는 2개, 3번째 줄에는 3개, .. 10번째 줄에는 10개의 '*' 을 왼쪽 여백을 가진 우측 정렬된 모습으로 출력하는 것이다.

변수는 다음과 같이 사용하라.

```
int height;           // 입력받은 높이  
int blank;            // 입력받은 여백 크기  
int i, j;             // 반복문 사용을 위한 변수
```

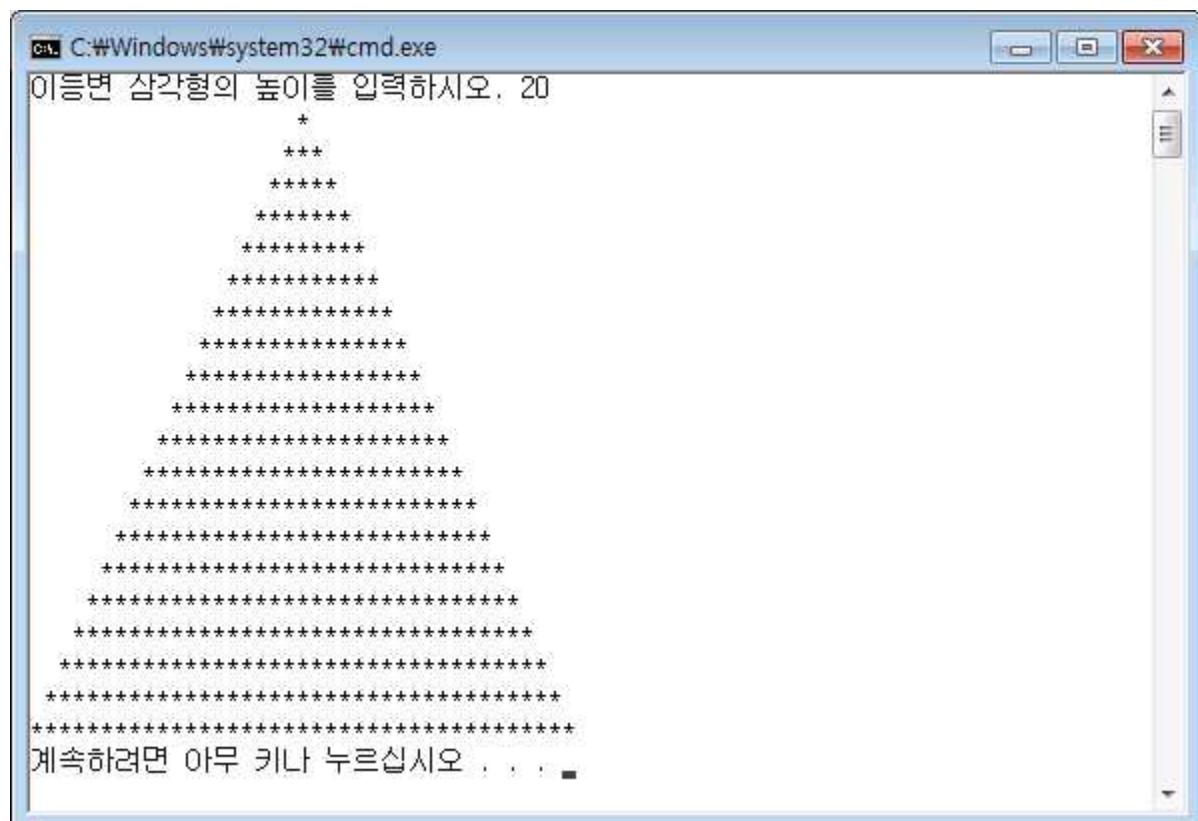


[E03] 입력한 숫자 크기의 높이를 갖는 이등변 삼각형 출력하기

숫자를 하나 입력받은 후에 이 숫자만큼의 높이를 갖는 이등변삼각형을 '*' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 첫 줄에는 1개, 2번째 줄에는 3개, 3번째 줄에는 5개, ..., 10번째 줄에는 19개의 '*'을 가운데 정렬한 모습으로 출력하는 것이다.

변수는 다음과 같이 사용하라.

```
int height;           // 입력받은 높이  
int i, j;            // 반복문 사용을 위한 변수
```



[E04] 홀수단 또는 짝수단의 구구단 출력하기

출력모드(홀수 또는 짝수)를 입력받아 이에 따라 홀수 단 또는 짝수 단의 구구단 만을 1줄에 3개씩 출력하라. 단, 출력모드 입력 내용이 1이면 홀수단, 2이면 짝수단으로 결정하도록 한다.
변수는 다음과 같이 사용하라.

```
int mode;           // 출력모드(1: 홀수단, 2: 짝수단)
int i, j;          // 반복문 사용을 위한 변수
```

C:\Windows\system32\cmd.exe

구구단의 출력모드(1: 홀수단, 2: 짝수단)를 입력하시오. 1

3 x 1 = 3	3 x 2 = 6	3 x 3 = 9
3 x 4 = 12	3 x 5 = 15	3 x 6 = 18
3 x 7 = 21	3 x 8 = 24	3 x 9 = 27
5 x 1 = 5	5 x 2 = 10	5 x 3 = 15
5 x 4 = 20	5 x 5 = 25	5 x 6 = 30
5 x 7 = 35	5 x 8 = 40	5 x 9 = 45
7 x 1 = 7	7 x 2 = 14	7 x 3 = 21
7 x 4 = 28	7 x 5 = 35	7 x 6 = 42
7 x 7 = 49	7 x 8 = 56	7 x 9 = 63
9 x 1 = 9	9 x 2 = 18	9 x 3 = 27
9 x 4 = 36	9 x 5 = 45	9 x 6 = 54
9 x 7 = 63	9 x 8 = 72	9 x 9 = 81

계속하려면 아무 키나 누르십시오 . . .

C:\Windows\system32\cmd.exe

구구단의 출력모드(1: 홀수단, 2: 짝수단)를 입력하시오. 2

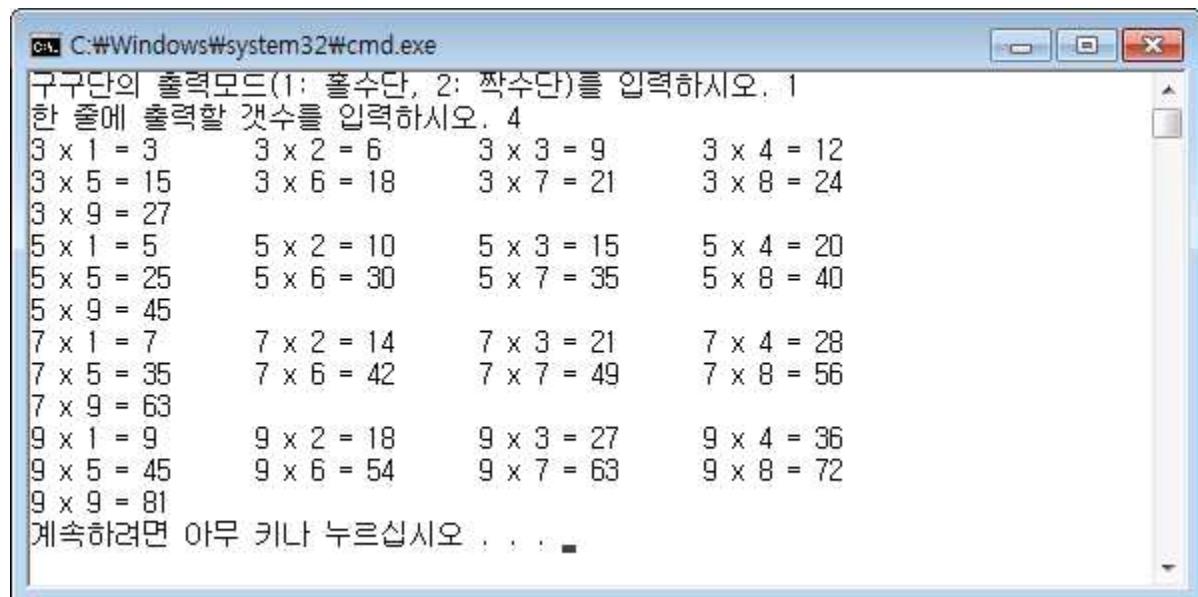
2 x 1 = 2	2 x 2 = 4	2 x 3 = 6
2 x 4 = 8	2 x 5 = 10	2 x 6 = 12
2 x 7 = 14	2 x 8 = 16	2 x 9 = 18
4 x 1 = 4	4 x 2 = 8	4 x 3 = 12
4 x 4 = 16	4 x 5 = 20	4 x 6 = 24
4 x 7 = 28	4 x 8 = 32	4 x 9 = 36
6 x 1 = 6	6 x 2 = 12	6 x 3 = 18
6 x 4 = 24	6 x 5 = 30	6 x 6 = 36
6 x 7 = 42	6 x 8 = 48	6 x 9 = 54
8 x 1 = 8	8 x 2 = 16	8 x 3 = 24
8 x 4 = 32	8 x 5 = 40	8 x 6 = 48
8 x 7 = 56	8 x 8 = 64	8 x 9 = 72

계속하려면 아무 키나 누르십시오 . . .

[E05] 홀수단 또는 짝수단의 구구단을 열의 개수를 맞추어 출력하기

출력모드(홀수 또는 짝수)와 열 갯수를 입력받아 이에 따라 홀수 단 또는 짝수 단의 구구단만을 1줄에 열 개수만큼씩 출력하라. 단, 출력모드 입력 내용이 1이면 홀수단, 2이면 짝수단으로 결정하도록 한다.
변수는 다음과 같이 사용하라.

```
int mode;           // 출력모드(1: 홀수단, 2: 짝수단)
int column;         // 열 개수
int i, j;           // 반복문 사용을 위한 변수
```



[E06] 2차원 숫자 출력하기

행의 크기(rows)와 열의 크기(columns)를 입력받은 후에, 이 크기만큼의 바둑판 모양의 2차원 공간의 각 칸마다 행 번호(1, 2, ..., width)와 열 번호(1, 2, ..., height)를 곱한 값을 출력하라. (아래 그림 참고)
변수는 다음과 같이 사용하라.

```
int rows, columns;           // 행의 개수, 열의 개수
int number;                  // 각 칸에 출력하는 값
int i, j;                    // 반복문 사용을 위한 변수
```

```
C:\Windows\system32\cmd.exe
출력하려는 행의 크기와 열의 크기를 입력하시오. 4 5
 1  2  3  4  5
 2  4  6  8  10
 3  6  9  12 15
 4  8  12 16 20
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
출력하려는 행의 크기와 열의 크기를 입력하시오. 10 15
 1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
 2  4  6  8  10 12 14 16 18 20 22 24 26 28 30
 3  6  9  12 15 18 21 24 27 30 33 36 39 42 45
 4  8  12 16 20 24 28 32 36 40 44 48 52 56 60
 5  10 15 20 25 30 35 40 45 50 55 60 65 70 75
 6  12 18 24 30 36 42 48 54 60 66 72 78 84 90
 7  14 21 28 35 42 49 56 63 70 77 84 91 98 105
 8  16 24 32 40 48 56 64 72 80 88 96 104 112 120
 9  18 27 36 45 54 63 72 81 90 99 108 117 126 135
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150
계속하려면 아무 키나 누르십시오 . . .
```

[Step F] 배열 사용하기

프로그램에서 비슷한 용도로 사용되는 여러 개의 변수가 필요한 경우에 배열이라는 묶음형 변수로 선언하면 아주 편리하다. 예를 들어 5개의 정수형 변수를 사용해서 숫자 5개를 입력받아 그 합계를 계산하려면 다음과 같다.

```
int num1, num2, num3, num4, num5;           // 5개의 숫자
int sum=0;                                     // 합계 (초기값 0)

printf("1번 숫자를 입력하시오. ");
scanf("%d", &num1);
sum = sum + num1;

printf("2번 숫자를 입력하시오. ");
scanf("%d", &num2);
sum = sum + num2;

printf("3번 숫자를 입력하시오. ");
scanf("%d", &num3);
sum = sum + num3;

printf("4번 숫자를 입력하시오. ");
scanf("%d", &num4);
sum = sum + num4;

printf("5번 숫자를 입력하시오. ");
scanf("%d", &num5);
sum = sum + num5;

printf("숫자의 합계는 %d입니다.\n", sum);
```

위의 프로그램을 자세히 들여다보면 다섯 개의 숫자에 대해 비슷한 모양의 구문이 반복되는 것을 볼 수 있다. 여기에서 다섯 개의 숫자를 위한 변수를 배열로 선언해서 프로그램을 변경하면 다음과 같다.

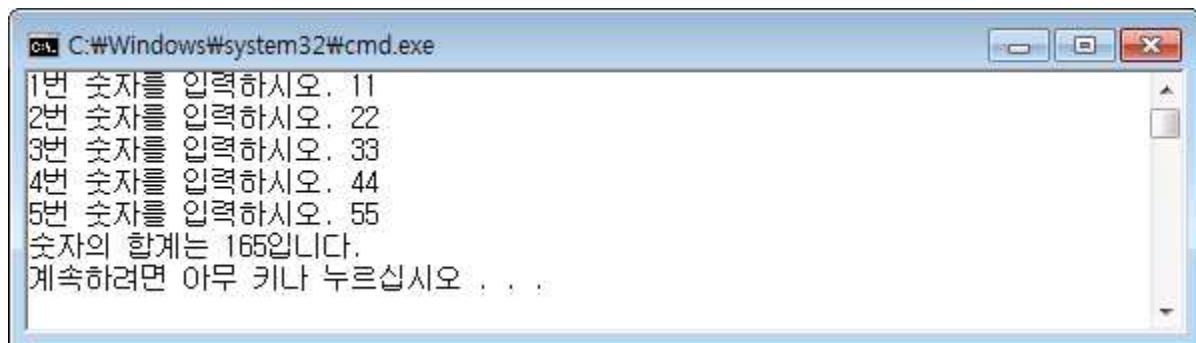
```
int num[5];           // 5개의 숫자
int sum=0;             // 합계 (초기값 0)
int i;                 // 반복문을 위한 변수
for (i=0; i<5; i++)   // i의 값은 0, 1, 2, 3, 4 까지 반복된다.
{
```

```

printf("%d번 숫자를 입력하시오. ", i+1);
scanf("%d", &num[i]);           // 배열변수 num에서 인덱스 i에 해당되는 곳에 입력
sum = sum + num[i];           // 합계 계산
}
printf("숫자의 합계는 %d입니다.\n", sum);

```

배열을 사용할 때에 주의해야 할 점은 선언할 때에 미리 배열 안에 들어갈 요소의 개수를 지정해야 한다는 것이다. 그리고 배열 변수를 사용할 때에는 인덱스 값이 언제나 0부터 시작한다는 점이다. 위의 2종류의 프로그램 구문은 동일하게 다음 화면과 같은 실행 결과를 보인다.



2차원 배열은 여러 개의 데이터를 가로와 세로 또는 행과 열의 크기를 지정해서 다룰 수 있는 묶음 자료형이다. 예를 들어 정수형 숫자 4행 3열의 2차원 배열을 선언하려면 다음과 같다.

```
int number[4][3];
```

위 배열의 구조를 그림으로 표시하면 다음 그림과 같다.

number[0][0]	number[0][1]	number[0][2]
number[1][0]	number[1][1]	number[1][2]
number[2][0]	number[2][1]	number[2][2]
number[3][0]	number[3][1]	number[3][2]

그렇다면 반복문을 사용해서 위 배열에 모두 100이라는 값을 넣으려면 어떻게 하면 될까? 반복문을 사용하기 전에 12개의 칸에 순서대로 일일이 100을 넣는 방법을 생각해보자.

```

number[0][0] = 100; number[0][1] = 100; number[0][2] = 100;
number[1][0] = 100; number[1][1] = 100; number[1][2] = 100;
number[2][0] = 100; number[2][1] = 100; number[2][2] = 100;
number[3][0] = 100; number[3][1] = 100; number[3][2] = 100;

```

위 구문을 자세히 살펴보면 매 줄마다 배열의 두 번째 인덱스만 0에서 2까지 반복되고 있으며, 첫 줄에서 네 번째 줄까지는 첫 번째 인덱스만 0에서 3까지 반복되고 있음을 알 수 있다. 그러므로 첫 번째 인덱스는 i로 처리하고, 두 번째 인덱스를 j로 처리하여 복합 반복문인 이중 **for** 구문으로 만들면 다음과 같다.

```
int i, j;
for (i=0; i<4; i++){
    for (j=0; j<3; j++){
        number[i][j] = 100;
    }
}
```

이렇듯 반복문과 배열을 사용하여 프로그램을 만들 때에는 인덱스를 사용하는 부분을 정확하게 파악하여 프로그램에 적용하는 것이 중요하다.

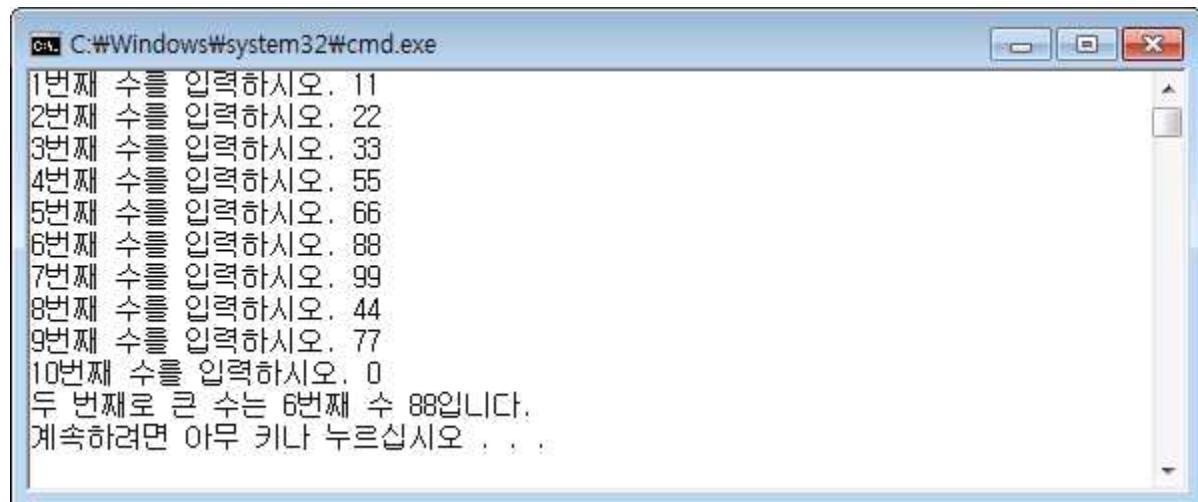
○ 실습 문제

[F01] 두 번째로 큰 수의 순서 찾기

10개의 숫자를 입력받아 배열에 저장한 후에 이 중에서 두 번째로 큰 수가 몇 번째 숫자인지 찾아내어 출력하라.

변수는 다음과 같이 사용하라.

```
int num[10];           // 10개의 숫자  
int first;            // 첫 번째로 큰 수  
int second;           // 두 번째로 큰 수  
int second_max_index; // 두 번째로 큰 수의 인덱스  
int i;                // 반복문을 위한 변수
```

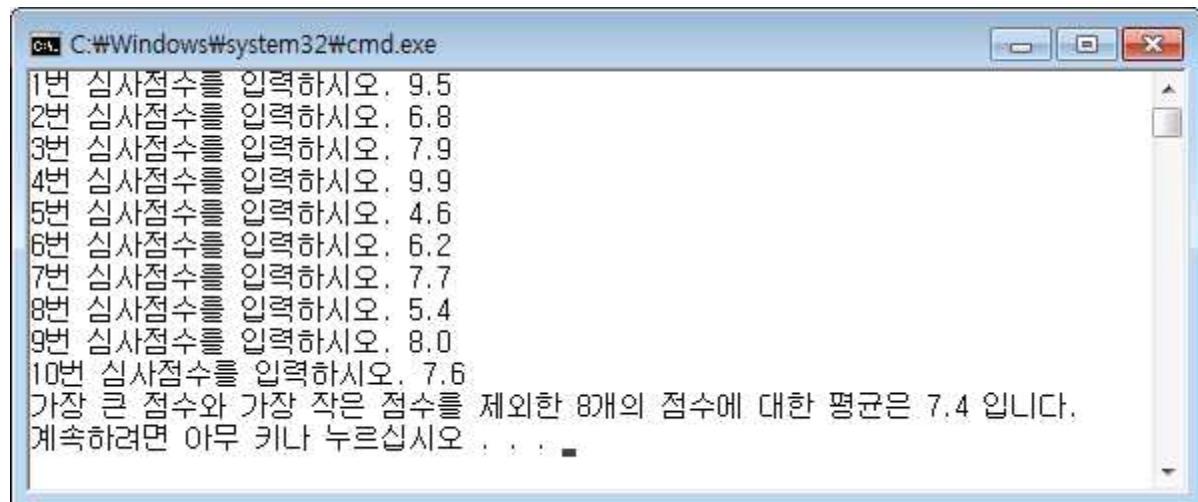


[F02] 심사점수 계산

심사점수를 10개 입력받아 배열에 저장한 후, 이 중에서 가장 큰 점수와 가장 작은 점수를 제외한 8개의 점수에 대한 평균을 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
float score[10];           // 심사점수
float max, min;            // 가장 큰 점수, 가장 작은 점수
float total;                // 점수 총 합계
float average;              // 평균점수
int i;                      // 반복문을 위한 변수
```



[F03] 5명의 국, 영, 수 3과목 점수의 과목별 총점, 평균값 구하기

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 각 과목별 총점과 평균 점수를 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
int jumsu[5][3];           // 5명의 3과목 점수를 저장하고 있는 2차원 배열  
int sum[3];                // 3과목 총점  
float average[3];          // 3과목 평균  
int i, j;                  // 반복문을 위한 변수
```

The screenshot shows a Windows Command Prompt window titled 'cmd' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following output:

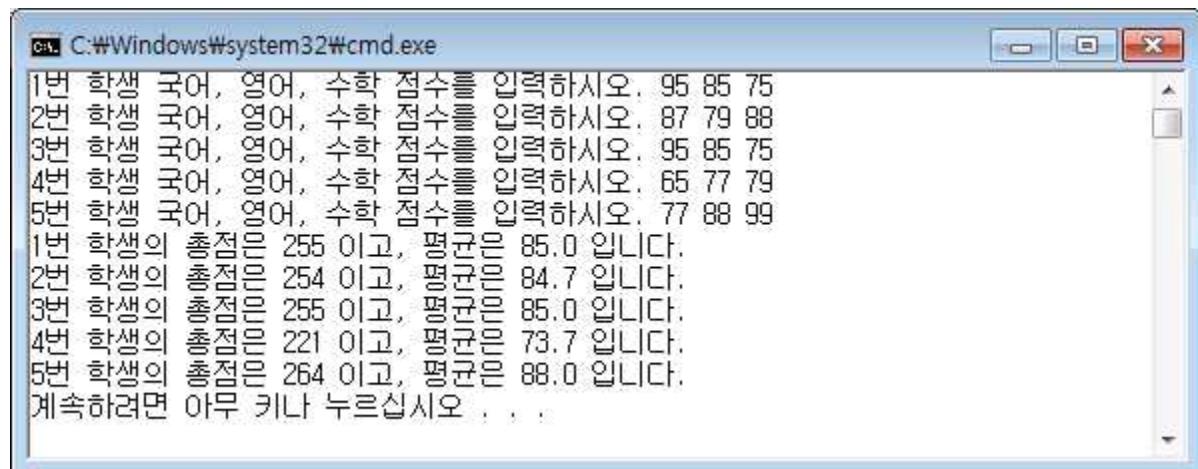
```
1번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75  
2번 학생 국어, 영어, 수학 점수를 입력하시오. 87 79 88  
3번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75  
4번 학생 국어, 영어, 수학 점수를 입력하시오. 65 77 79  
5번 학생 국어, 영어, 수학 점수를 입력하시오. 77 88 99  
국어의 총점은 419이고, 평균은 83.8입니다.  
영어의 총점은 414이고, 평균은 82.8입니다.  
수학의 총점은 416이고, 평균은 83.2입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

[F04] 5명의 국, 영, 수 3과목 점수의 학생별 총점, 평균값 구하기

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 각 과목별 총점과 평균 점수를 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
int jumsu[5][3];           // 5명의 3과목 점수를 저장하고 있는 2차원 배열  
int sum[5];                // 학생별 총점  
float average[5];          // 학생별 평균  
int i, j;                  // 반복문을 위한 변수
```



[F05] 비만 판정

10명의 신장(cm단위)과 체중(kg단위)를 입력받은 후, 이들 중 몇 번째 사람들이 비만인지를 판정하여 출력하라. 그리고 도합 몇 명이 비만인지 출력하라.

단, 비만여부는 다음 비만도 수치가 25이상인 경우에 "비만"으로 판단한다.

비만도 수치 = 체중(kg) / (신장(m)의 제곱) 으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.

변수는 다음과 같이 사용하라.

```
int height[10], weight[10];           // 10명의 신장(cm), 체중(kg)
float bmi[10];                      // 10명의 비만도 수치
int count;                          // 비만인 사람의 숫자
int i;                            // 반복문을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32'. The window displays the following text:

```
1번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 166 56
2번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 176 90
3번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 187 60
4번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 155 47
5번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 170 75
6번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 173 80
7번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 165 50
8번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 182 73
9번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 158 48
10번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 170 66
2번째 사람은 비만입니다.
5번째 사람은 비만입니다.
6번째 사람은 비만입니다.

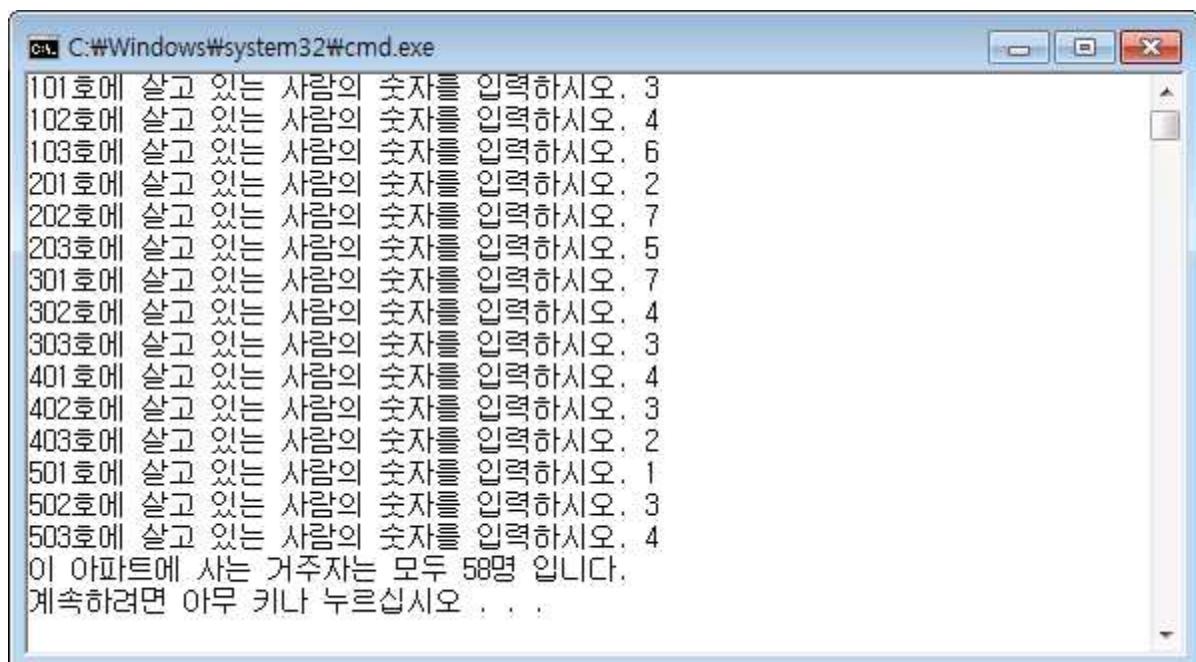
총 3명의 사람이 비만입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[F06] 5층 아파트의 거주자 숫자 구하기

한 층에 3집(1호, 2호, 3호)으로 되어 있는 5층짜리 아파트가 있다. 2차원 배열을 사용하여 101호부터 503호까지 각 집에 살고 있는 사람의 숫자를 입력받아 보관하라. 그리고 이 아파트에 사는 거주자는 모두 몇 명인지 출력하라.

변수는 다음과 같이 사용하라.

```
int number[5][3];           // 각 집의 거주자 수
int total;                  // 아파트의 총 거주자 수
int ho;                     // 아파트 호를 나타내는 변수
int i, j;                   // 반복문 사용을 위한 변수
```



[F07] 5층 아파트의 층별, 호수별 거주자 숫자 구하기

한 층에 3집(1호, 2호, 3호)으로 되어 있는 5층짜리 아파트가 있다. 2차원 배열을 사용하여 101호부터 503호까지 각 집에 살고 있는 사람의 숫자를 입력받아 보관하라. 그리고 이 아파트에 사는 거주자의 숫자를 층별(1층~5층)로 합산하여 출력하고, 호수별(1호~3호)로 합산하여 출력하라. 예를 들어 1층 거주자의 수는 101호, 102호, 103호 거주자의 수를 합한 것이고, 2호 라인 거주자의 수는 102호, 202호, 302호, 402호, 502호 거주자의 수를 합한 것이다.

변수는 다음과 같이 사용하라.

```
int number[5][3];           // 각 집의 거주자 수
int floor_total[5];         // 층별 거주자 합계 (1층, 2층, 3층, 4층, 5층)
int line_total[3];          // 호수별 거주자 합계 (1호라인, 2호라인, 3호라인)
int total;                  // 아파트의 총 거주자 수
int ho;                     // 아파트 호를 나타내는 변수
int i, j;                   // 반복문 사용을 위한 변수
```

[F08] 겹치지 않는 숫자 10개 입력 받기

사용자에게 1부터 100사이의 숫자를 10개 입력받아 이를 순서대로 출력하라. 단, 사용자가 입력하는 동안 이미 입력한 숫자와 같은 숫자를 입력하면 "잘못 입력하였습니다. 다시 입력하세요."라는 문구와 함께 다시 입력받도록 하라. 입력이 완료되면 10개의 수를 모두 출력하라.

변수는 다음과 같이 사용하라.

```
int number[10];           // 사용자가 입력한 숫자 10개
int count;                // 현재까지 입력된 숫자의 갯수(0~10)
int i;                     // 반복문을 위한 변수
```

```
cmd C:\Windows\system32\cmd.exe
1부터 100사이의 숫자를 입력하시오.
1번째 숫자를 입력하시오. 1
2번째 숫자를 입력하시오. 1
잘못 입력하였습니다. 다시 입력하세요.
2번째 숫자를 입력하시오. 2
3번째 숫자를 입력하시오. 4
4번째 숫자를 입력하시오. 6
5번째 숫자를 입력하시오. 4
잘못 입력하였습니다. 다시 입력하세요.
5번째 숫자를 입력하시오. 2
잘못 입력하였습니다. 다시 입력하세요.
5번째 숫자를 입력하시오. 7
6번째 숫자를 입력하시오. 10
7번째 숫자를 입력하시오. 35
8번째 숫자를 입력하시오. 10
잘못 입력하였습니다. 다시 입력하세요.
8번째 숫자를 입력하시오. 23
9번째 숫자를 입력하시오. 43
10번째 숫자를 입력하시오. 35
잘못 입력하였습니다. 다시 입력하세요.
10번째 숫자를 입력하시오. 33
1번째 숫자는 1입니다
2번째 숫자는 2입니다
3번째 숫자는 4입니다
4번째 숫자는 6입니다
5번째 숫자는 7입니다
6번째 숫자는 10입니다
7번째 숫자는 35입니다
8번째 숫자는 23입니다
9번째 숫자는 43입니다
10번째 숫자는 33입니다
계속하려면 아무 키나 누르십시오 . . .
```

[F09] 배열을 이용한 연중 날짜 계산

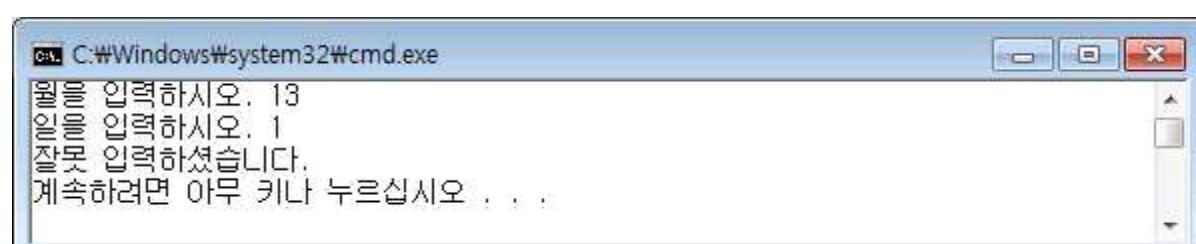
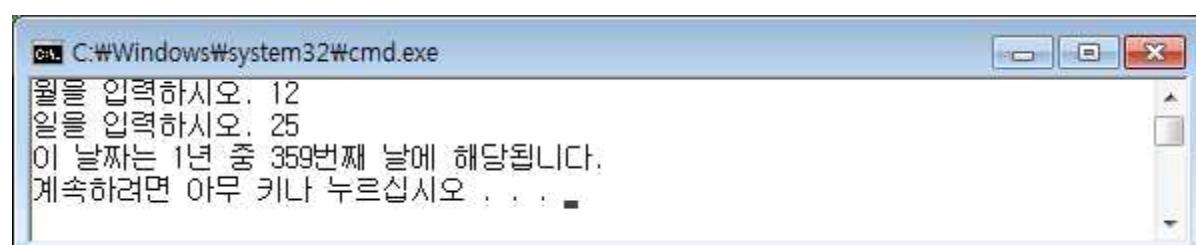
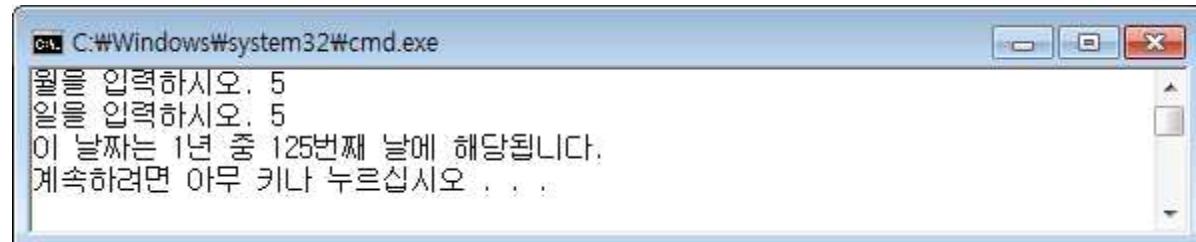
날짜를 월과 일로 입력받아 이 날짜는 1년 중 몇 번째 날에 해당되는지 계산하여 출력하라.

단, 매 월의 날 수를 다음과 같이 배열로 만들어 이를 이용하여 계산하라.

```
int monthdays[12] = {31,28,31,30,31,30,31,31,30,31,30,31}; // 1~12월의 날 수
```

변수는 다음과 같이 사용하라.

```
int monthdays[12]; // 1~12월의 날 수  
int month, day; // 입력받은 월, 일  
int day_count; // 1년 중 날 수  
int i; // 반복문을 위한 변수
```



[Step G] 조건과 반복을 활용하는 응용 예제 해결하기

이번 단계에서는 그동안 연습해 온 조건문과 반복문을 함께 응용하여 해결해야 하는 문제들을 풀어보려고 한다. 예를 들어서 어떤 레스토랑에서 하루 동안의 식사 금액을 계산하는 프로그램을 만들여보자. 이 식당에는 4가지 메뉴가 있으며 각 메뉴별 가격은 다음과 같다.

- 1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000원)

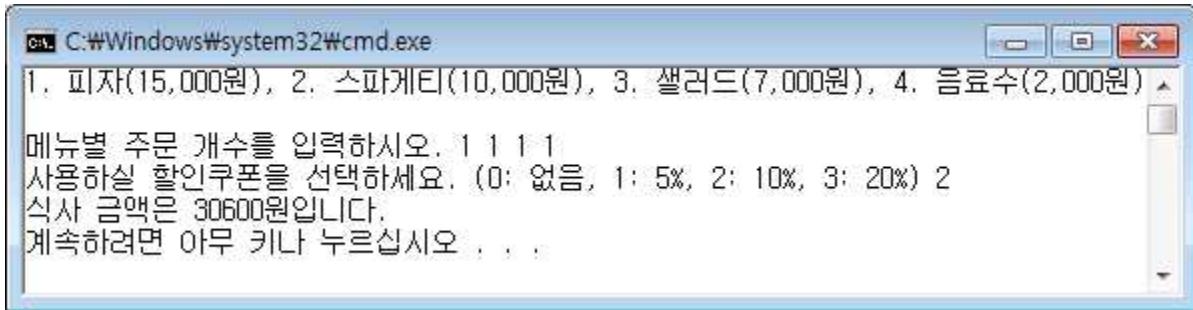
어떤 한 팀의 주문서 내용 즉, 메뉴별 주문 개수를 입력받아 식대를 계산하는 구문은 다음과 같다.

```
int order[4];           // 주문개수 (피자, 스파게티, 샐러드, 음료수)
int price[4]={15000, 10000, 7000, 2000};      // 메뉴별 가격
int sum;                // 식사금액
int i;
printf("1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000원)\n");
printf("메뉴별 주문 개수를 입력하시오. ");
scanf("%d %d %d %d", &order[0], &order[1], &order[2], &order[3]);
sum = 0;
for (i=0; i<4; i++){
    sum = sum + (order[i] * price[i]);
}
printf("식사 금액은 %d원입니다.\n", sum);
```

그런데, 이 레스토랑에서는 할인쿠폰제도가 있어서 5%, 10%, 20% 세 종류의 할인쿠폰 중 하나를 제시하는 팀에게는 할인해 주는 제도를 시행하고 있다면, 프로그램의 하단 부분은 다음과 같이 변경 되어야 할 것이다.

```
int coupon;
... 중략
sum = 0;
for (i=0; i<4; i++){
    sum = sum + order[i];
}
printf("사용하실 할인쿠폰을 선택하세요. (0: 없음, 1: 5%, 2: 10%, 3: 20%) ");
scanf("%d", &coupon);
if (coupon == 1) sum = sum * 0.95;
else if (coupon == 2) sum = sum * 0.90;
else if (coupon == 3) sum = sum * 0.80;
printf("식사 금액은 %d원입니다.\n", sum);
```

현재까지 만든 프로그램의 실행 화면은 다음과 같다.



이제는 하루 동안의 매출을 계산해보자. 하루 동안 식사하러 온 팀이 모두 몇 팀인지를 입력받아 이 숫자만큼 위 프로그램을 반복해가면서 총 매출액을 계산하는 것이다. 그렇기 위해서는 다음과 같이 앞의 프로그램을 전체적으로 반복문으로 둘러싸야 한다. 그리고 매 반복 때마다 계산한 `sum`의 값을 누적해서 합산해야 한다.

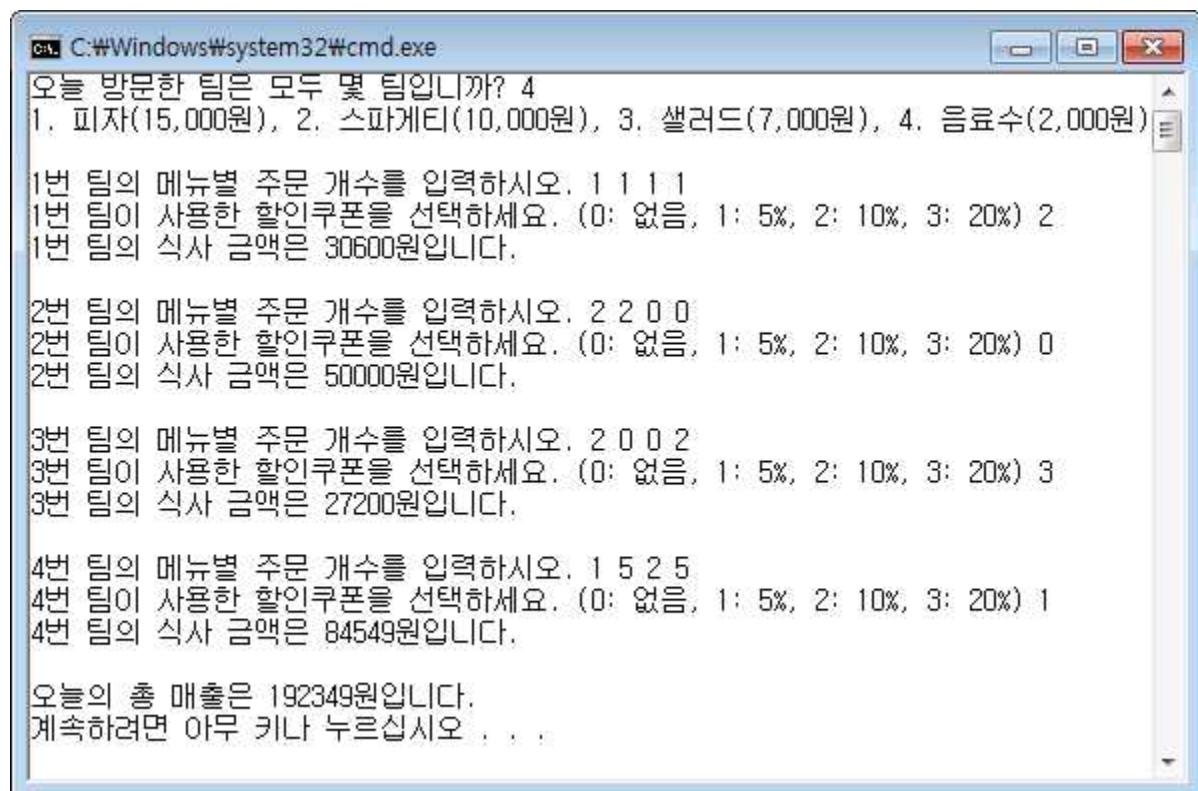
```
/*
ex_G.c
*/
#include <stdio.h>
void main()
{
    int order[4]; // 주문개수 (피자, 스파게티, 샐러드, 음료수)
    int price[4]={15000, 10000, 7000, 2000}; // 메뉴별 가격
    int sum; // 식사금액
    int total_sum=0; // 총 매출액
    int team; // 방문한 팀의 개수
    int coupon; // 사용한 쿠폰 종류
    int i, k;

    printf("오늘 방문한 팀은 모두 몇 팀입니까? ");
    scanf("%d", &team);
    printf("1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000
    원)\n");
    for (k=0; k<team ; k++ )
    {
        printf("%d번 팀의 메뉴별 주문 개수를 입력하시오. ", k+1);
        scanf("%d %d %d %d", &order[0], &order[1], &order[2], &order[3]);
        sum = 0;
        for (i=0; i<4; i++){
            sum = sum + (order[i] * price[i]); // 팀의 식사금액 계산
    }
}
```

```

    }
    printf("%d번 팀이 사용한 할인쿠폰을 선택하세요. (0: 없음, 1: 5%, 2: 10%, 3:
20%) ", k+1);
    scanf("%d", &coupon);
    if (coupon == 1) sum = sum * 0.95;           // 5% 할인쿠폰 적용
    else if (coupon == 2) sum = sum * 0.90;       // 10% 할인쿠폰 적용
    else if (coupon == 3) sum = sum * 0.80;       // 20% 할인쿠폰 적용
    printf("%d번 팀의 식사 금액은 %d원입니다.\n\n", k+1, sum);
    total_sum = total_sum + sum;                 // 총매출액 누적 합산
}
printf("오늘의 총 매출은 %d원입니다.\n", total_sum);
}

```



○ 실습 문제

[G01] 나이 계산 및 연령대 판정

최대 100명까지 사람들의 태어난 년도를 입력받아 나이를 계산한 후, 그 값을 저장하되, 2012보다 큰 년도가 입력되기 전까지 태어난 년도를 반복하여 입력받도록 하라. 반복이 끝나면 지금까지 입력된 사람들의 나이를 모두 출력하고, 각자의 나이에 따라 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 이 중에 유아, 어린이, 청소년, 청년, 중년, 노년이 각각 몇 명인지 출력하라.

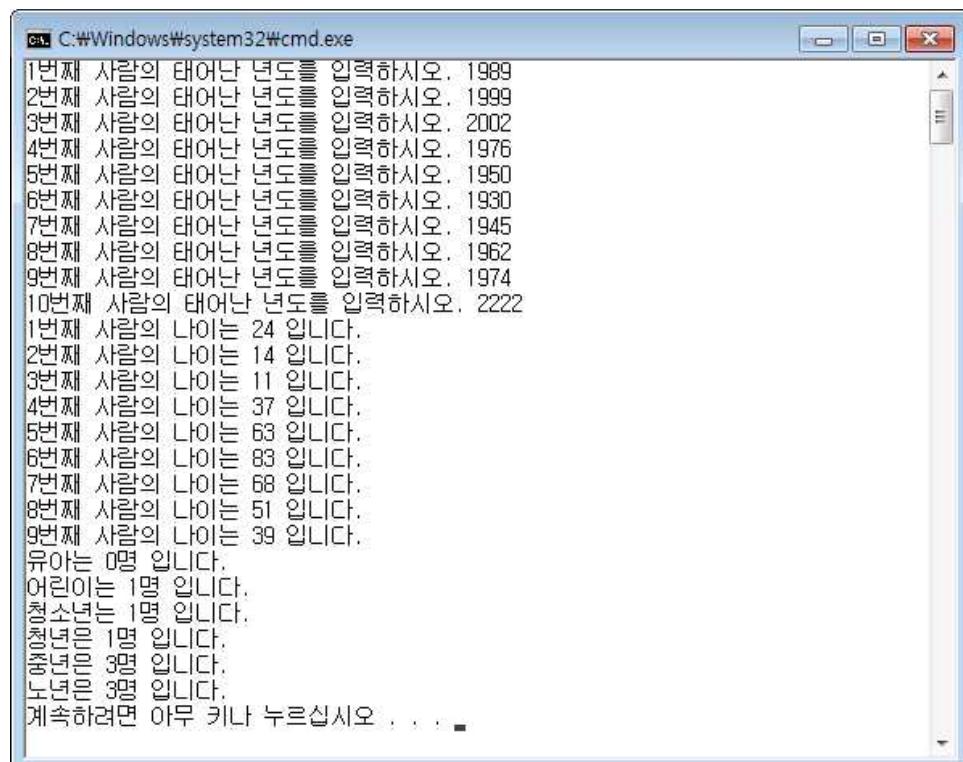
단, 나이 = $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아, 7세 이상 ~ 13세미만 : 어린이, 13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년, 30세 이상 ~ 60세 미만 : 중년, 60세 이상 : 노년

변수는 다음과 같이 사용하라.

```
int birth_year;      // 입력받은 태어난 년도
int age[100];        // 각 사람들의 나이 (최대 100명)
int count_person;    // 입력된 인원 수
int count_baby;      // 유아 수
int count_child;     // 어린이 수
int count_youth;     // 청소년 수
int count_young;     // 청년 수
int count_adult;     // 중년 수
int count_old;       // 노년 수
int i;                // 반복문을 위한 변수
```



[G02] 물의 온도 구간 판정

물의 온도를 10회 입력받은 후, 이 물이 각각 어느 정도의 온수인지 판정하여 그 결과를 출력하라. 출력할 내용은 입력된 10개의 온도 값, 냉수 입력 횟수, 미온수 입력 횟수, 온수 입력 횟수, 끓는 물 입력 횟수를 각각 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

음수 값 (0미만) : 잘못입력

0도 ~ 25도 미만 : 냉수

25도 ~ 40도 미만 : 미온수

40도 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

변수는 다음과 같이 사용하라.

```
float input_degree[10];      // 입력받은 온도
int count1=0;                // 냉수 입력 횟수
int count2=0;                // 미온수 입력 횟수
int count3=0;                // 온수 입력 횟수
int count4=0;                // 끓는 물 입력 횟수
int i;                       // 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
1번 물의 온도를 입력하시오. 10.5
2번 물의 온도를 입력하시오. 25.1
3번 물의 온도를 입력하시오. 88.4
4번 물의 온도를 입력하시오. 45.0
5번 물의 온도를 입력하시오. 30.5
6번 물의 온도를 입력하시오. 15.7
7번 물의 온도를 입력하시오. 46.7
8번 물의 온도를 입력하시오. 85
9번 물의 온도를 입력하시오. 69.0
10번 물의 온도를 입력하시오. -0.5
1번 물의 온도는 10.5도 입니다.
2번 물의 온도는 25.1도 입니다.
3번 물의 온도는 88.4도 입니다.
4번 물의 온도는 45.0도 입니다.
5번 물의 온도는 30.5도 입니다.
6번 물의 온도는 15.7도 입니다.
7번 물의 온도는 46.7도 입니다.
8번 물의 온도는 85.0도 입니다.
9번 물의 온도는 69.0도 입니다.
10번 물의 온도는 -0.5도 입니다.
냉수 입력 횟수 2 입니다.
미온수 입력 횟수는 2 입니다.
온수 입력 횟수는 3 입니다.
끓는 물 입력 횟수는 2 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[G03] 점수 계산

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 다음 항목들을 계산하여 출력하라.

- 1) 각 과목별 총점과 평균 점수
- 2) 각 학생별 총점과 평균점수, 평균에 따른 등급

등급은 다음과 같은 기준으로 결정하라.

평균 90이상 : A

평균 80이상 ~ 90미만 : B

평균 70이상 ~ 80미만 : C

평균 60이상 ~ 70미만 : D

평균 60미만 : F

변수는 다음과 같이 사용하라.

```
int jumsu[5][3];           // 5명의 3과목 점수를 저장하고 있는 2차원 배열
int sum_student[5];         // 학생별 총점
float average_student[5];   // 학생별 평균
char grade[5];             // 학생별 등급
int sum_class[3];           // 과목별 총점
float average_class[3];     // 과목별 평균
int i, j;                  // 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
1번 학생 국어, 영어, 수학 점수를 입력하시오. 85 95 75
2번 학생 국어, 영어, 수학 점수를 입력하시오. 90 80 70
3번 학생 국어, 영어, 수학 점수를 입력하시오. 65 85 75
4번 학생 국어, 영어, 수학 점수를 입력하시오. 60 70 80
5번 학생 국어, 영어, 수학 점수를 입력하시오. 60 50 60
1) 각 과목별 총점과 평균 점수
국어 점수의 총점은 3600이고, 평균 점수는 72.0 입니다.
영어 점수의 총점은 3800이고, 평균 점수는 76.0 입니다.
수학 점수의 총점은 3600이고, 평균 점수는 72.0 입니다.
2) 각 학생별 총점과 평균점수, 평균에 따른 등급
1번 학생의 총점은 255 이고, 평균은 85.0 (등급 B)입니다.
2번 학생의 총점은 240 이고, 평균은 80.0 (등급 B)입니다.
3번 학생의 총점은 225 이고, 평균은 75.0 (등급 C)입니다.
4번 학생의 총점은 210 이고, 평균은 70.0 (등급 C)입니다.
5번 학생의 총점은 170 이고, 평균은 56.7 (등급 F)입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[G04] 부동산 중개 수수료 계산기

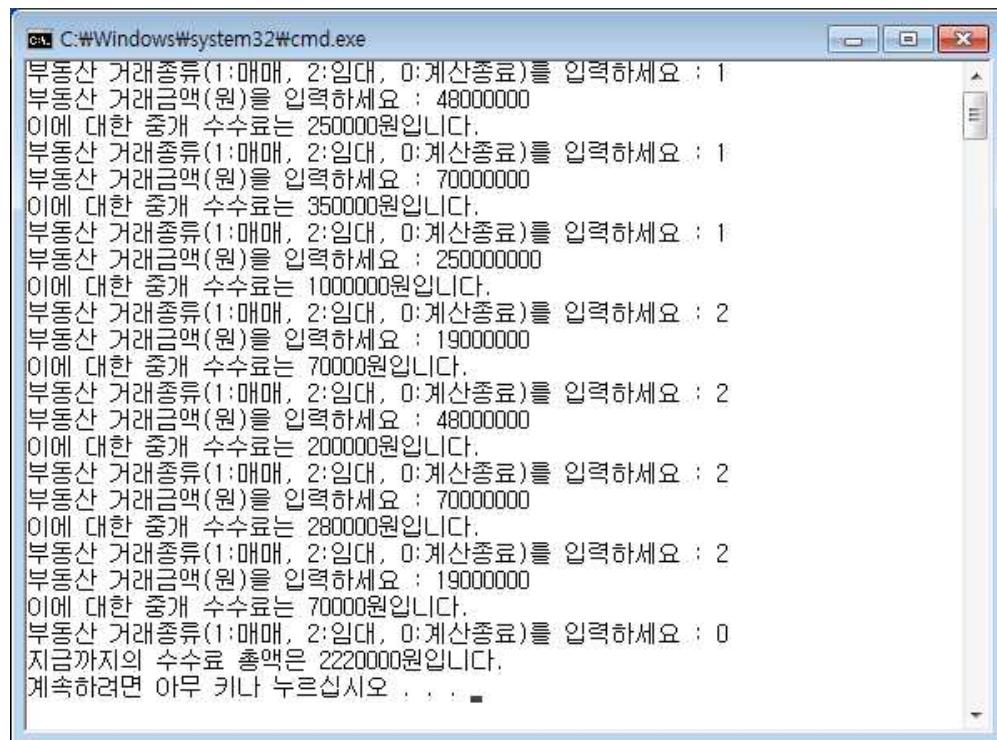
부동산 거래종류(1:매매, 2:임대, 0:계산종료)와 거래금액을 입력받은 후에 이에 대한 중개 수수료를 계산하여 출력하라. 입력과 출력을 계속 반복하되 계산종료(0)를 입력하면 반복을 중단하고 지금까지의 수수료 총액을 출력하라.

중개 수수료 계산 방법은 아래 표를 참고하라.

거래종류	거래금액	수수료비율	상한금액
매매	5천만원 미만	0.6%	250,000
	5천만원 이상 ~ 2억원 미만	0.5%	800,000
	2억원 이상	0.4%	없음
임대	2천만원 미만	0.5%	70,000
	2천만원 이상 ~ 5천만원 미만	0.5%	200,000
	5천만원 이상 ~ 1억원 미만	0.4%	300,000
	1억원 이상	0.3%	없음

변수는 다음과 같이 사용하라.

```
int kind;           // 거래종류(1:매매, 2:임대, 0:계산종료)
int money;          // 거래금액
int charge;         // 중개 수수료
int total_charge;   // 수수료 총액
```



The screenshot shows a Windows command prompt window titled 'cmd.exe' running on a Windows system. The window displays a series of interactions between the user and a program. The user inputs transaction types (1 or 2) and amounts (e.g., 48000000, 70000000), and the program calculates and prints the commission rates (e.g., 250000, 350000, 700000) and the total commission so far (e.g., 2220000). The program ends with a message about exiting.

```
C:\Windows\system32\cmd.exe
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)을 입력하세요 : 48000000
이에 대한 중개 수수료는 250000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)을 입력하세요 : 70000000
이에 대한 중개 수수료는 350000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)을 입력하세요 : 250000000
이에 대한 중개 수수료는 1000000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)을 입력하세요 : 190000000
이에 대한 중개 수수료는 700000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)을 입력하세요 : 48000000
이에 대한 중개 수수료는 200000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)을 입력하세요 : 70000000
이에 대한 중개 수수료는 280000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)을 입력하세요 : 19000000
이에 대한 중개 수수료는 70000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 0
지금까지의 수수료 총액은 2220000원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[G05] PC방 이용료 계산기

PC방 이용료를 계산하는 프로그램을 작성하라. 사용한 시간(시간, 분)을 입력받은 후 이에 따른 이용료를 화면에 출력하는 것을 반복한다. 시간과 분이 모두 0으로 입력되면 계산을 마치고 지금까지의 이용료 총금액을 출력하라.

단, 이용료는 매 30분 당 1,000원씩으로 계산하며, 다음과 같이 시간에 따라 할인혜택을 적용한다.

- 1) 2시간 이상 3시간 미만 이용자는 비용의 5%를 할인받는다.
- 2) 3시간 이상 5시간 미만 이용자는 비용의 10%를 할인받는다.
- 3) 5시간 이상 이용자는 비용의 20%를 할인받는다.

예) 이용시간이 4시간 20분이면 정상금액 9,000원에서 10% 할인을 받아 이용료는 8,100원이 된다.

변수는 다음과 같이 사용한다.

```
int hour, minute;           // 이용한 시간, 분
int charge;                 // 이용료
int total_charge;           // 이용료 총액
```

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays a series of user inputs and corresponding output responses, illustrating the logic of the program. The user inputs are times in hours and minutes, and the program outputs the calculated charges and a final total at the end.

```

사용한 시간을 시간과 분으로 입력하세요 : 1 20
고객님의 이용료는 3000원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 2 40
고객님의 이용료는 5700원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 4 20
고객님의 이용료는 8100원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 6 0
고객님의 이용료는 9600원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 3 30
고객님의 이용료는 6300원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 1 50
고객님의 이용료는 4000원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 0 50
고객님의 이용료는 2000원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 3 40
고객님의 이용료는 7200원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 2 10
고객님의 이용료는 4750원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 2 59
고객님의 이용료는 5700원입니다.
사용한 시간을 시간과 분으로 입력하세요 : 0 0

지금까지의 이용료 총금액은 56350원입니다.
계속하려면 아무 키나 누르십시오 . . .

```

[G06] 쇼핑몰 매출 계산기

어떤 가게에서 세 종류의 제품(가죽장갑 1만원, 텔장갑 6천원, 비닐장갑 3천원)을 판매하고 있다. 손님들이 들어오면 이 제품들에 대해 각각 몇 개를 구입할 것인지를 입력받아서 판매금액을 계산하여 출력하라. 구입 개수를 모두 0으로 입력하게 되면 판매가 종료되도록 하며, 지금까지 판매한 제품의 종류별 개수와 총 매출 금액을 화면에 출력하라.

변수는 다음과 같이 사용한다.

```
int order[3];           // 입력받는 구매 제품 개수 (가죽장갑, 텔장갑, 비닐장갑)
int sale;               // 계산한 판매금액
int total_sale;         // 총 매출액 총액
int total_order[3];     // 총 판매 개수 (가죽장갑, 텔장갑, 비닐장갑)
int i;                  // 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
세 종류의 제품(1.가죽장갑 1만원, 2.텔장갑 6천원, 3.비닐장갑 3천원)이 있습니다.

1번 제품은 몇 개를 구입하실래요? 1
2번 제품은 몇 개를 구입하실래요? 1
3번 제품은 몇 개를 구입하실래요? 1
판매 금액은 19000원입니다.

1번 제품은 몇 개를 구입하실래요? 1
2번 제품은 몇 개를 구입하실래요? 2
3번 제품은 몇 개를 구입하실래요? 3
판매 금액은 31000원입니다.

1번 제품은 몇 개를 구입하실래요? 3
2번 제품은 몇 개를 구입하실래요? 0
3번 제품은 몇 개를 구입하실래요? 0
판매 금액은 30000원입니다.

1번 제품은 몇 개를 구입하실래요? 0
2번 제품은 몇 개를 구입하실래요? 20
3번 제품은 몇 개를 구입하실래요? 1
판매 금액은 123000원입니다.

1번 제품은 몇 개를 구입하실래요? 5
2번 제품은 몇 개를 구입하실래요? 5
3번 제품은 몇 개를 구입하실래요? 5
판매 금액은 95000원입니다.

1번 제품은 몇 개를 구입하실래요? 0
2번 제품은 몇 개를 구입하실래요? 0
3번 제품은 몇 개를 구입하실래요? 0

지금까지 판매한 가죽장갑은 10개입니다.
지금까지 판매한 텔장갑은 28개입니다.
지금까지 판매한 비닐장갑은 10개입니다.
지금까지의 총 매출금액은 298000원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[G07] 놀이공원 매표소

놀이공원 매표소 프로그램을 제작하라. 프로그램 시작하면 몇 팀이 방문하였는지 입력받아 팀의 수만큼 다음과 같은 순서로 진행하도록 한다.

1) 팀별 인원 구성을 입력받는다. 인원구성은 (초등학생, 청소년, 일반인, 경로대상) 4종류별 인원수를 숫자로 입력받는다.

2) 팀별 정상요금을 계산한다. 1인당 요금은 다음과 같다.

- 초등학생 : 5000원, - 청소년 : 10000원, - 일반인 : 15000원, - 경로대상 : 3000원

3) 팀 별로 할인카드 소지 여부를 확인한다. 할인카드 종류별 할인율은 다음과 같다.

- 카드없음 : 할인 없음, - 일반등급 카드 : 10% 할인, - VIP 등급 카드 : 20% 할인

4) 최종 계산된 팀별 입장료를 출력한다.

모든 팀에 대한 처리가 완료되면 다음 내용을 화면에 출력하라.

1) 총 방문자 수

2) 인원구성별 방문자 합계 (4종류)

3) 총 입장료

변수는 다음과 같이 사용하라.

```
int team_count; // 방문한 팀수
int charge[4] = {5000, 10000, 15000, 3000}; // 연령별 수
int count[4]; // 입력받은 연령별 인원 수
int v_count[4]; // 연령별 방문자 합계
int total_count = 0; // 총 방문자 수
int sum; // 팀별 계산한 요금
int total_sum = 0; // 총 요금
int membership = 0; // 할인카드 종류(카드없음:0, 일반등급 카드 : 1, VIP 등급 카드 : 2)
int i, j; // 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
오늘 방문한 팀 수를 입력하세요. 4
1번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 2 1 2 0
1번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 2
1번팀 입장료는 40000원입니다.
2번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 5 0 1 0
2번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 0
2번팀 입장료는 40000원입니다.
3번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 4 0 2 2
3번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 1
3번팀 입장료는 50400원입니다.
4번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 0 5 1 0
4번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 0
4번팀 입장료는 65000원입니다.

오늘 총 방문자 수는 25명입니다.
초등학생 수는 11명입니다.
청소년 수는 6명입니다.
일반인 수는 6명입니다.
경로대상 수는 2명입니다.

총 입장료는 195400원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[Step H] C언어의 주요 함수 사용하기

C 프로그램에서는 기본적으로 `main()` 함수에서 프로그램이 시작되고 끝난다. 하지만 프로그램에서 처리하는 모든 작업을 `main()` 함수에서 처리하는 것은 번거롭기도 하고, 효율적이지 못한 방법이다. 필요에 따라 역할을 분담하는 별도의 서브 프로시저를 만들어 사용하는 것이 필요한데, 이를 함수(function)이라고 부른다. C 프로그램에서는 종류별로 언어 자체적으로 제공하는 라이브러리 함수들이 있는데 이번 단계에서는 자주 사용되는 C 라이브러리 함수들을 연습하려고 한다. 그리고 영역 3에서는 프로그램을 제작할 때에 필요한 함수들을 직접 만드는 방법을 연습하게 될 것이다.

먼저 주로 사용되는 수학 관련 함수들을 살펴보도록 하자.

함수 원형	설명
<code>double fabs (double x);</code>	x 의 절대값을 리턴한다.
<code>double ceil (double x);</code>	x 보다 크거나 같은 정수 값을 리턴한다.
<code>double floor (double x);</code>	x 보다 작거나 같은 정수 값을 리턴한다.
<code>double exp (double x);</code>	e^x 의 값을 리턴한다.
<code>double pow (double x, double y);</code>	x^y 의 값을 리턴한다.
<code>double log (double x);</code>	$\log_e x$ 의 값을 리턴한다.
<code>double log10 (double x);</code>	$\log_{10} x$ 의 값을 리턴한다.
<code>double sqrt (double x);</code>	x 의 제곱근을 리턴한다.

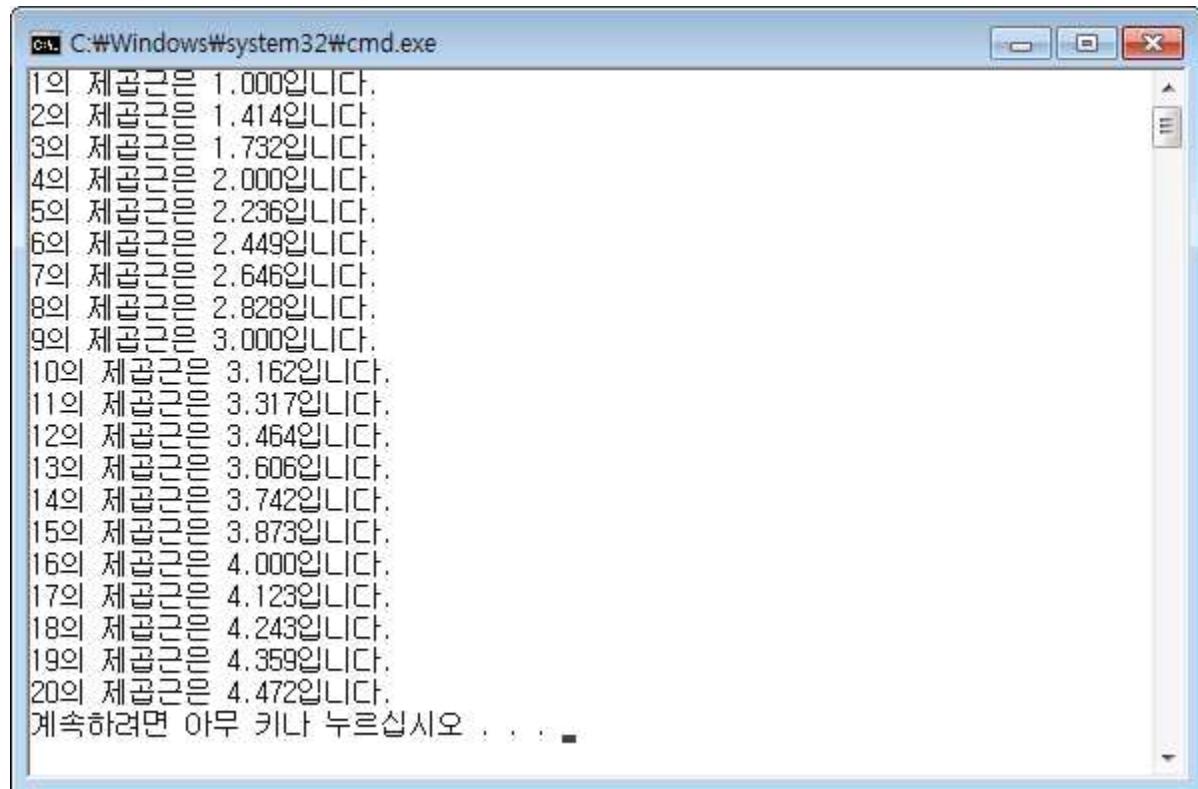
수학 관련 함수들을 프로그램에서 사용하기 위해서는 프로그램 상단에 다음과 같이 `math.h` 헤더 파일의 사용을 정의해야 한다.

```
#include <math.h>
```

그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 1부터 20까지의 숫자들의 제곱근 값을 구하는 구문을 작성해보면 다음과 같다.

```
for (i=1; i<=20; i++){
    printf("%d의 제곱근은 %.3f입니다.\n", i, sqrt( (double)i ));
```

위 프로그램의 실행 화면은 다음과 같다.



위의 구문에서 `(double)i` 라고 해 주는 이유는 `sqrt()` 함수에 넣어 주어야 하는 파라미터의 형식이 실수형이어야 하기 때문에 형 변환을 해주는 것이다.

이번에는 절대값을 구하는 함수인 `fabs()`를 사용해 보자. 10개의 숫자를 입력받은 후에 이 숫자들 중에 절대값이 가장 큰 숫자를 찾아서 출력하는 프로그램을 만들려면 어떻게 하면 될까? 먼저 10개짜리 실수형 숫자의 배열을 만든 다음에 반복문을 10번 돌려가면서 숫자를 하나씩 입력받으면서 배열에 저장한다. 그런 다음에 [Step D]에서 연습한 최댓값 구하는 해결방법을 이용하되 숫자들을 비교하기 전에 절댓값으로 바꾼 값을 비교하도록 하면 된다.

```

// ex_H01.c
#include <stdio.h>
#include <math.h>

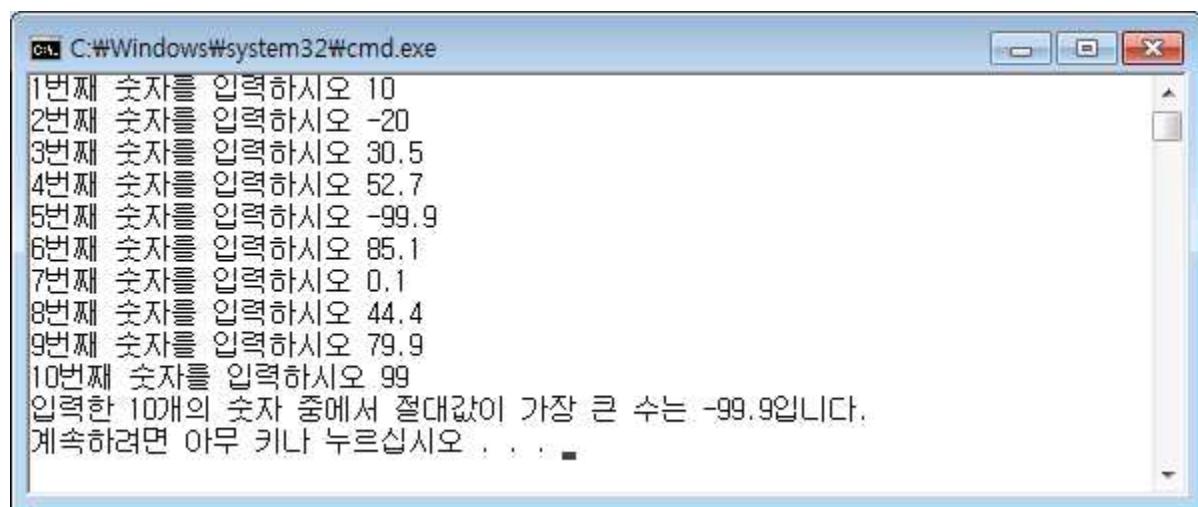
void main()
{
    float number[10];
    int max_index;
    int i;
    for (i=0; i<10; i++){
        printf("%d번째 숫자를 입력하시오 ", i+1);
        scanf("%f", &number[i]);
    }
}

```

```

max_index=0;           // 우선 첫번째 숫자를 가장 큰 수로 정한다.
for (i=1; i<10; i++)   // 두 번째 숫자부터 열 번째 숫자까지 반복해가면서
{
    if ( fabs(number[i]) > fabs(number[max_index]) ) // 절대값을 서로 비교
    {
        max_index = i;          // max_index를 절대값이 큰 수의 인덱스로 변경.
    }
}
printf("10개의 숫자 중에서 절대값이 가장 큰 수는 %.1f입니다.\n", number[max_index]);
}

```



다음은 임의의 숫자를 만들어 사용하는 랜덤 함수를 사용해보자. 랜덤 함수는 숫자를 정해주면 그 숫자만큼의 임의의 수를 자동으로 만들어 리턴하는 것이다. 이 함수들을 사용할 때에는 반드시 다음과 같이 `<stdlib.h>` 헤더파일의 사용을 정의해야 한다.

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

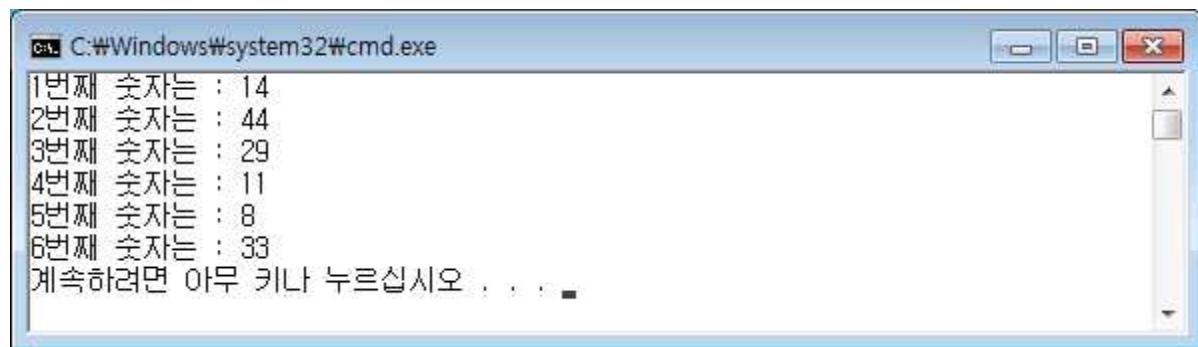
여기에서 사용되는 주요 함수는 `randomize()`와 `random()`이며, 예를 들어 `random(100)`을 호출하면 0에서 99사이의 숫자 100개 중에서 임의의 숫자를 리턴하게 된다. `randomize()` 함수는 `random()` 함수를 사용하기 전에 반드시 실행해주어야 한다. 만일 이 함수를 실행하지 않으면 실행할 때마다 동일한 숫자들이 나오게 된다.

그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 로또 복권은 1부터 45까지의 숫자 6개를 맞추는 것인데, 이 6개의 번호를 임의로 만들어보는 프로그램을 작성하려면 어떻게 할까?

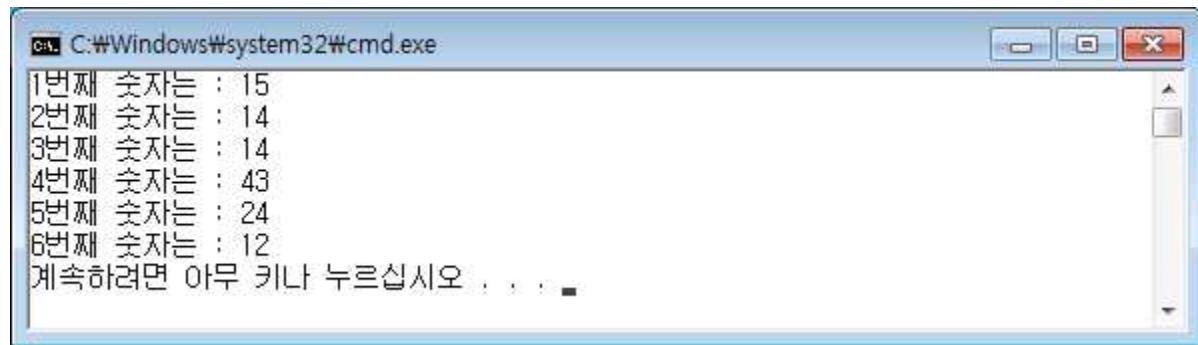
그리고 1부터 45까지의 숫자를 임의로 만들기 위해서는 `random(45)+1`을 넣어주면 된다. `random(45)`은 0부터 44까지의 숫자 중 하나를 임의로 만들어 내기 때문에 여기에 1을 더해주어

야 하는 것이다. 작성된 프로그램은 다음과 같다.

```
randomize();           // 랜덤함수를 사용하기 위해 준비
for (i=0; i<6; i++){
    printf("%d번째 숫자는 : %d\n", i+1, random(45)+1); // 1부터 45까지의 수 만듬
}
```



물론 위 프로그램에서는 1부터 45 사이의 임의의 숫자를 6개 만들어내기 때문에 아래 화면처럼 6개의 숫자들 간에 겹치는 숫자가 나올 가능성이 있다. 절대로 겹치지 않게 만드는 방법은 실습문제를 통해 각자 해결해보도록 하라.

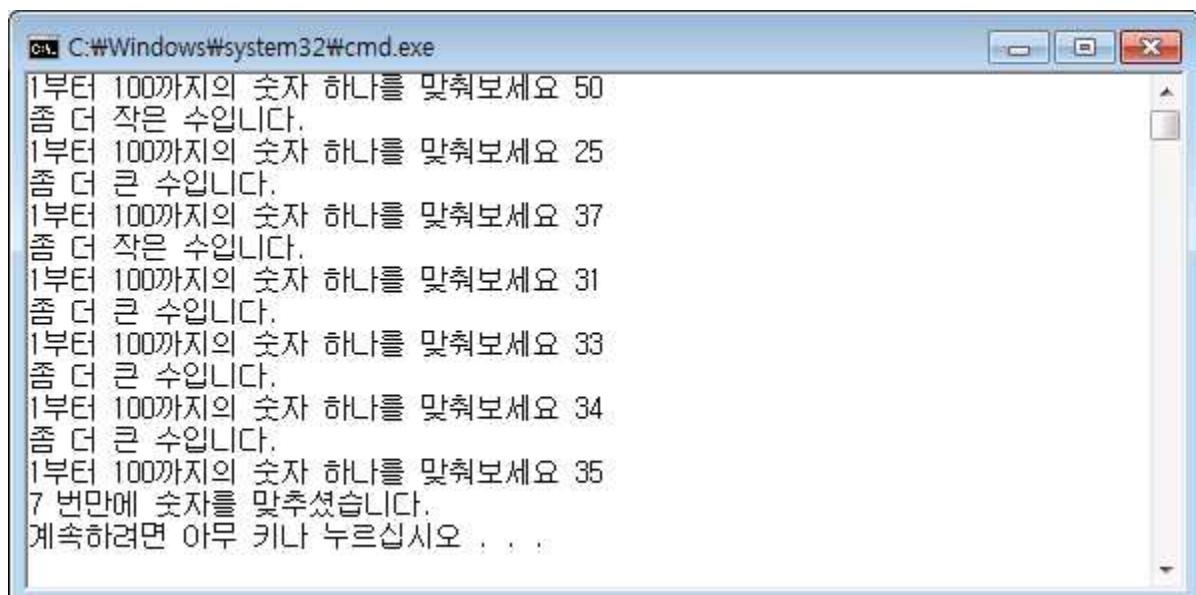


○ 실습 문제

[H01] 숫자 알아 맞추기

1부터 100 사이의 임의의 숫자를 만든 후에 사용자로 하여금 이 숫자를 맞추도록 하라. 사용자가 제시한 숫자보다 큰 수인지 작은 수인지를 알려주면서 몇 번 만에 맞추었는지 출력하라.
변수는 다음과 같이 사용하라.

```
int answer;           // 컴퓨터가 만들어 낸 1부터 100사이의 임의의 숫자  
int number_try;      // 사용자가 제시한 숫자  
int count;           // 사용자가 맞추려고 시도한 횟수
```



[H02] 로또 번호 만들기

1부터 45사이의 임의의 숫자를 6개 만들어 로또 번호를 완성하라. 단, 6개의 번호 중에 중복되는 번호가 없도록 해야 한다. 출력한 후에 사용자에게 "더 만드시겠습니까? (Y/N)"를 물어보고 N을 입력할 때까지 계속해서 반복하라.

변수는 다음과 같이 사용하라.

```
int lotto[6];           // 컴퓨터가 만들어 낸 로또 번호
int count;              // 현재 만들어지고 있는 로또 번호의 순서(0, 1, 2, 3, 4, 5)
char onemore;            // 반복여부를 입력하는 문자 (Y/N)
int i;                  // 반복문을 위한 변수
```

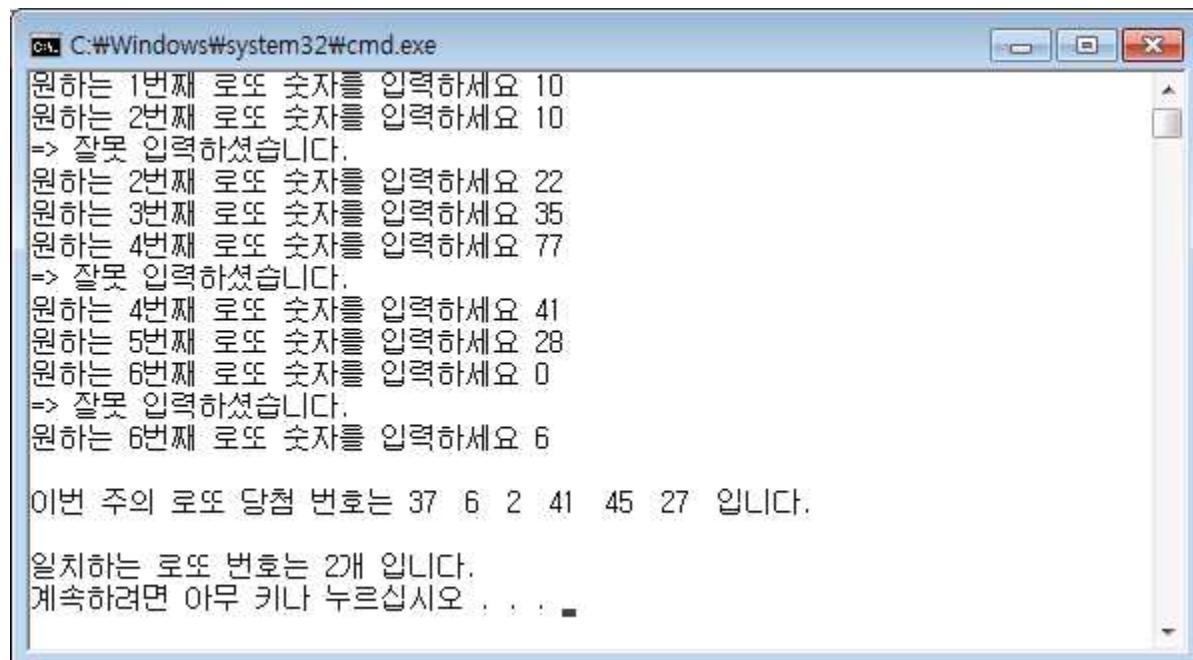
```
C:\Windows\system32\cmd.exe
생성된 로또 번호는 1 25 6 28 35 45 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 32 35 34 31 19 17 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 36 18 21 5 4 28 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 25 44 35 20 10 37 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 41 22 28 14 43 31 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 37 5 35 20 28 41 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 44 9 20 14 39 1 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 11 42 12 35 34 40 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 13 17 45 11 6 34 입니다.
더 만드시겠습니까? (Y/N) Y
생성된 로또 번호는 11 5 6 28 37 30 입니다.
더 만드시겠습니까? (Y/N) N
계속하려면 아무 키나 누르십시오 . . .
```

[H03] 로또 번호 당첨 확인하기

사용자에게 1부터 45사이의 임의의 숫자를 6개 입력받은 후에, 프로그램에서 만든 임의의 로또 번호와 대조하여 몇 개의 숫자가 일치하는지 출력하라. 단, 사용자가 입력한 6개의 번호 중에 중복되는 번호가 없도록 입력받아야 하며, 프로그램에서 만든 임의의 로또 번호에서도 중복되는 번호가 없도록 해야 한다.

변수는 다음과 같이 사용하라.

```
int lotto_com[6];           // 컴퓨터가 만들어 낸 로또 번호
int lotto_user[6];          // 사용자가 입력한 로또 번호
int i;                      // 반복문을 위한 변수
int count;                  // 현재 만들어지고 있는 로또 번호의 순서(0, 1, 2, 3, 4, 5)
int match_count;            // 일치하는 로또 번호의 갯수 (0~6)
```



[H04] 가위바위보 게임하기

다음과 같이 사용자와 컴퓨터가 가위바위보를 하는 프로그램을 만들어라.

- 1) 랜덤으로 가위(1), 바위(2), 보(3) 셋 중에 하나를 만든다.
- 2) 사용자에게 가위(1), 바위(2), 보(3) 중에 하나를 숫자로 입력받는다.
- 3) 사용자가 입력한 것과 컴퓨터가 만들어 낸 것을 비교하여
 컴퓨터가 이기면, "컴퓨터 승!"
 사용자가 이기면, "사용자 승!"
 비기면, "비김~" 으로 출력한다.
- 4) 사용자가 0을 입력할 때까지 위 1)부터 3)을 계속 반복하다가, 끝나면 그동안 컴퓨터와 사용자가 이긴 횟수와 진 횟수, 비긴 횟수를 출력하라.
변수는 다음과 같이 사용하라.

```
int com_finger;           // 컴퓨터가 낸 가위(1), 바위(2), 보(3)
int my_finger;            // 사용자가 낸 가위(1), 바위(2), 보(3)
int score[2][3];          // 결과 횟수 (1행 : 컴퓨터의 승, 무, 패, 2행 : 사용자의 승, 무, 패)
int i, j;                 // 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
가위(1), 바위(2), 보(3)를 입력하세요. 1
컴퓨터가 낸 것은 보입니다. -----> 사용자 승!

가위(1), 바위(2), 보(3)를 입력하세요. 2
컴퓨터가 낸 것은 가위입니다. -----> 사용자 승!

가위(1), 바위(2), 보(3)를 입력하세요. 3
컴퓨터가 낸 것은 보입니다. -----> 비김~

가위(1), 바위(2), 보(3)를 입력하세요. 3
컴퓨터가 낸 것은 바위입니다. -----> 사용자 승!

가위(1), 바위(2), 보(3)를 입력하세요. 2
컴퓨터가 낸 것은 바위입니다. -----> 비김~

가위(1), 바위(2), 보(3)를 입력하세요. 1
컴퓨터가 낸 것은 가위입니다. -----> 비김~

가위(1), 바위(2), 보(3)를 입력하세요. 1
컴퓨터가 낸 것은 보입니다. -----> 사용자 승!

가위(1), 바위(2), 보(3)를 입력하세요. 1
컴퓨터가 낸 것은 바위입니다. -----> 컴퓨터 승

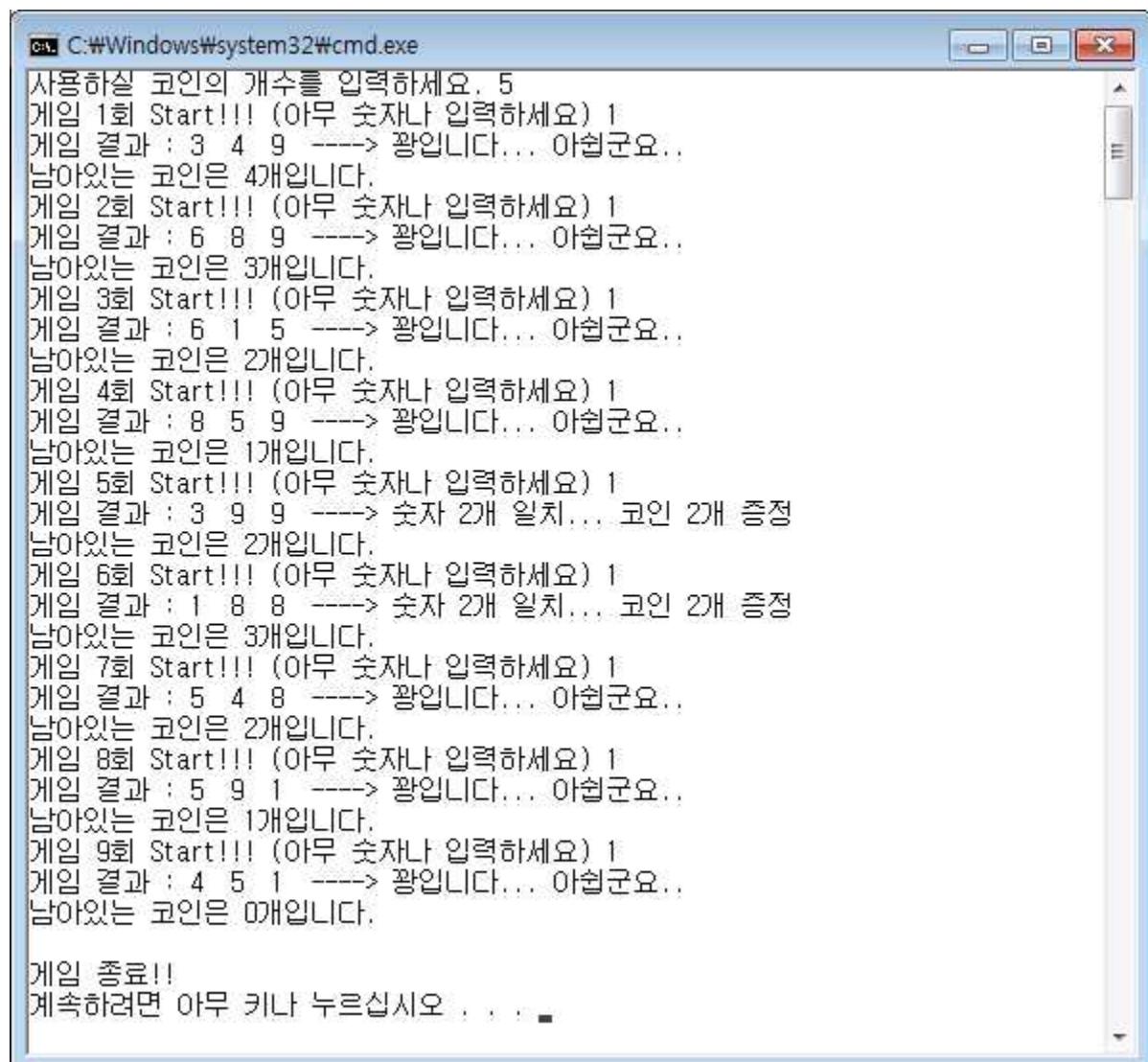
가위(1), 바위(2), 보(3)를 입력하세요. 0

컴퓨터 : 이긴 횟수는 1회, 진 횟수 4회, 비긴 횟수는 3 입니다.
사용자 : 이긴 횟수는 4회, 진 횟수 1회, 비긴 횟수는 3 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[H05] 슬롯머신 만들기

간단한 슬롯 머신을 만들어라. 사용자에게 코인의 개수를 숫자로 입력받은 다음에, 1부터 9까지의 숫자 3개를 랜덤으로 만들어 3개의 숫자가 같으면 상금으로 코인 4개, 2개의 숫자가 같으면 코인 2개를 더해준다. 한 번 할 때마다 코인을 1개 씩 소모하게 되고, 사용자의 코인이 모두 소모될 때까지 게임을 반복시켜라. 변수는 다음과 같이 사용하라.

```
int coins;           // 사용자의 코인 수. (최초에 입력받음)
int number[3];      // 랜덤하게 만들어진 슬롯 머신의 숫자 3개
int dummy;          // 게임스타트를 위한 의미 없는 입력 숫자
int i;              // 반복문을 위한 변수
```

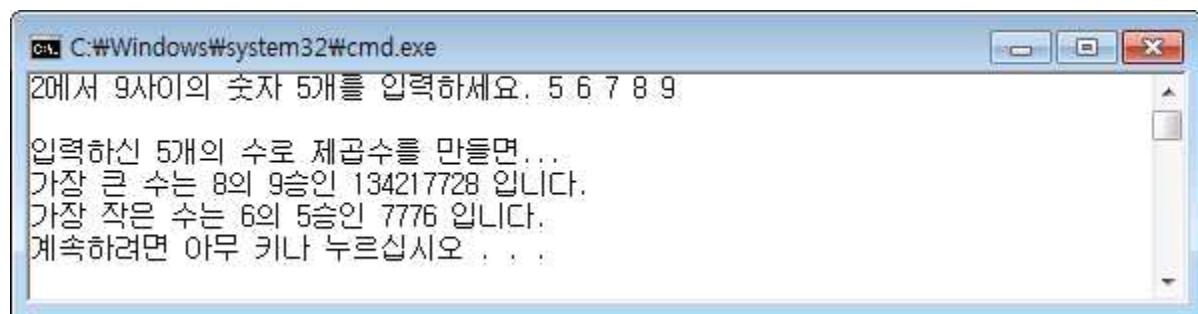
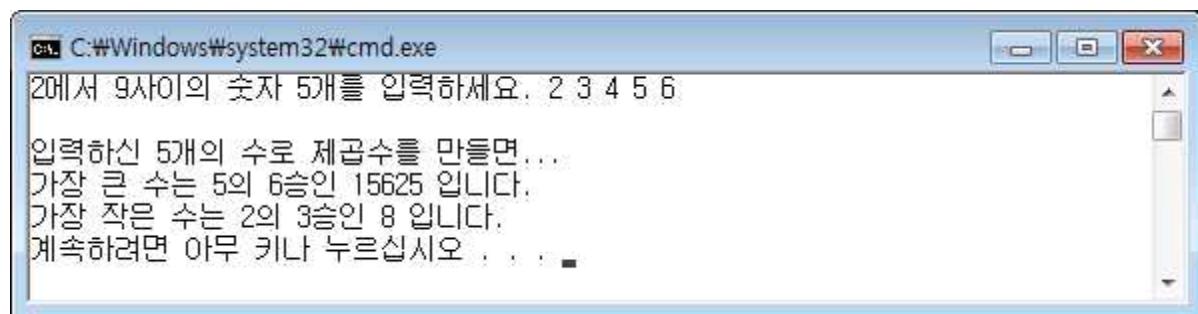


[H06] 5개 숫자의 제곱수 조합 구하기

2에서 9사이의 숫자 5개를 입력받아 배열에 저장한 후, 이 5개의 숫자들 중 임의의 2개의 숫자 a 와 b 를 선택하여 a^b 의 값을 계산하여 이 중에서 가장 큰 수와 가장 작은 수를 만들 수 있는 경우를 찾아내라. 계산할 때에는 함수 `pow()`를 사용하라. 예를 들어 2, 3, 4, 5, 6을 입력한 경우에는 가장 작은 수는 2^3 이고, 가장 큰 수는 5^6 이 된다.

변수는 다음과 같이 사용하라.

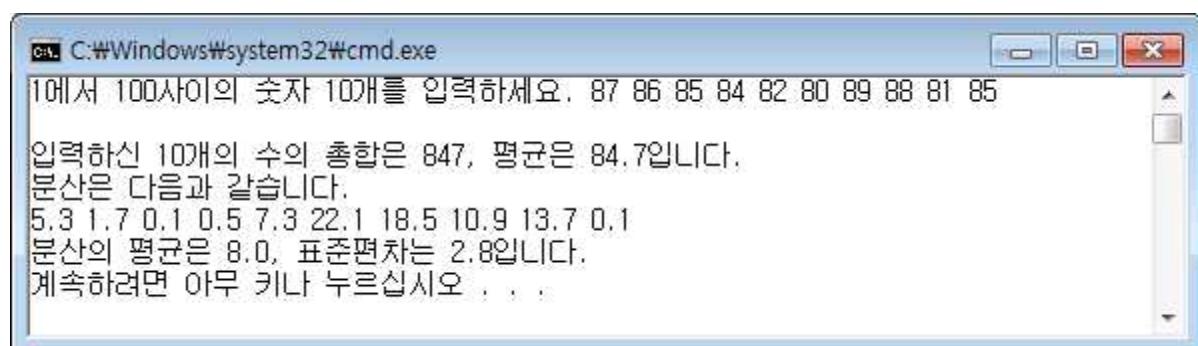
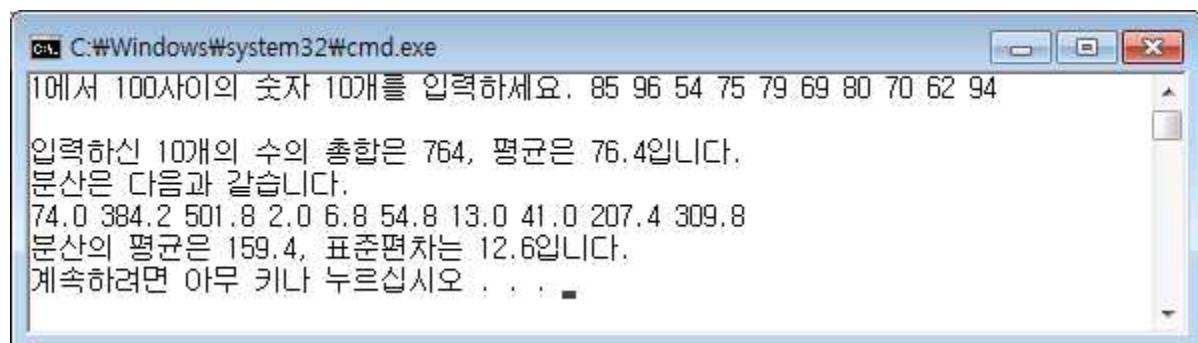
```
int number[5];           // 입력받은 5개의 숫자
int pow_value[5][5];     // 임의의 두 수 a, b로 만들 수 있는  $a^b$ 의 값들
int max, min;           // 최댓값과 최솟값
int max_a, max_b;       // 최댓값을 만들어 내는 경우의 a와 b의 값
int min_a, min_b;       // 최솟값을 만들어 내는 경우의 a와 b의 값
int i, j;               // 반복문을 위한 변수
```



[H07] 표준편차 구하기

1에서 100사이의 숫자 10개를 입력받아 배열에 저장한 후, 반복문을 사용하여 이 숫자들의 총합계와 평균은 구한 후, 표준편차를 구하라. 표준편차를 계산할 때에는 함수 `sqrt()`와 `pow()`를 사용하라. 표준편차는 각 숫자와 평균과의 차이 값을 제곱한 수 10개를 구한 후, 이들 평균값의 제곱근을 계산한 것이다. 변수는 다음과 같이 사용하라.

```
int number[10];           // 입력받은 10개의 숫자
int total;                // 총 합계
float average;            // 평균
float bunsan[10];          // 10개 숫자의 분산 값 (각각 (숫자-평균)의 제곱)
float total_bunsan;        // 분산의 총합
float average_bunsan;      // 분산의 평균
float std_dev;             // 표준편차 (분산 평균의 제곱근)
int i;                     // 반복문을 위한 변수
```



[Step H2] C언어의 주요 함수 사용하기(문자열)

C 프로그램에서는 종류별로 언어 자체적으로 제공하는 라이브러리 함수들이 있는데 이번 단계에서는 자주 사용되는 C 라이브러리 함수들 중에 문자열 관련 프로그램을 연습하려고 한다. 문자열 관련 함수들은 C언어에서 많이 사용되므로 주어진 예제를 하나하나 풀어보면서 익히도록 한다.

먼저 문자열을 입출력 하는 방법을 보도록 하자.

지금까지 만들어 본 프로그램에서는 여러 가지 타입의 값들을 입력받기 위해서 scanf함수를 사용하고 출력하기 위해서 printf를 사용하였는데 문자열을 입출력할 때에도 이 두 함수를 사용하여 입출력이 가능하다.

문자열을 입력할 때에는 먼저 char 배열을 사용하여 문자열 변수를 선언하고 입출력을 할 때 “%s”를 사용한다. 또한 문자열을 입력하기 위해 scanf를 사용할 때 주의할 점은 & 표시를 하지 않는다는 점이다.

```
char str[80];

printf("당신의 이름을 입력하세요:");
scanf("%s", str);
printf("입력하신 이름은 %s입니다.", str);
```

또는 문자열 전용 입출력 함수를 다음과 같이 사용할 수 있으며 gets함수는 공백과 함께 문자열을 입력 받을 수 있다.

```
char str[80];

printf("당신의 이름을 입력하세요:");
gets(str);
printf("입력하신 이름은");
puts(str);
printf("입니다.");
```

주로 사용되는 문자열 관련 함수들을 살펴보도록 하자.

함수 원형	설명
char *strcpy (char *dest, const char *src);	src의 문자열을 dest에 복사한다.
char *strncpy (char *dest, const char *src, size_t n);	문자열을 복사할 때 지정된 문자의 수만큼 복사한다.
size_t strlen(const char *str);	주어진 문자열의 개수를 리턴한다.
char *strcat(char *dest, const char *src);	ex의 값을 리턴한다.
int strcmp(const char *str1, const char *str2);	str1이 str2보다 사전에서 뒤에 나오면 1, str1이 str2보다 사전에서 앞에 나오면 0, 같은 문자열이면 0반환

문자열 관련 함수들을 프로그램에서 사용하기 위해서는 프로그램 상단에 다음과 같이 **string.h** 헤더파일의 사용을 정의해야 한다.

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

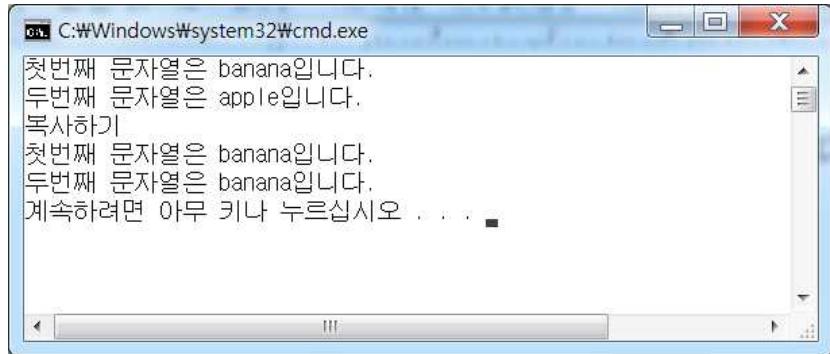
그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 문자열 str1를 문자열 str2에 복사하여 화면에 출력하는 것은 다음과 같다.

```
char str1[20] = "banana";
char str2[20] = "apple";

printf("첫번째 문자열은 %s입니다.\n", str1);
printf("두번째 문자열은 %s입니다.\n", str2);

printf("복사하기\n");
strcpy(str2, str1);
printf("첫번째 문자열은 %s입니다.\n", str1);
printf("두번째 문자열은 %s입니다.\n", str2);
```

위 프로그램의 실행 화면은 다음과 같다.



이번에는 `strcpy` 함수를 사용하지 않고 문자열을 복사하는 나만의 `strcpy` 함수를 구현하여 프로그램을 만들어 보자. 이 함수의 이름은 `m strcpy` 라고 하자.

```

// ex_H21.c
#include <stdio.h>
#include <string.h>
void m strcpy(char *str1, char *str2);
void main()
{
    int kor, eng;
    int avg;
    char hakjum[20] = "학점은 여기에";
    printf("국어 점수를 입력하세요");
    scanf("%d", &kor);

    printf("영어 점수를 입력하세요");
    scanf("%d", &eng);

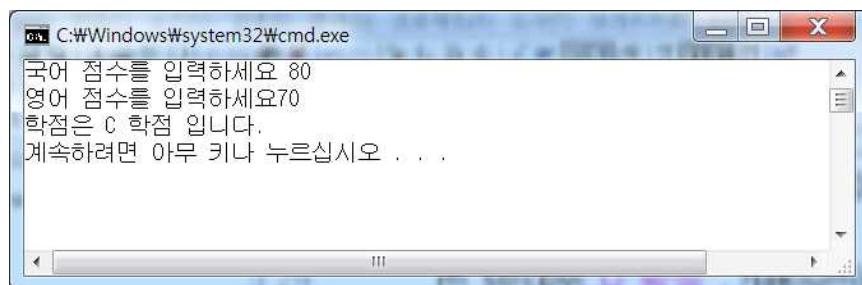
    avg = (kor + eng)/2;

    if(avg >= 90)
        m strcpy("A 학점", hakjum);
    else if(avg >= 80)
        m strcpy("B 학점", hakjum);
    else if(avg >= 70)
        m strcpy("C 학점", hakjum);
    else if(avg >= 60)
        m strcpy("D 학점", hakjum);
    else
        m strcpy("F 학점", hakjum);

    printf("학점은 %s 입니다.\n", hakjum);
}

```

```
void m_strcpy(char *str1, char *str2){  
    while( *str1 != '\0'){  
        *str2 = *str1;  
        str1++;  
        str2++;  
    }  
    *str2 = '\0';  
}
```



○ 실습 문제

[H2-1] 원하는 개수의 문자만을 복사하기

문자열 한 개를 입력받고, 원하는 개수를 입력받아 다른 문자열 변수에 입력한 개수만큼의 문자를 복사하여 출력하도록 하라. 문자열을 입력할 때 0을 입력하면 프로그램이 종료한다.

변수는 다음과 같이 사용하라.

```
char str1[80];      // 사용자가 입력한 문자열을 저장
int count;          // 사용자가 입력한 문자열 개수
```

복사하는 함수는 기존의 문자열 함수인 `strncpy`를 사용하지 말고 아래의 함수를 사용하여 구현하라.

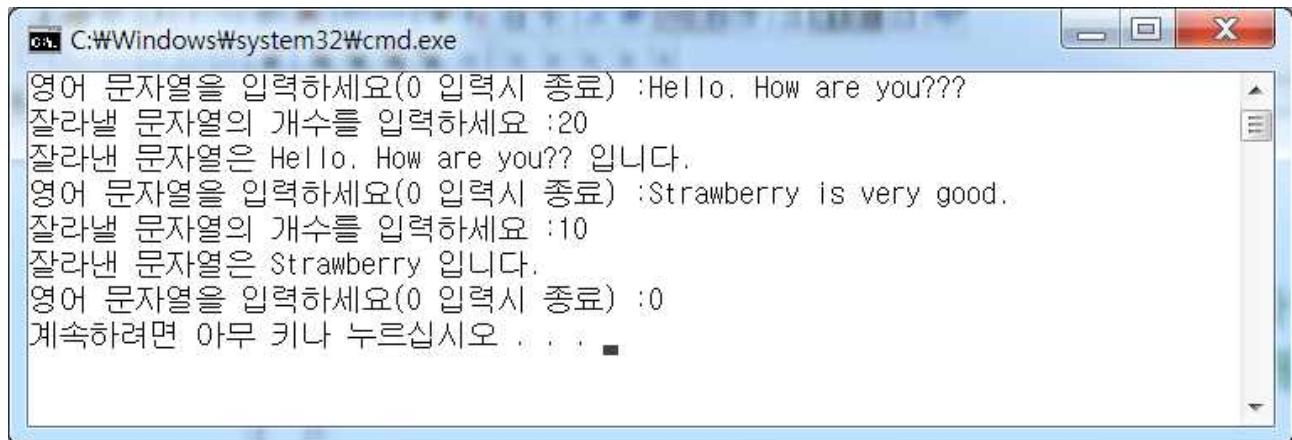
함수 원형은 다음과 같다.

```
char *m_strncpy(char *str1, int count);
```

파라미터) 복사할 원본 문자열 변수 및 잘라낼 문자 개수

리턴 값) 복사된 문자열

수행내용) `str1`의 문자열에서 `count`개수만큼 잘라서 복사한 값을 리턴하는 함수이다.



[H2-2] 문자열 긴 단어 출력하기

문자열 두 개를 입력받아서 두 문자열의 길이를 구하여 두 문자열 중에서 긴 문자열을 화면에 출력한다.

문자열을 입력할 때 0을 입력하면 프로그램이 종료한다.

변수는 다음과 같이 사용하라.

```
char str1[80];      // 사용자가 입력한 문자열을 저장
char str2[80];      // 사용자가 입력한 문자열을 저장
```

문자열 길이를 구하는 함수는 `strlen` 함수를 사용하고, 또한 다음의 함수를 만들어서 두 문자열을 비교하여 둘 중의 긴 문자열이 나오도록 만들어 본다.

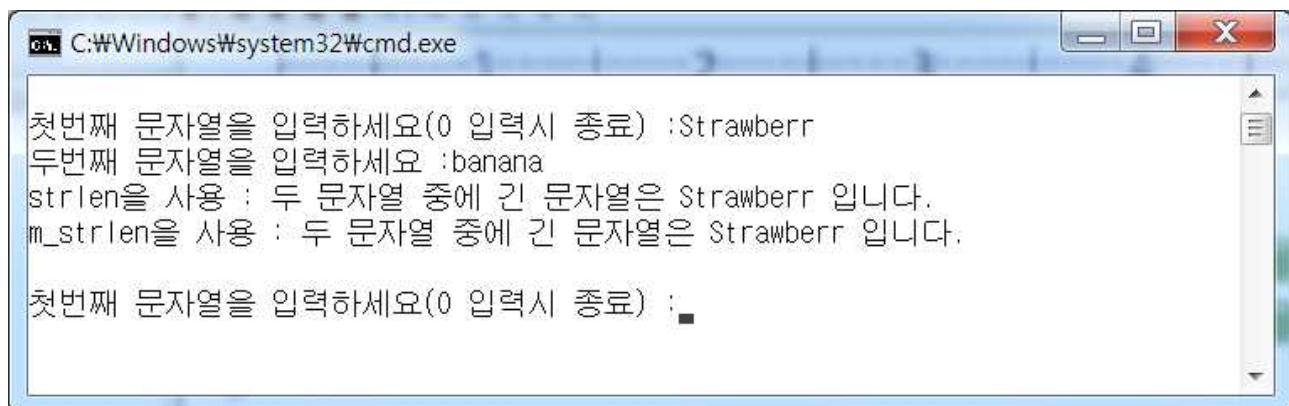
함수 원형은 다음과 같다.

```
char * m_strlen(char *str1, char *str2);
```

파라미터) 비교할 두 개의 문자열

리턴 값) 둘 중에 긴 문자열

수행내용) 두 문자열을 넣으면 두 개 중에 긴 문자열이 리턴되는 함수이다.



[H2-3] 사용자 ID와 암호 출력

사용자의 ID와 암호를 입력받아서 두 개를 출력하는데 암호는 3글자 이상이면 3글자부터 글자의 수까지 *를 입력하여 표현한다. 아래의 화면과 같은 결과가 나오도록 프로그램을 만들자.
문자열을 입력할 때 0을 입력하면 프로그램이 종료한다.
변수는 다음과 같이 사용하라.

```
char userid[80];      // 사용자가 입력한 문자열을 저장  
char password[80];    // 사용자가 입력한 문자열을 저장
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following interaction:

```
사용자 ID를 입력하세요 : hong
사용자 암호를 입력하세요 : 12
==>암호는 3글자 이상 입력하세요
사용자 암호를 입력하세요 : 1234
사용자 이름을 입력하세요 : 흥길동
*****
사용자 ID : hong
사용자 암호 : 12**
사용자 이름 : 흥길동
*****
==>저장되었습니다.

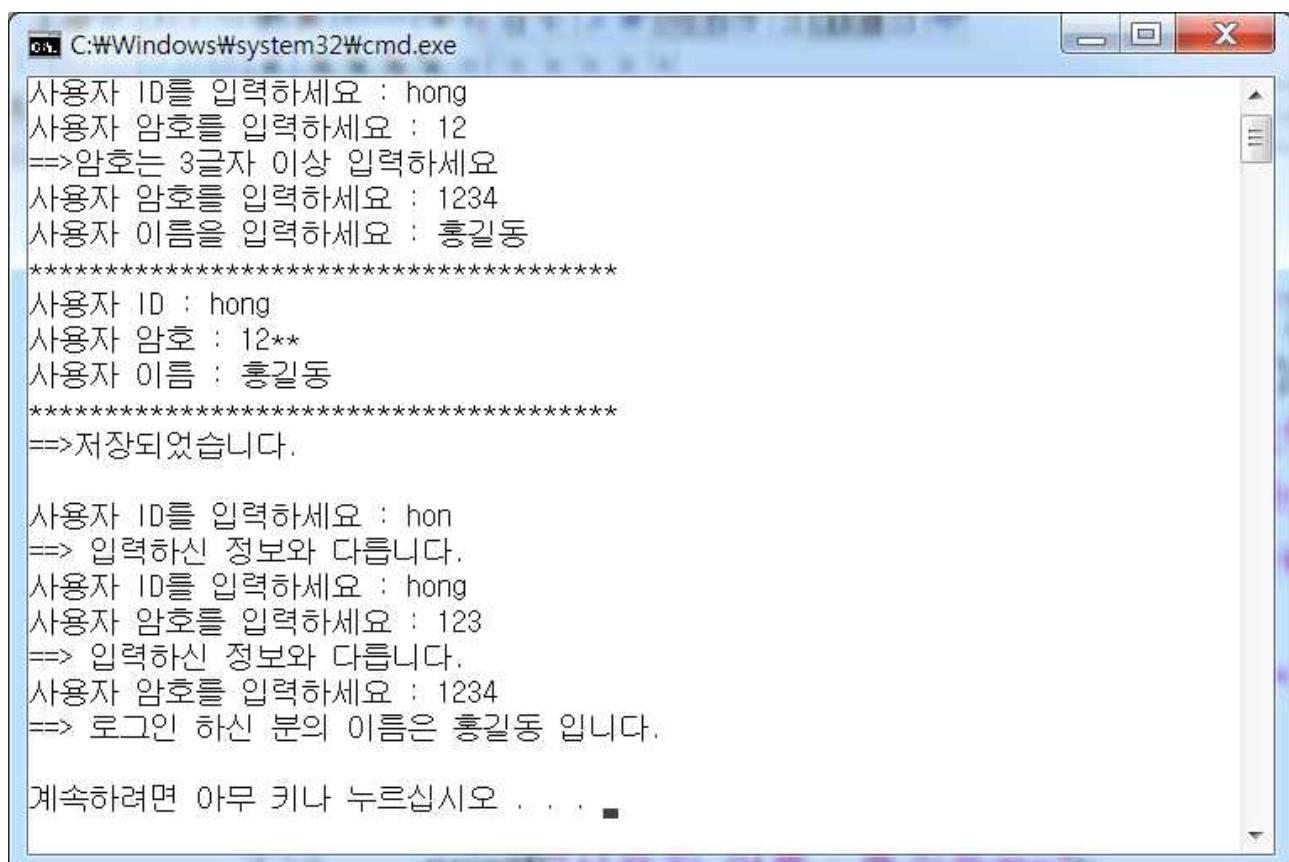
사용자 ID를 입력하세요 : hon
==> 입력하신 정보와 다릅니다.
사용자 ID를 입력하세요 : hong
사용자 암호를 입력하세요 : 123
==> 입력하신 정보와 다릅니다.
사용자 암호를 입력하세요 : 1234
==> 로그인 하신 분의 이름은 흥길동 입니다.

계속하려면 아무 키나 누르십시오 . . . ■
```

[H2-4] 사용자 ID와 암호 출력

사용자의 ID와 암호를 입력받아서 두 개를 출력하는데 암호는 3글자 이상이면 3글자부터 글자의 수까지 * 를 입력하여 표현한다. 아래의 화면과 같은 결과가 나오도록 프로그램을 만들자.
문자열을 입력할 때 0을 입력하면 프로그램이 종료한다.
변수는 다음과 같이 사용하라.

```
char userid[80];      // 사용자가 입력한 문자열을 저장
char password[80];    // 사용자가 입력한 문자열을 저장
```



영역 3 : 함수 만들어 사용하기

영역 3에서는 프로그램을 제작할 때에 필수 능력이라고 할 수 있는 함수 제작을 연습한다. 프로그램에서 처리해야 하는 작업들을 내용과 성격에 따라 서브 함수로 제작하여 역할을 분담하는 기술은 반드시 터득해야 하는 기술이다. 영역 3의 연습을 마치고 나면 영역 1과 영역 2에서 습득한 프로그래밍의 제어구조 기술을 메인 함수와 서브 함수 모두에 골고루 적용할 수 있게 될 것이다.

함수는 별도로 작업해야 하는 프로그램 구문들을 따로 정해놓은 것이며, 함수를 호출할 때에 전달해야 하는 데이터가 있으면 이를 파라미터로 전달할 수 있다. 그리고 함수의 동작이 마친 후에 결과와 값을 돌려받아야 하는 경우에는 리턴 값을 명시해서 돌려줄 수 있다. 물론 함수에 전달할 데이터가 없는 경우에는 파라미터 없이 함수를 제작하면 되며, 리턴할 결과 값이 없는 경우에는 리턴 값이 없다고 명시하면 된다.

영역 3의 실습 문제에서는 사용해야 하는 변수를 알려주지 않는다. 여러분이 문제를 충분히 읽고 분석한 후, 적절하게 필요한 변수들을 선언하여 사용하기 바란다.

영역 3는 다음과 같이 4개의 단계로 구성된다.

Step I : 리턴 값이 없는 함수 만들기

Step J : 파라미터는 없고 리턴 값만 있는 함수 만들기

Step K : 파라미터와 리턴 값이 모두 있는 함수 만들기

Step L : 재귀함수 만들기

[Step 1] 리턴 값이 없는 함수 만들기

이번 단계에서는 가장 기본적인 함수들을 만드는 연습을 하려고 한다. 함수를 호출하게 되면 뭔가 정해진 작업을 수행하기만 하면 함수의 역할이 끝나는 경우이다. 이런 경우에는 함수의 리턴 값이 없다(**void**)고 말한다. 예를 들어 다음 예제와 같이 화면에 구분선을 출력하는 경우가 여러 번 있다 고 가정한다면, 이 작업이 필요할 때마다 해당 구문을 일일이 프로그램에 넣기 보다는 함수로 만들어 호출하게 하면 프로그램을 읽거나 만들 때에 좀 더 편리하다. 소스에서 보는 바와 같이 메인 함수 위에 함수 원형을 정의하고, 메인 함수 아래에 함수의 내용을 구현하였다. 함수 원형의 정의를 보면 파라미터가 없고 리턴 값도 없음을 명시하기 위해 **void**라는 키워드를 넣어 주었다.

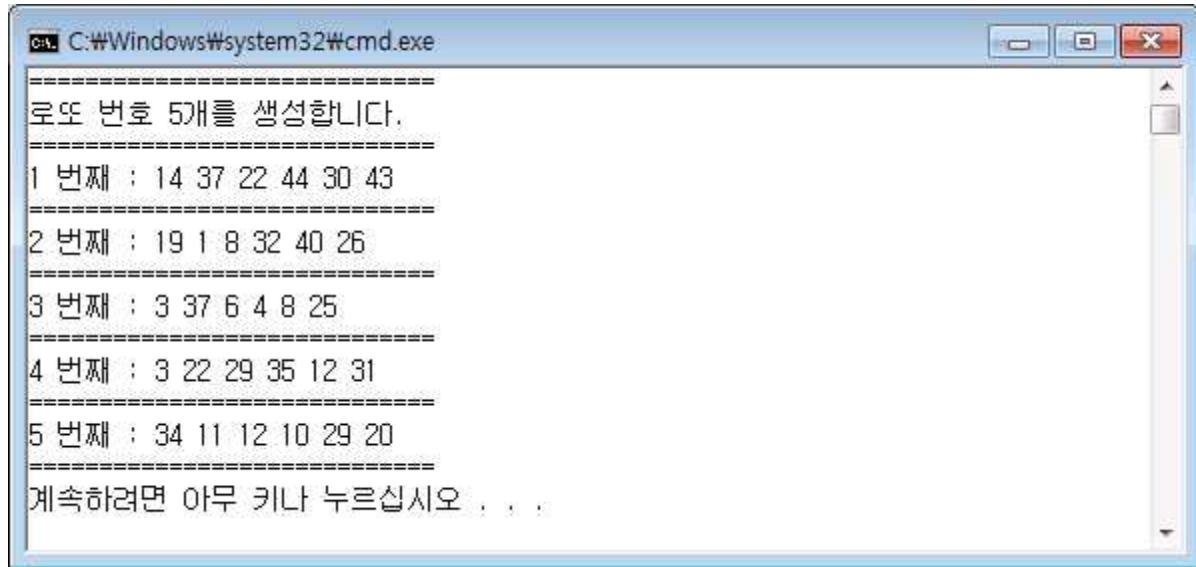
```
// 예제 ex_I_1.c
#include <stdio.h>
#include <stdlib.h>

void print_border(void); // 함수 원형 정의

void main()
{
    int i, j;
    print_border(); // 함수 호출

    randomize();
    printf("로또 번호 5개를 생성합니다.\n");
    print_border(); // 함수 호출
    for (i=0;i<5;i++){
        printf("%d 번째 : ", i+1);
        for (j=0; j<6; j++){
            printf("%d ", random(45)+1);
        }
        printf("\n");
        print_border(); // 함수 호출
    }
}

void print_border() // 함수내용 구현
{
    printf("=====\\n");
}
```



이번에는 파라미터가 있고 리턴 값이 없는 함수를 만들어보자. 위의 예제에서 print_border() 함수는 미리 정해진 개수만큼의 테두리('=')을 출력하고 있는데, 이 함수에 테두리의 개수를 지정하는 값을 파라미터로 주어서 출력 모양을 바꿔보자. 그러면 함수 원형은 다음과 같이 바뀌어야 한다.

```
void print_border(int size); // 변경된 함수 원형 정의, 파라미터로 size가 추가되었다.
```

이에 따라 함수의 구현 내용 역시 다음과 같이 바뀌어야 한다.

```
void print_border(int size) // 변경된 함수내용 구현
{
    int i;
    for (i=0; i<size; i++){ // 파라미터로 넘어온 size는 로컬변수처럼 사용하면 된다.
        printf("=");
    }
    printf("\n");
}
```

이렇게 바뀐 함수를 사용하도록 변경된 프로그램은 다음과 같다.

```
// 예제 ex_I_2.c
#include <stdio.h>
#include <stdlib.h>

void print_border(int size); // 함수원형 정의

void main()
{
```

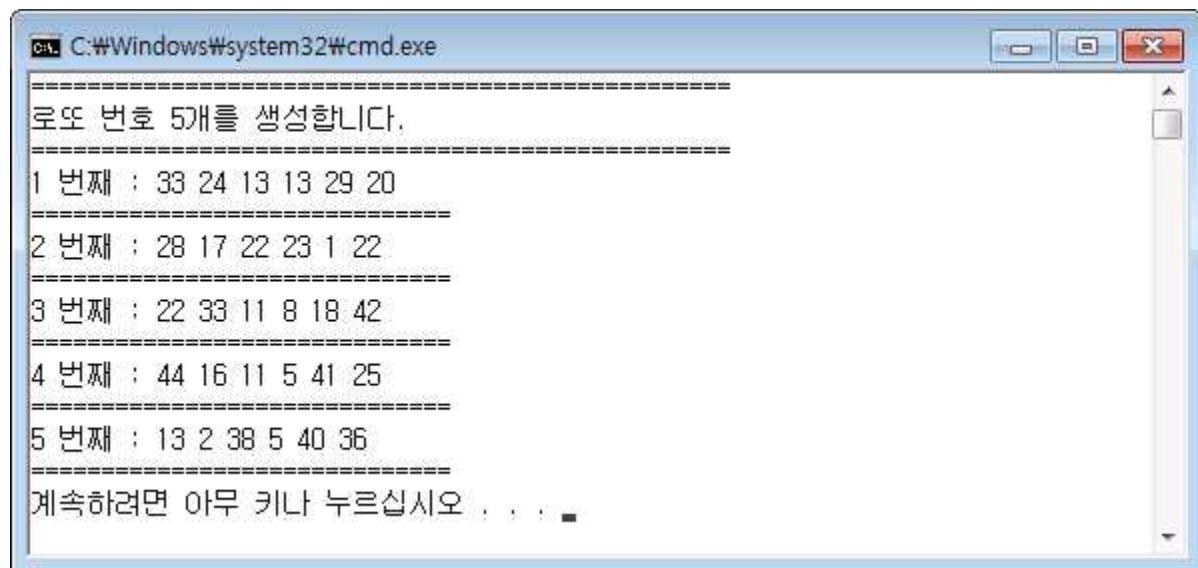
```

int i, j;
print_border(50); // 함수 호출

randomize();
printf("로또 번호 5개를 생성합니다.\n");
print_border(50); // 함수 호출
for (i=0;i<5;i++){
    printf("%d 번째 : ", i+1);
    for (j=0; j<6; j++){
        printf("%d ", random(45)+1);
    }
    printf("\n");
    print_border(30); // 함수 호출
}
}

void print_border(int size) // 변경된 함수내용 구현
{
    int i;
    for (i=0; i<size; i++)// 파라미터로 넘어온 size는 로컬변수처럼 사용하면 된다.
        printf("=");
    }
    printf("\n");
}

```



○ 실습 문제

[I01] 메뉴판 보여주는 함수 만들기

어떤 식당의 메뉴판을 보여주는 함수를 만들어라. 메인 함수에서 이 함수를 호출하여 메뉴판을 보여준 다음 메뉴번호를 입력받아 해당 메뉴의 가격을 합산하되 메뉴선택 종료를 의미하는 5값을 입력 받을 때까지 계속 반복하여 메뉴를 선택하게 하고, 선택종료 후 선택한 메뉴의 총 합계금액을 출력하라.

단, 사용할 메뉴는 다음과 같다.

1. 피자(15,000원)
2. 스파게티(10,000원)
3. 샐러드(7,000원)
4. 음료수(2,000원)
5. 종료

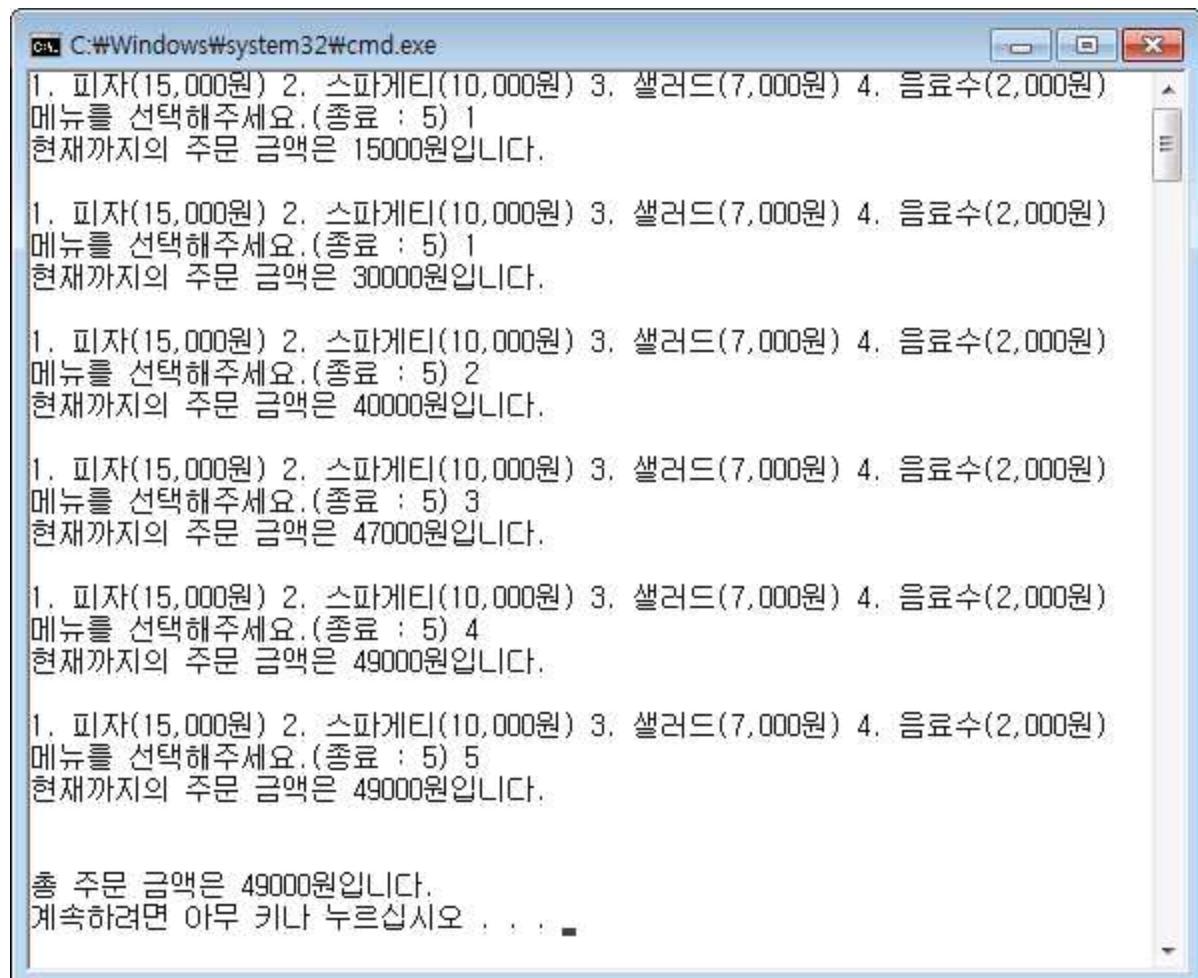
함수 원형은 다음과 같다.

```
void ShowMenu();
```

파라미터) 없음

리턴 값) 없음

수행내용) 메뉴판 출력



[102] 개수만큼 별 찍는 함수 만들기

파라미터로 숫자 하나를 넘겨주면 이 숫자만큼 별('*')을 화면에 출력하는 함수를 만들어라. 그리고 사용자로부터 10개의 숫자를 입력받은 후, 이 함수를 사용해서 10줄에 걸쳐서 그 숫자만큼의 별을 화면에 출력시켜라.

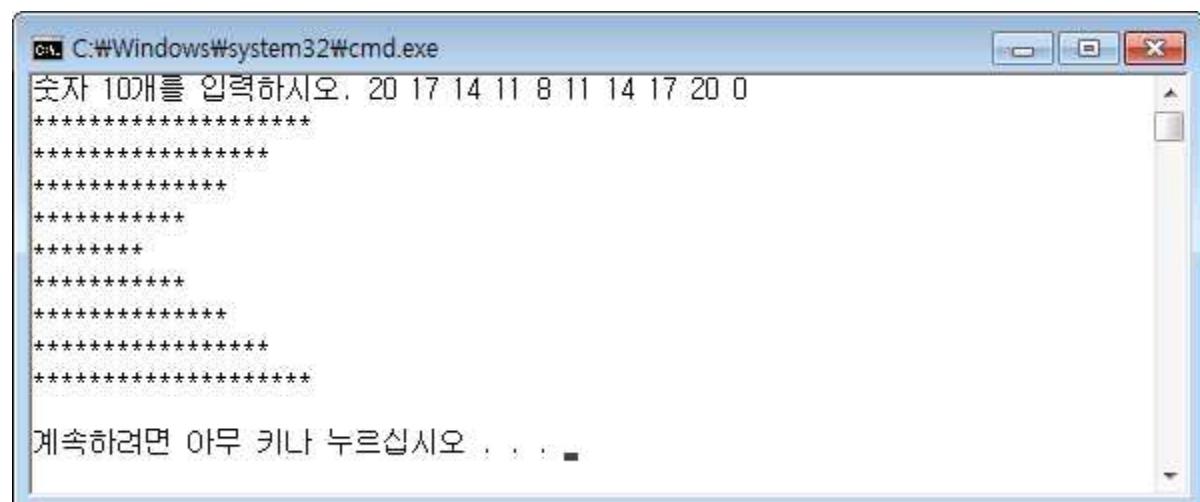
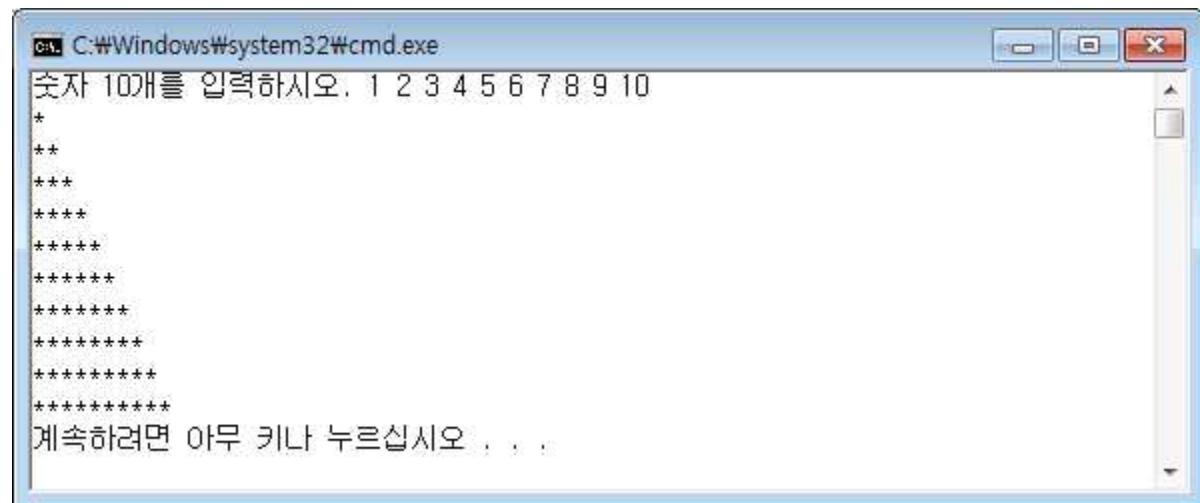
함수 원형은 다음과 같다.

```
void PrintStar(int size);
```

파라미터) size : 출력할 별의 갯수

리턴 값) 없음

수행내용) size 개수만큼 '*' 문자 출력 후 개행



[I03] 특정 문자를 개수만큼 찍는 함수 만들기

파라미터로 문자 하나와 숫자 하나를 넘겨주면 이 숫자만큼 넘겨받은 문자를 화면에 출력하는 함수를 만들 어라. 그리고 사용자로부터 문자 하나와 10개의 숫자를 입력받은 후, 이 함수를 사용해서 10줄에 걸쳐서 그 숫자만큼의 문자를 화면에 출력시켜라.

함수 원형은 다음과 같다.

```
void PrintChar(int size, char ch);
```

파라미터) size : 출력할 문자의 갯수, ch : 출력할 문자

리턴 값) 없음

수행내용) size 개수만큼 ch 문자 출력 후 개행

```
C:\Windows\system32\cmd.exe
사용할 문자를 입력하시오. $
숫자 10개를 입력하시오. 50 45 40 35 40 45 50 45 40 35
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
사용할 문자를 입력하시오. ~
숫자 10개를 입력하시오. 5 10 15 20 25 30 35 40 45 50
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
~~~~~
계속하려면 아무 키나 누르십시오 . . .
```

[I04] 빈칸과 함께 특정 문자를 개수만큼 찍는 함수 만들기

파라미터로 문자 하나와 숫자 두 개를 넘겨주면 한 줄에 첫 번째 숫자만큼 빈칸을 출력한 후, 바로 이어서 두 번째 숫자만큼 넘겨받은 문자를 화면에 출력하는 함수를 만들어라. 그리고 사용자로부터 모양(문자 하나)과 높이, 여백을 입력받은 후, 이 함수를 사용해서 입력한 크기만큼의 여백과 높이를 갖는 우직각 삼각형을 입력한 문자모양으로 화면에 출력시켜라.

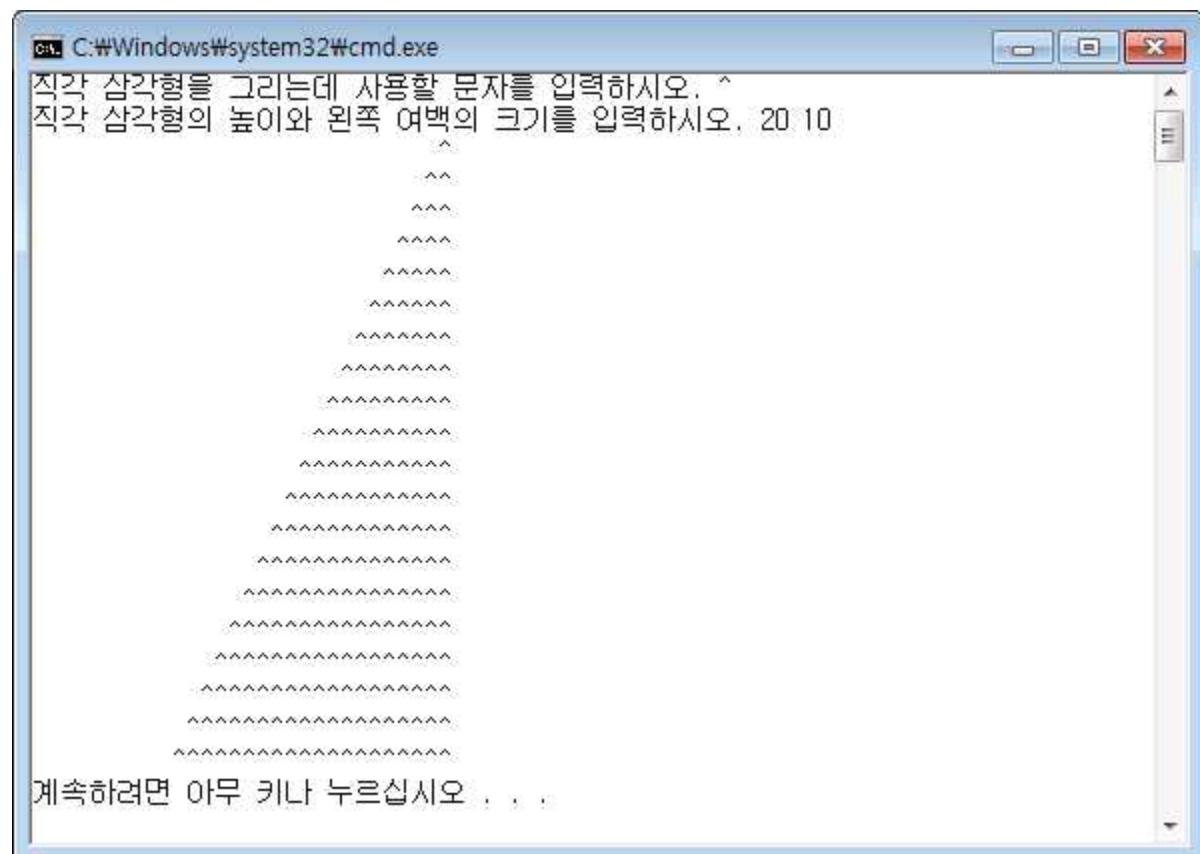
함수 원형은 다음과 같다.

```
void PrintCharWithBlank(int blanks, int size, char ch);
```

파라미터) blank : 빈칸의 개수, size : 출력할 문자의 개수, ch : 출력할 문자

리턴 값) 없음

수행내용) blanks 숫자만큼 빈칸 출력, size 개수만큼 ch 문자 출력 후 개행



[I05] 비만 판정

10명의 신장(cm단위)과 체중(kg단위)를 입력받으면서 AskBiman() 함수를 통해 이들이 비만도를 출력하고 다음 기준에 따라 비만여부를 판정하여 출력하라.

비만도 수치 = 체중(kg) / (신장(m)의 제곱) 으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.

비만도 수치에 따른 비만도 판정

1. 18.5 미만 : 저체중
2. 18.5 ~ 23 미만 : 정상체중
3. 23~25미만 : 과체중
4. 25~30미만 : 경도비만
5. 30이상 : 고도비만

함수 원형은 다음과 같다.

```
void AskBiman(int height, int weight);
```

파라미터) height : 신장(cm), weight : 체중(kg)

리턴 값) 없음

수행내용) 비만도 계산 후 판정결과 출력

```
C:\Windows\system32\cmd.exe
1번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 176 80
경도 비만입니다.
2번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 182 99
경도 비만입니다.
3번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 168 54
정상입니다.
4번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 170 45
저체중입니다.
5번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 167 60
정상입니다.
6번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 176 98
고도 비만입니다.
7번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 152 60
경도 비만입니다.
8번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 172 80
경도 비만입니다.
9번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 167 70
경도 비만입니다.
10번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 168 75
경도 비만입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[Step J] 파라미터는 없고 리턴 값만 있는 함수 만들기

이번 단계에서 연습할 함수의 형태는 파라미터 없이 리턴 값만 돌려주는 함수이다. 예를 들어 입력 받아야 하는 데이터가 바르게 입력되도록 점검해야 하는 일이 반복된다면 이 작업을 함수로 제작해서 편리하게 불러 오면 된다. 성적을 처리하는 프로그램에서 점수를 입력받을 때에 반드시 0에서 100사이의 정수를 입력받아야 하는데 이 범위에 맞는 수인지 검사하는 함수를 만들어보자. 먼저 이 함수의 원형은 다음과 같이 정한다.

```
int GetScore( void ); // 파라미터는 없고, 리턴 값만 정수형으로 정한다.
```

이 함수의 구현 내용은 다음과 같다.

```
int GetScore()
{
    int jumsu;
    while(1) {
        printf("점수를 입력하시오. ");
        scanf("%d", &jumsu);
        if (jumsu < 0 || jumsu > 100)
            printf("잘못 입력하였습니다. \n");
        else
            return jumsu; // 올바른 범위의 숫자이므로 리턴한다.
    }
}
```

이 함수를 사용하여 국어, 영어, 수학 점수를 입력받아 총점과 평균을 계산하는 프로그램은 다음과 같다.

```
/*
ex_J.c
*/
#include <stdio.h>
int GetScore( void ); // 함수 원형 선언

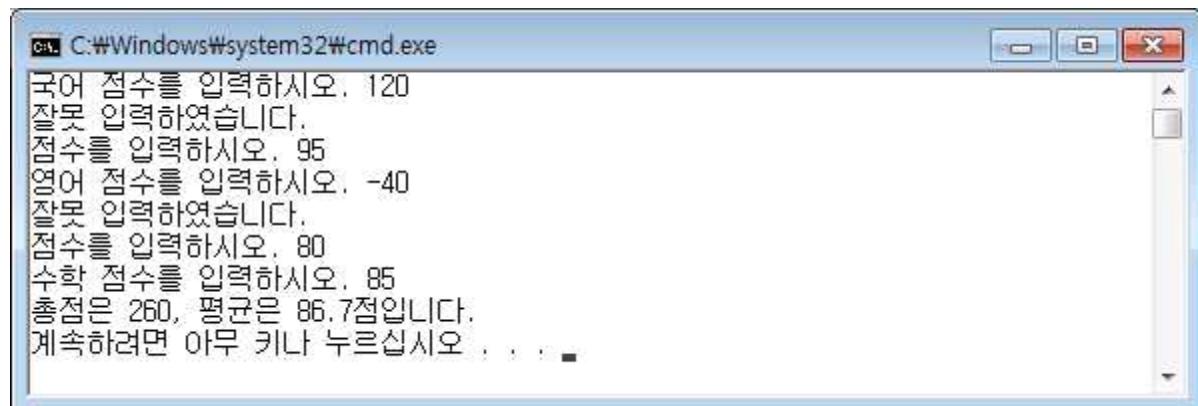
void main()
{
    int jumsu[3]; // 국어, 영어, 수학 점수
    int sum; // 총점
    float average; // 평균
    int i;
```

```

printf("국어 ");
jumsu[0] = GetScore();           // 0부터 100사이의 숫자 검사하며 입력
printf("영어 ");
jumsu[1] = GetScore();           // 0부터 100사이의 숫자 검사하며 입력
printf("수학 ");
jumsu[2] = GetScore();           // 0부터 100사이의 숫자 검사하며 입력
sum = 0;
for (i=0; i<3 ; i++)
    sum = sum + jumsu[i];
average = sum / 3.0;
printf("총점은 %d, 평균은 %.1f점입니다.\n", sum, average);
}

int GetScore()
{
    int jumsu;
    while(1) {
        printf("점수를 입력하시오. ");
        scanf("%d", &jumsu);
        if (jumsu < 0 || jumsu > 100)
            printf("잘못 입력하였습니다. \n");
        else
            return jumsu;           // 올바른 범위의 숫자이므로 리턴한다.
    }
}

```



○ 실습 문제

[J01] 메뉴 번호 받아오는 함수 만들기

어떤 식당의 메뉴판을 보여준 후에 메뉴번호를 입력받아 그 가격을 리턴하는 함수를 만들어라. 메인 함수에서 이 함수를 호출하여 리턴 받은 가격을 합산하되 메뉴선택 종료를 의미하는 5를 리턴 받을 때까지 계속 반복하여 메뉴를 선택하게 하고, 선택종료 후 선택한 메뉴의 총 합계금액을 출력하라.

단, 사용할 메뉴는 다음과 같다.

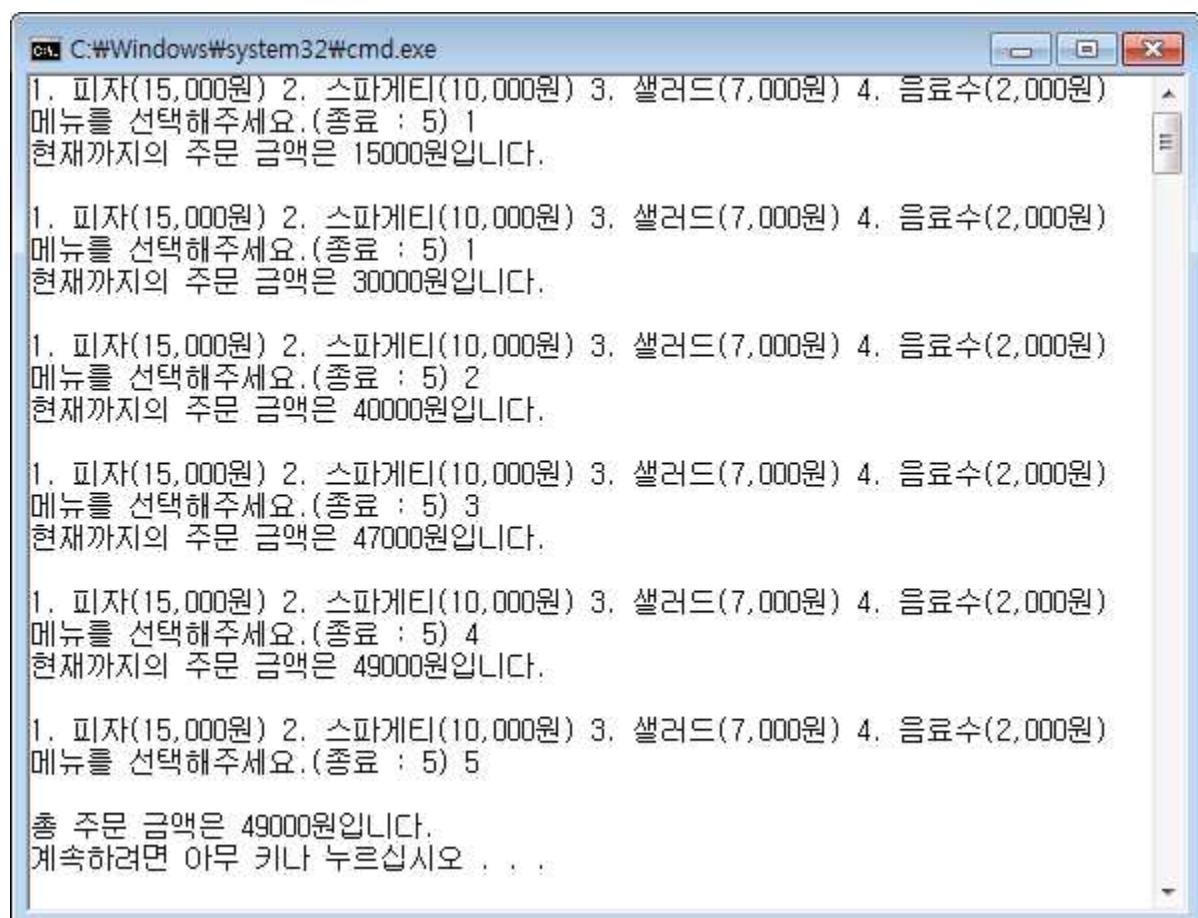
1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000원), 5. 종료

함수 원형은 다음과 같다.

```
int SelectMenu();
```

파라미터) 없음

리턴 값) 1~4를 선택하면 선택한 메뉴의 가격, 5를 선택하면 -1



[J02] 최댓값 리턴하는 함수 만들기

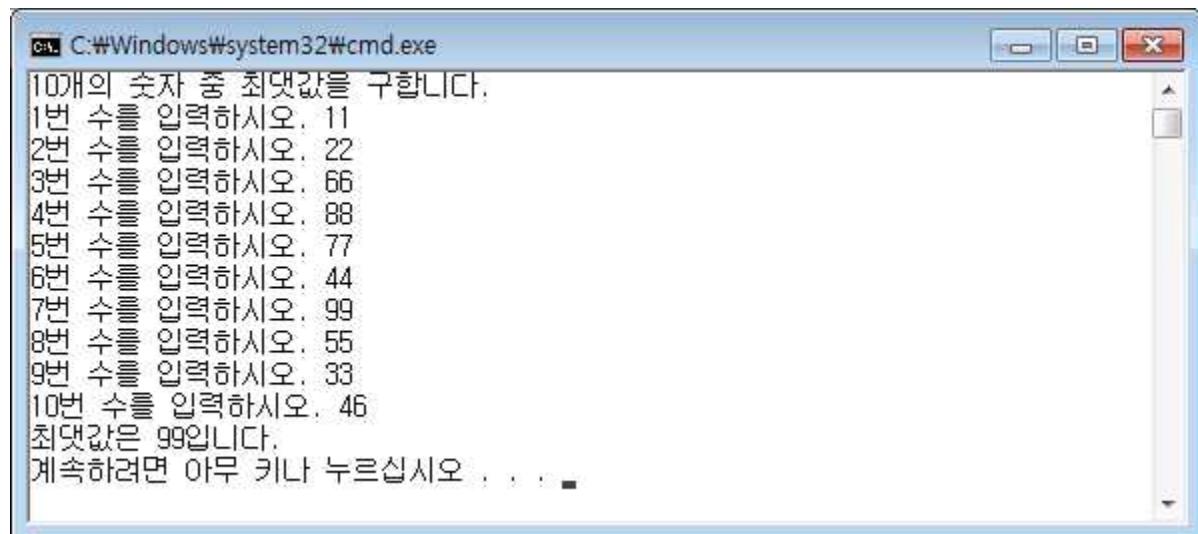
숫자 10개를 입력받아 이 중 최댓값을 찾아서 리턴하는 함수 MaxOfTen()을 만들고, 이 함수를 이용하여 10개의 숫자 중 최댓값을 출력하라.

함수 원형은 다음과 같다.

```
int MaxOfTen(void);
```

파라미터) 없음

리턴 값) 최댓값



[J03] 임의의 숫자를 만들어 구간을 리턴하는 함수 만들기

1부터 100사이의 임의의 숫자를 만들어서 대(70 이상), 중(40이상~70미만), 소(40미만) 셋 중에 하나를 판정하여 결과를 리턴하는 함수 GetRandom()을 만들어라. 그리고 이 함수를 이용해서 임의의 숫자 10개에 대해 대, 중, 소가 각각 몇 번씩 포함되어 있는지 개수를 출력하라.

함수 원형은 다음과 같다.

```
int GetRandom(void);
```

파라미터) 없음

리턴 값) 임의로 만들어낸 숫자가 속하는 구간번호 0.대(70 이상), 1.중(40이상~70미만), 2.소(40미만)

```
C:\Windows\system32\cmd.exe
10개의 숫자를 생성합니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 87입니다.
생성된 임의의 숫자는 45입니다.
생성된 임의의 숫자는 13입니다.
생성된 임의의 숫자는 84입니다.
생성된 임의의 숫자는 57입니다.
생성된 임의의 숫자는 35입니다.
생성된 임의의 숫자는 87입니다.
생성된 임의의 숫자는 62입니다.

1. 대 (70 이상) : 3회 생성
2. 중 (40 이상) : 4회 생성
3. 소 (40 미만) : 3회 생성
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
10개의 숫자를 생성합니다.
생성된 임의의 숫자는 27입니다.
생성된 임의의 숫자는 98입니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 54입니다.
생성된 임의의 숫자는 97입니다.
생성된 임의의 숫자는 1입니다.
생성된 임의의 숫자는 68입니다.
생성된 임의의 숫자는 30입니다.
생성된 임의의 숫자는 41입니다.
생성된 임의의 숫자는 94입니다.

1. 대 (70 이상) : 3회 생성
2. 중 (40 이상) : 3회 생성
3. 소 (40 미만) : 4회 생성
계속하려면 아무 키나 누르십시오 . . .
```

[J04] 자판기에서 선택한 음료 가격을 리턴하는 함수 만들기

자판기의 메뉴를 보여주고 선택하게 하여 선택된 음료의 가격을 리턴하는 함수 SelectCan()을 만들어라.
그리고 이 함수를 이용해서 자판기에서 음료를 반복해서 선택하게 하여 총 음료의 개수와 가격을 출력하라.
자판기의 음료 종류와 가격은 다음과 같다.

- 1.콜라(700원) 2.원두커피(300원) 3.레몬주스(1000원) 4.홍차(500원) 5.코코아(600원)

함수 원형은 다음과 같다.

```
int SelectCan(void);  
파라미터) 없음  
리턴 값) 선택한 음료의 가격
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays a menu selection loop. It lists five items: 1.콜라(700원), 2.원두커피(300원), 3.레몬주스(1000원), 4.홍차(500원), and 5.코코아(600원). It then asks '메뉴를 선택해주세요 : ' followed by a number input. It then asks '더 필요하십니까?(Y/N)' followed by a response of 'Y'. This loop repeats three more times, each time selecting item 1 (콜라). Finally, it asks '5개의 음료를 선택하여 총 3200원입니다.' and ends with '계속하려면 아무 키나 누르십시오 . . .'. The window has standard Windows UI elements like minimize, maximize, and close buttons.

[Step K] 파라미터와 리턴 값이 모두 있는 함수 만들기

프로그램을 만들 때에 가장 일반적으로 사용하는 함수의 형태는 파라미터와 리턴 값이 모두 있는 함수이다. 함수를 호출할 때에 함수 실행에 필요한 값을 파라미터로 전달하고, 실행이 끝나면 그 결과로 얻어내야 하는 값을 리턴받는 것이다. 이번 단계에서는 이런 형식의 함수를 제작하는 연습을 하도록 한다.

앞의 [Step J]에서 만들어 본 GetScore() 함수를 어떤 과목의 점수를 입력받아야 하는지를 명시해서 호출하도록 변형해보도록 하자. 즉, 파라미터로 과목의 이름을 전달하는 것이다. 먼저 함수의 원형을 다음과 같이 파라미터로 문자의 배열 즉 문자열을 전달하도록 바꾸어야 한다.

```
int GetScore( char[] );           // 함수 원형 선언
```

함수의 원형이 바뀌었으므로 메인함수에서 이 함수를 호출하는 구문도 다음과 같이 바꿔어야 한다.

GetScore() 함수의 파라미터에 과목의 이름을 문자열로 넣어주면 된다.

```
int jumsu[3];           // 국어, 영어, 수학 점수

jumsu[0] = GetScore("국어");
jumsu[1] = GetScore("영어");
jumsu[2] = GetScore("수학");
```

마지막으로 변경된 함수의 구현 내용은 다음과 같이 바꿔주면 된다. 진하게 표시한 부분이 이전 소스에서 변경된 부분이다.

```
int GetScore(char classname[])
{
    int jumsu;
    while(1) {
        printf("%s 점수를 입력하시오. ", classname);      // 과목이름을 표시해준다.
        scanf("%d", &jumsu);
        if (jumsu < 0 || jumsu > 100)
            printf("잘못 입력하였습니다. \n");
        else
            return jumsu;          // 올바른 범위의 숫자이므로 리턴한다.
    }
}
```

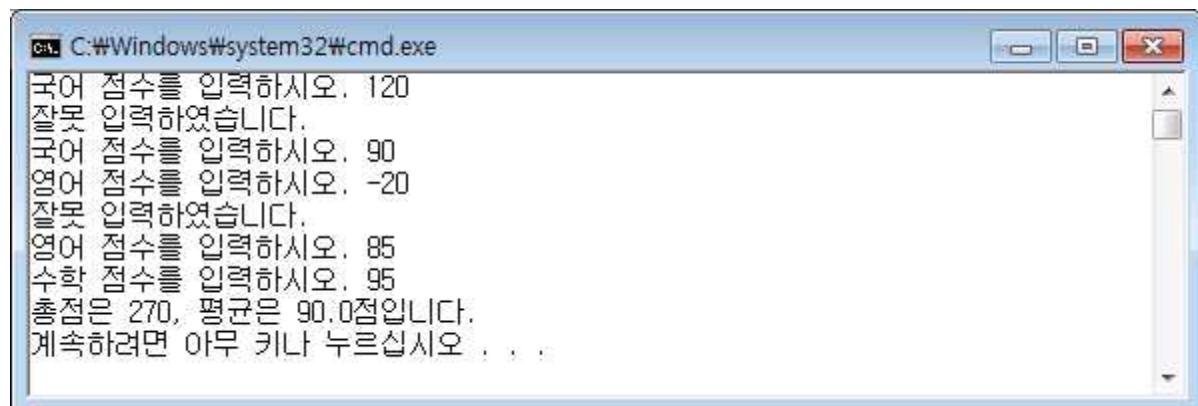
완성된 최종 프로그램 소스와 실행화면은 다음과 같다.

```
/*
ex_K.c
*/
#include <stdio.h>
int GetScore( char[] ); // 함수 원형 선언

void main()
{
    int jumsu[3]; // 국어, 영어, 수학 점수
    int sum; // 총점
    float average; // 평균
    int i;

    jumsu[0] = GetScore("국어");
    jumsu[1] = GetScore("영어");
    jumsu[2] = GetScore("수학");
    sum = 0;
    for (i=0; i<3 ; i++ )
        sum = sum + jumsu[i];
    average = sum / 3.0;
    printf("총점은 %d, 평균은 %.1f점입니다.\n", sum, average);
}

int GetScore(char classname[])
{
    int jumsu;
    while(1) {
        printf("%s 점수를 입력하시오. ", classname);
        scanf("%d", &jumsu);
        if (jumsu < 0 || jumsu > 100)
            printf("잘못 입력하였습니다. \n");
        else
            return jumsu; // 올바른 범위의 숫자이므로 리턴한다.
    }
}
```



○ 실습 문제

[K01] 나이 계산 및 연령대 판정

[G01] 문제를 참고하여 최대 100명까지 사람들의 2012보다 큰 년도가 입력되기 전까지 태어난 년도를 입력받도록 하라. 입력이 끝나면 AskAge() 함수를 사용해서 지금까지 입력된 사람들의 나이를 모두 출력하고, 연령대 별로 각각 몇 명인지 출력하라.

AskAge()에서는 태어난 년도를 입력하면 나이를 출력한 후, 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 연령대 번호를 리턴한다.

단, 나이 = $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아, 7세 이상 ~ 13세미만 : 어린이, 13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년, 30세 이상 ~ 60세 미만 : 중년, 60세 이상 : 노년

함수의 원형은 다음과 같다.

```
int AskAge(int birthyear);
```

파라미터) birthyear : 태어난 년도

리턴값) 계산한 나이에 따른 연령대 번호 (0.유아, 1.어린이, 2.청소년, 3.청년, 4.중년, 5.노년)

```
C:\Windows\system32\cmd.exe
1번째 사람의 태어난 년도를 입력하시오. 1999
나이는 14 입니다.
2번째 사람의 태어난 년도를 입력하시오. 2005
나이는 8 입니다.
3번째 사람의 태어난 년도를 입력하시오. 1940
나이는 73 입니다.
4번째 사람의 태어난 년도를 입력하시오. 1970
나이는 43 입니다.
5번째 사람의 태어난 년도를 입력하시오. 1966
나이는 47 입니다.
6번째 사람의 태어난 년도를 입력하시오. 1988
나이는 25 입니다.
7번째 사람의 태어난 년도를 입력하시오. 2010
나이는 3 입니다.
8번째 사람의 태어난 년도를 입력하시오. 1979
나이는 34 입니다.
9번째 사람의 태어난 년도를 입력하시오. 1990
나이는 23 입니다.
10번째 사람의 태어난 년도를 입력하시오. 1981
나이는 32 입니다.
11번째 사람의 태어난 년도를 입력하시오. 2008
나이는 5 입니다.
12번째 사람의 태어난 년도를 입력하시오. 2222
유아는 2명 입니다.
어린이는 1명 입니다.
청소년은 1명 입니다.
청년은 2명 입니다.
중년은 4명 입니다.
노년은 1명 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[K02] 심사점수 계산

심사점수를 10개를 입력받아 배열에 저장한 후, 이 배열을 파라미터로 하여 가장 큰 점수를 구하는 Max() 와 가장 작은 점수를 구하는 Min()을 사용하여 10개의 점수 중 최대점수와 최소점수를 제외한 8개의 점수에 대한 평균을 계산하여 출력하라.

함수의 원형은 다음과 같다.

```
float Max(float num[], int size);
```

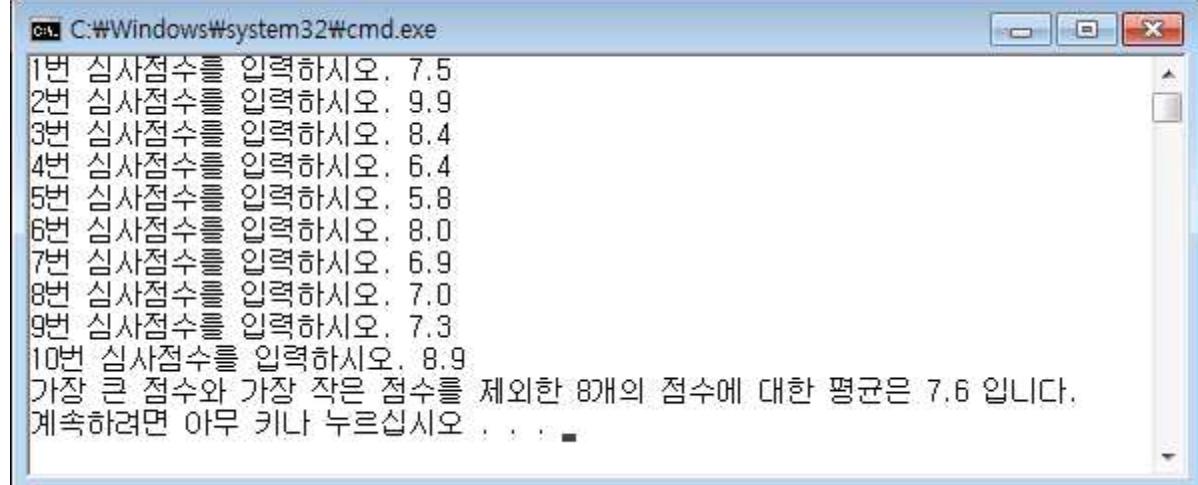
파라미터) num[] : 숫자 배열, size : 배열의 갯수

리턴값) 숫자 배열에서 가장 큰 값

```
float Min(float num[], int size);
```

파라미터) num[] : 숫자 배열, size : 배열의 갯수

리턴값) 숫자 배열에서 가장 작은 값



[K03] 물의 온도 구간 개수 판정

물의 온도를 10회 입력받은 후, 이 물이 각각 어느 정도의 온수인지 AskWater()를 통해 판정하여 그 결과를 출력하라. 출력할 내용은 입력된 10개의 온도 값, 냉수 입력 횟수, 미온수 입력 횟수, 온수 입력 횟수, 끓는 물 입력 횟수를 각각 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

0도 ~ 25도 미만 : 냉수

25도 ~ 40도 미만 : 미온수

40도 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

함수의 원형은 다음과 같다.

int AskWater(float degree);

파라미터) degree: 온도

리턴 값) 온도 판정 번호 (0.냉수, 1.미온수, 2.온수, 3.끓는 물)

The screenshot shows a Windows command prompt window titled 'cmd' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1번째 물의 온도를 입력하시오. 15.5
2번째 물의 온도를 입력하시오. 20.5
3번째 물의 온도를 입력하시오. 27.7
4번째 물의 온도를 입력하시오. 35.0
5번째 물의 온도를 입력하시오. 40.5
6번째 물의 온도를 입력하시오. 66.7
7번째 물의 온도를 입력하시오. 90.1
8번째 물의 온도를 입력하시오. 100.0
9번째 물의 온도를 입력하시오. 77.7
10번째 물의 온도를 입력하시오. 28.0
1번 물의 온도는 15.5도 입니다.
2번 물의 온도는 20.5도 입니다.
3번 물의 온도는 27.7도 입니다.
4번 물의 온도는 35.0도 입니다.
5번 물의 온도는 40.5도 입니다.
6번 물의 온도는 66.7도 입니다.
7번 물의 온도는 90.1도 입니다.
8번 물의 온도는 100.0도 입니다.
9번 물의 온도는 77.7도 입니다.
10번 물의 온도는 28.0도 입니다.
냉수 입력 횟수 2 입니다.
미온수 입력 횟수는 3 입니다.
온수 입력 횟수는 3 입니다.
끓는 물 입력 횟수는 2 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[K04] 연중 날짜 계산 함수를 이용한 날짜 간격 세기

날짜 2개를 입력받은 후, 이 2개의 날짜 간격은 며칠인지 계산하여 출력하라. 단, 월과 일로 파라미터로 넘기면 이 날짜가 1년 중 몇 번째 날에 해당되는지 리턴하는 함수 CalcDate() 함수를 만들어 사용하라. 이 함수에서 매 월의 날 수 계산 시 다음과 같이 매월의 날 수를 배열로 만들어 이를 이용하여 계산하라.

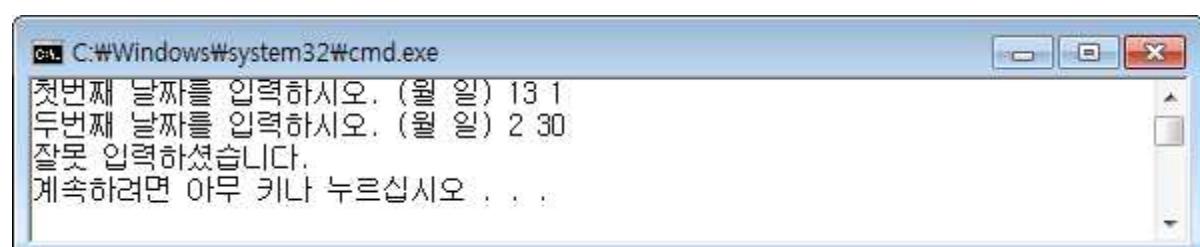
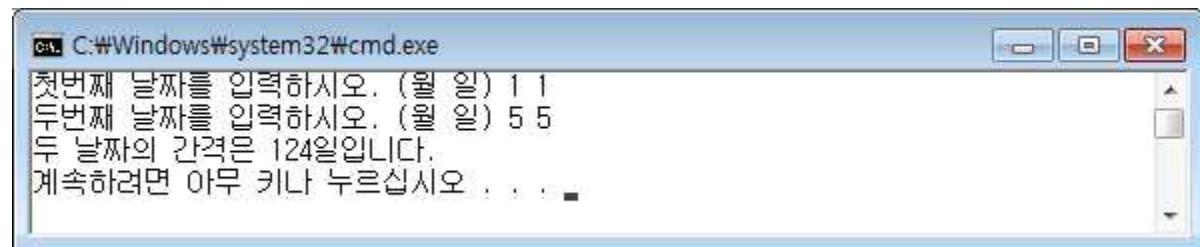
```
int monthdays[12] = {31,28,31,30,31,30,31,31,30,31,30,31}; // 1~12월의 날 수
```

함수의 원형은 다음과 같다.

```
int CalcDate(int month, int day);
```

파라미터) month : 월, day : 일

리턴값) 1년 중 해당 날짜가 몇 번째 날인지의 결과 값 (1~365)



[K05] 주차 관리 시스템

주차장에서 차량들의 주차 관리 시스템을 만들어라. 차량마다 주차를 시작한 시간을 시와 분으로 입력받고, 주차를 종료한 시간을 시와 분으로 입력받은 후, 이를 CalcParking() 함수에 파라미터로 넘겨 주차요금을 리턴받도록 하라. 차량이 더 있는지 물어서 더 이상 차량이 없을 때까지 반복해서 요금을 계산하되 반복이 끝나면 지금까지 계산한 차량의 수량과 총 주차요금을 화면에 출력하라. 주차요금은 10분당 500원으로 한다.

함수의 원형은 다음과 같다.

```
int CalcParking(int start_h, int start_m, int end_h, int end_m);
```

파라미터) start_h : 주차시작 시, start_m : 주차시작 분, end_h : 주차종료 시, end_m : 주차종료 분

리턴값) 주차시작 시간(시, 분)부터 종료 시간(시, 분)까지의 주차요금(원)



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1번 차량 주차 시작 시각 (시 분) : 10 30
1번 차량 주차 종료 시각 (시 분) : 11 15
주차요금 : 2500원
더 입력하시겠습니까?(Y/N) Y
2번 차량 주차 시작 시각 (시 분) : 9 10
2번 차량 주차 종료 시각 (시 분) : 15 10
주차요금 : 18000원
더 입력하시겠습니까?(Y/N) Y
3번 차량 주차 시작 시각 (시 분) : 12 10
3번 차량 주차 종료 시각 (시 분) : 14 55
주차요금 : 8500원
더 입력하시겠습니까?(Y/N) Y
4번 차량 주차 시작 시각 (시 분) : 11 00
4번 차량 주차 종료 시각 (시 분) : 11 05
주차요금 : 500원
더 입력하시겠습니까?(Y/N) N
주차차량 4대의 총 주차 요금은 29500원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[Step L] 재귀 호출 함수 만들기

이번 단계는 영역 3의 마지막 단계로 재귀 호출을 이용하는 함수를 제작하는 것이다. 재귀 호출이란 함수의 구현 내용 안에서 함수 자신을 다시 호출하는 것을 말한다. 즉 재귀 호출 함수는 함수가 자기 자신을 다시 호출하도록 만들어진 함수이다. 그렇다면 왜 이런 방식이 필요할까?

함수를 제작하다보면 함수의 구현 내용이 함수 자신을 사용하도록 정의되는 경우가 있다. 예를 들어 계승(factorial)을 구하는 경우를 생각해보자. 계승이란 1부터 자기 자신의 수까지를 모두 곱하는 것이다. 즉 5의 계승은 $1*2*3*4*5$ 인 120이다. 먼저 반복문을 사용해서 계승을 구하는 함수를 만들어보면 다음과 같다. 구현내용에서 보는 바와 같이 1부터 구하려는 수까지 모두 곱해나가면 되는 것이다. 아래 구문을 쉽게 이해할 수 있을 것이다.

```
int FactorialNoRecur( int n )
{
    int result;
    int i;
    result = 1;
    for (i=1;i<=n;i++)
        result = result * i;           // 1부터 파라미터로 넘어온 n까지를 곱한다.

    return result;
}
```

이 함수를 사용하는 프로그램의 예를 다음과 같이 만들어보았다.

```
/*
ex_L.c
*/
#include <stdio.h>
int FactorialNoRecur( int );           // 함수 원형 선언

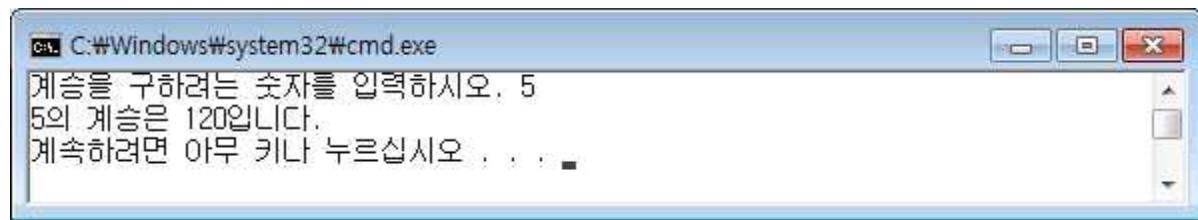
void main()
{
    int number;

    printf("계승을 구하려는 숫자를 입력하시오. ");
    scanf("%d", &number);
    printf("%d의 계승은 %d입니다.\n", number, FactorialNoRecur(number));
}
```

C Workbook

```
int FactorialNoRecur( int n )
{
    int result;
    int i;
    result = 1;
    for (i=1;i<=n;i++)
        result = result * i;           // 1부터 파라미터로 넘어온 n까지를 곱한다.

    return result;
}
```



그리면 이제 같은 작업을 수행하는 함수를 재귀 호출 함수로 만들어보도록 하자. 재귀 호출 함수는 자신을 불러야 하기 때문에 계승의 원리를 다음과 같이 생각해야 한다.

$$\begin{aligned} n\text{의 계승} &= n * (n-1) * (n-2) * \dots * 2 * 1 \\ &= n * \{(n-1) * (n-2) * \dots * 2 * 1\} \\ &= n * \{(n-1)\text{의 계승}\} \end{aligned}$$

이렇게 해서 계승의 계산 원리 속에 자신보다 하나 적은 수의 계승을 넣을 수 있다. 물론 계승을 무한대로 재귀 호출 할 수 없으며, 기본이 되는 조건이 있어야 한다. 여기에서는 n 의 값이 1인 경우에 더 이상 자기 자신을 호출하지 않도록 해 준다. 즉 다음과 같이 n 의 계승을 재귀 호출의 형태로 정의할 수 있다.

$$\begin{aligned} n\text{의 계승} &= n * \{(n-1)\text{의 계승}\} && (n이 1보다 큰 경우) \\ 또는 &= 1 && (n이 1인 경우) \end{aligned}$$

이렇게 정의한 계승을 함수로 만들어 보면 다음과 같다.

```
int Factorial( int n )
{
    if ( n == 1) return 1;                      // n이 1인 경우
    else return (n * Factorial( n-1 ) );        // n이 1보다 큰 경우
}
```

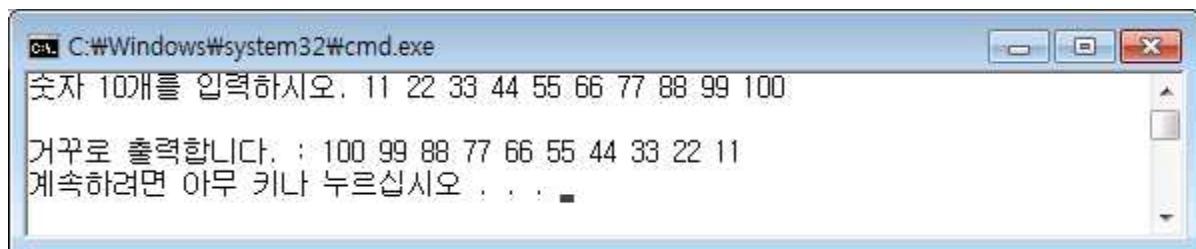
이렇게 재귀 호출 함수를 만들 때에는 반드시 함수 자기 자신을 사용하여 정의할 수 있어야 하며, 또한 더 이상의 재귀 호출이 일어나지 않는 기본 조건을 넣어 주어야 한다.

비슷한 예로 1부터 n까지의 숫자들을 모두 더한 값을 알아내는 작업을 재귀 호출 함수로 만든다면 다음 구문처럼 하면 된다.

```
int SumToOne( int n )
{
    if ( n == 1) return 1;                                // n이 1인 경우
    else return (n + SumToOne( n-1 ) );                  // n이 1보다 큰 경우
}
```

10개의 숫자가 들어 있는 배열을 역순으로 출력하는 재귀 호출 함수는 다음과 같다.

```
void PrintReverse(int number[], int length) // number[] 숫자 배열, length 출력문자수
{
    printf("%d ", number[length-1]);                // 가장 마지막 숫자 출력
    if ( length > 1) PrintReverse(number, length-1); // 출력문자수 1개 감소 후 출력
}
```



○ 실습 문제

[L01] 피보나치 수 구하기

n이 1부터 20까지 증가하는 경우 각각의 피보나치 수 $\text{fibonacci}(n)$ 을 출력하라. 피보나치 수는 다음과 같이 정의한다.

$$\text{fibonacci}(n) = \begin{cases} 1 & (n = 1 \text{ 또는 } n = 2 \text{인 경우}) \\ \text{fibonacci}(n-1) + \text{fibonacci}(n-2) & (n > 2 \text{인 경우}) \end{cases}$$

단, 함수의 원형은 다음과 같이 사용하라.

```
int fibonacci(int n);
```

```
1부터 20까지 피보나치 수는 다음과 같습니다.  
1번째 : 1  
2번째 : 1  
3번째 : 2  
4번째 : 3  
5번째 : 5  
6번째 : 8  
7번째 : 13  
8번째 : 21  
9번째 : 34  
10번째 : 55  
11번째 : 89  
12번째 : 144  
13번째 : 233  
14번째 : 377  
15번째 : 610  
16번째 : 987  
17번째 : 1597  
18번째 : 2584  
19번째 : 4181  
20번째 : 6765  
계속하려면 아무 키나 누르십시오 . . .
```

[L02] 2의 제곱수 구하기

반복해서 임의의 숫자 n 을 입력받은 후 2^n 을 계산하여 출력하되, 재귀함수를 이용하여 계산하라. 이 때 사용할 재귀함수 poweroftwo()의 정의는 다음과 같다.

$$\text{poweroftwo}(n) = \begin{cases} 1 & (n = 0 \text{인 경우}) \\ 2 \times \text{poweroftwo}(n-1) & (n > 0 \text{인 경우}) \end{cases}$$

단, 함수의 원형은 다음과 같이 사용하라.

```
int poweroftwo(int n);
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays a series of recursive calls to the poweroftwo function, starting with an input of 2 and showing the result of each step:

```

숫자를 입력하시오. (0.종료) : 2
2의 2승은 : 4
숫자를 입력하시오. (0.종료) : 4
2의 4승은 : 16
숫자를 입력하시오. (0.종료) : 8
2의 8승은 : 256
숫자를 입력하시오. (0.종료) : 10
2의 10승은 : 1024
숫자를 입력하시오. (0.종료) : 16
2의 16승은 : 65536
숫자를 입력하시오. (0.종료) : 20
2의 20승은 : 1048576
숫자를 입력하시오. (0.종료) : 24
2의 24승은 : 16777216
숫자를 입력하시오. (0.종료) : 5
2의 5승은 : 32
숫자를 입력하시오. (0.종료) : 7
2의 7승은 : 128
숫자를 입력하시오. (0.종료) : 0
계속하려면 아무 키나 누르십시오 . . .

```

[L03] Ackermann 수 구하기

Ackermann's Function A 는 다음과 같이 정의된다. $A(i, j)$ 를 재귀 호출 함수로 만들고, 이 함수를 이용하여 $A(0,0)$ 에서 $A(3,3)$ 의 값을 구하라.

Ackermann's Function A

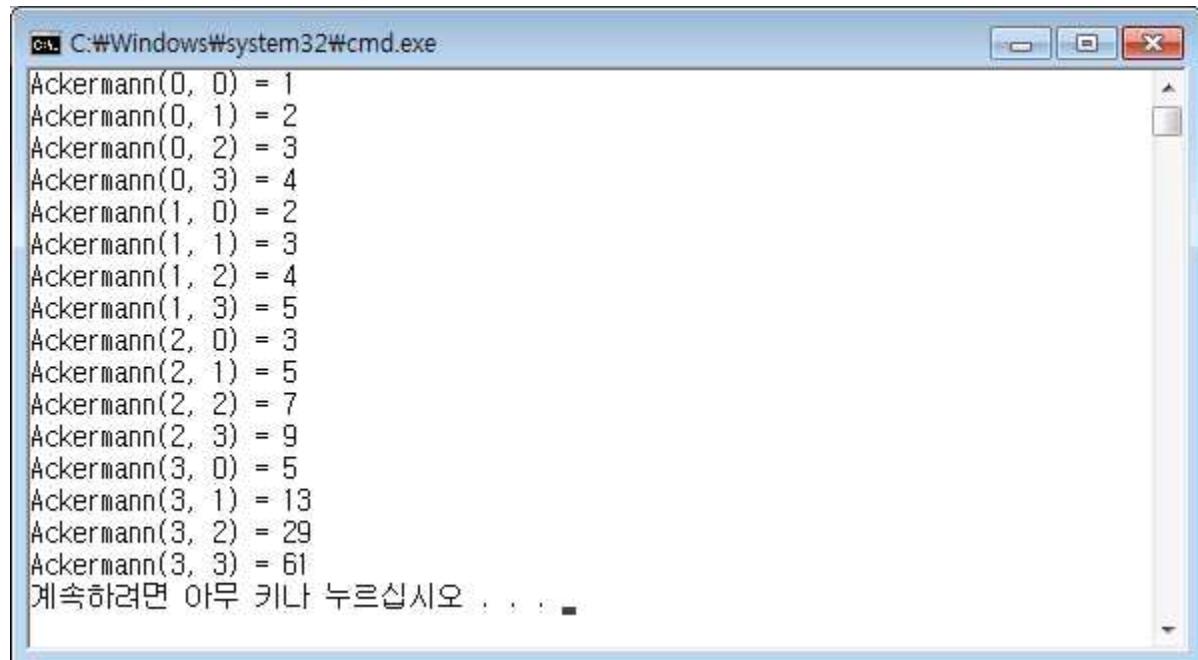
$$A(0, j) = j+1 \quad i = 0 \text{ 이고, } j \geq 0 \text{ 인 경우}$$

$$A(i, 0) = A(i-1, 1) \quad i > 0 \text{ 이고 } j = 0 \text{ 인 경우}$$

$$A(i, j) = A(i-1, A(i, j-1)) \quad i \text{ 와 } j \text{ 모두 } 0\text{보다 큰 경우}$$

단, 함수의 원형은 다음과 같이 사용하라.

```
int Ackermann(int i, int j);
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the output of a recursive Ackermann function. The output consists of multiple lines of text, each showing a call to the Ackermann function with its result. The calls start from Ackermann(0, 0) and follow the recursive pattern to higher values. The last visible call is Ackermann(3, 3), which results in a value of 61. Below this, there is a message in Korean: '계속하려면 아무 키나 누르십시오 . . .' (Press any key to continue . . .).

```
Ackermann(0, 0) = 1
Ackermann(0, 1) = 2
Ackermann(0, 2) = 3
Ackermann(0, 3) = 4
Ackermann(1, 0) = 2
Ackermann(1, 1) = 3
Ackermann(1, 2) = 4
Ackermann(1, 3) = 5
Ackermann(2, 0) = 3
Ackermann(2, 1) = 5
Ackermann(2, 2) = 7
Ackermann(2, 3) = 9
Ackermann(3, 0) = 5
Ackermann(3, 1) = 13
Ackermann(3, 2) = 29
Ackermann(3, 3) = 61
계속하려면 아무 키나 누르십시오 . . .
```

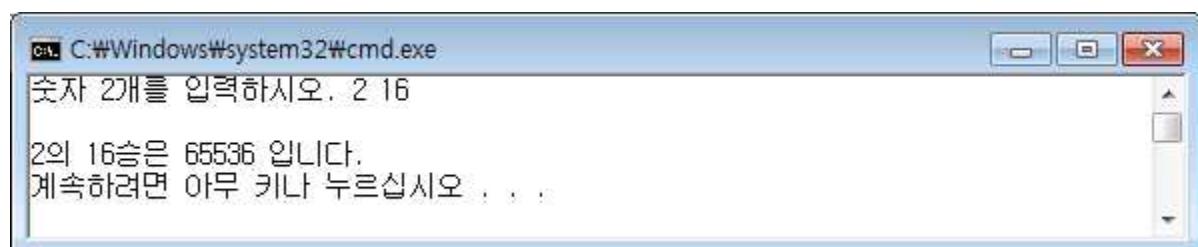
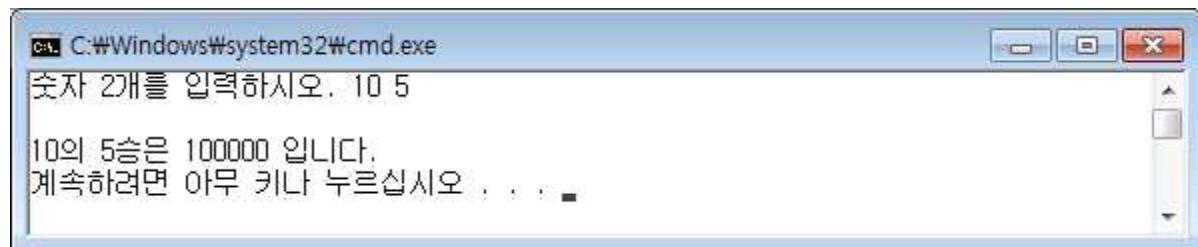
[L04] pow() 함수 만들기

n^a 를 계산할 수 있는 <math.h>내의 pow() 함수와 같은 일을 하는 power() 함수를 재귀 호출을 이용하여 만들어라. 그리고 숫자 2개(num1, num2)를 입력받아 $num1^{num2}$ 를 계산하라.
단, power() 함수는 다음과 같이 정의된다.

$$power(n, a) = \begin{cases} 1 & (n=0 \text{인 경우}) \\ n & (n=1 \text{인 경우}) \\ power(n, \frac{a}{2}) \times power(n, \frac{a}{2}) & (n > 1 \text{이고 짝수인 경우}) \\ power(n, \frac{a}{2}) \times power(n, \frac{a}{2}) \times n & (n > 1 \text{이고 홀수인 경우}) \end{cases}$$

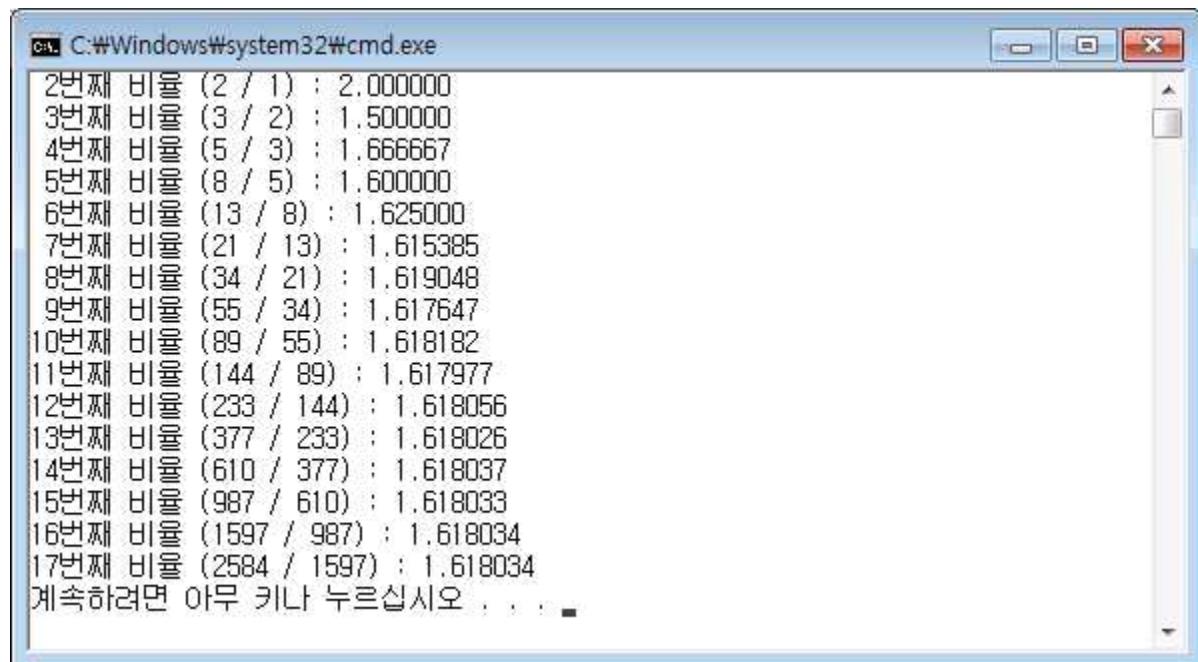
단, 함수의 원형은 다음과 같이 사용하라.

```
int power( int n, int a );
```



[L05] 피보나치 수열로 황금비율 구하기

문제 [L01]에서 제작한 피보나치 함수 fibonacci(n)를 사용하여 황금비율을 찾아내라. n 번째 황금비율이란 피보나치 수열의 연속적인 2개의 숫자의 비율($n+1$ 번째수 / n 번째 수)로 정한다.
단, 계산한 비율이 직전의 비율에 비해 그 차이가 백만분의 1보다 작게 되면 멈추도록 하라.



The screenshot shows a Windows Command Prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays a series of calculations for the golden ratio, which is the ratio of two consecutive Fibonacci numbers. The output is as follows:

```
2번째 비율 (2 / 1) : 2.000000
3번째 비율 (3 / 2) : 1.500000
4번째 비율 (5 / 3) : 1.6666667
5번째 비율 (8 / 5) : 1.600000
6번째 비율 (13 / 8) : 1.625000
7번째 비율 (21 / 13) : 1.615385
8번째 비율 (34 / 21) : 1.619048
9번째 비율 (55 / 34) : 1.617647
10번째 비율 (89 / 55) : 1.618182
11번째 비율 (144 / 89) : 1.617977
12번째 비율 (233 / 144) : 1.618056
13번째 비율 (377 / 233) : 1.618026
14번째 비율 (610 / 377) : 1.618037
15번째 비율 (987 / 610) : 1.618033
16번째 비율 (1597 / 987) : 1.618034
17번째 비율 (2584 / 1597) : 1.618034
계속하려면 아무 키나 누르십시오 . . .
```

영역 4 : 구조적 데이터 다루기

영역 4에서는 프로그램을 제작할 때에 반드시 사용하게 되는 데이터들을 여러 개 합쳐서 하나의 구조로 만들어 활용하는 방법을 연습한다. 구체적으로는 구조체, 포인터, 구조체 포인터, 포인터 배열, 파일 등을 다루게 된다.

구조체는 논리적으로 연관되어 있는 서로 다른 타입의 변수를 2개 이상 한 묶음으로 만들어 사용하는 것이다. 포인터는 메모리에 보관되고 있는 데이터를 다룰 때에 현재의 보관 주소로 다루는 방식이다. 구조체처럼 여러 데이터가 한 묶음으로 다뤄지는 경우에는 이를 포인터로 접근하도록 하는 것이 편리하다. 또한 여러 개의 구조체 데이터를 다룰 때에는 이들을 포인터 배열로 다루게 되면 효과적인 프로그램 제작이 가능하다. 또한 다양한 데이터를 파일에 저장하고, 다시 파일로부터 읽는 방법을 연습하도록 한다.

영역 4는 다음과 같이 4개의 단계로 구성된다.

[Step M] 구조체 사용하기

[Step N] 포인터 사용하기

[Step O] 구조체와 배열 함께 사용하기

[Step P] 파일 사용하기

[Step M] 구조체 사용하기

구조체를 정의하면 다음과 같다.

- ① 구조체는 자료형이 서로 다른 자료들의 집합이다.
- ② 의미상으로 연관성이 있는 여러 가지의 자료들을 하나로 묶어 마치 하나의 변수를 사용하듯 프로그래밍을 할 수 있다.
- ③ 배열이 동일한 자료들의 집합이라면, 구조체는 기본자료형이나 배열, 포인터 등의 여러 자료형을 동시에 가질 수 있다.

예를 들어 다음과 같은 성적처리 프로그램을 작성한다고 가정하자.

```
char name[30]; // 사람 이름
int kor; // 국어 성적
int eng; // 영어 성적
int math; // 수학 성적
int total; // 총점
float ave; // 평균
```

위와 같이 한 사람에 필요한 변수들을 정하고, 이번에는 이 변수들을 하나의 구조체로 묶으면 다음과 같다.

```
struct man {
    char name[30];
    int kor;
    int eng;
    int math;
    int total;
    float ave;
}
```

그리고 이 구조체를 따르는 구조체 변수는 다음과 같이 선언한다.

```
struct man student1, student2; // 학생 2명의 점수 관리를 위한 구조체 변수
```

구조체 변수를 다룰 때에는 구조체 변수이름과 그 안에 들어있는 구조체 멤버의 이름을 점(.)으로 연결해 주면 된다.

```
student1.kor = 100;      // student1 학생의 국어점수를 100으로 할당
student1.total = student1.kor + student1.eng + student1.math; // 총점 계산
```

다음 프로그램은 한 사람에 대한 주소 정보를 입력받아 다시 출력하는 것으로 구조체의 선언과 사용 방법을 보여주고 있다.

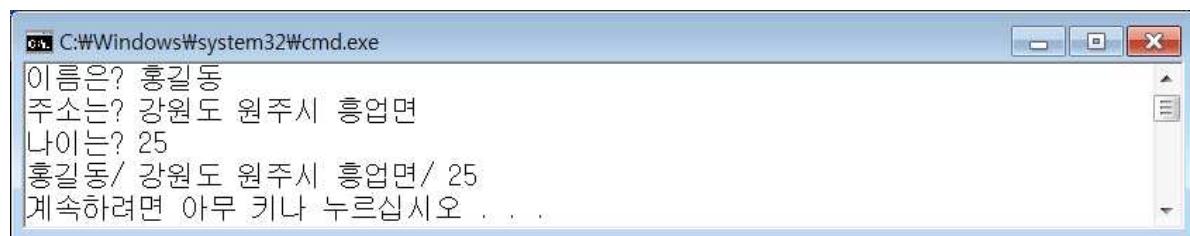
```
// 예제 ex_M_1.c
#include <stdio.h>

struct person{
    char name[20];
    char address[80];
    int age;
};

void main()
{
    struct person employee;

    printf("이름은? ");
    gets(employee.name);
    printf("주소는? ");
    gets(employee.address);
    printf("나이는? ");
    scanf("%d", &employee.age);

    printf("%s/ %s/ %d\n", employee.name, employee.address, employee.age);
}
```



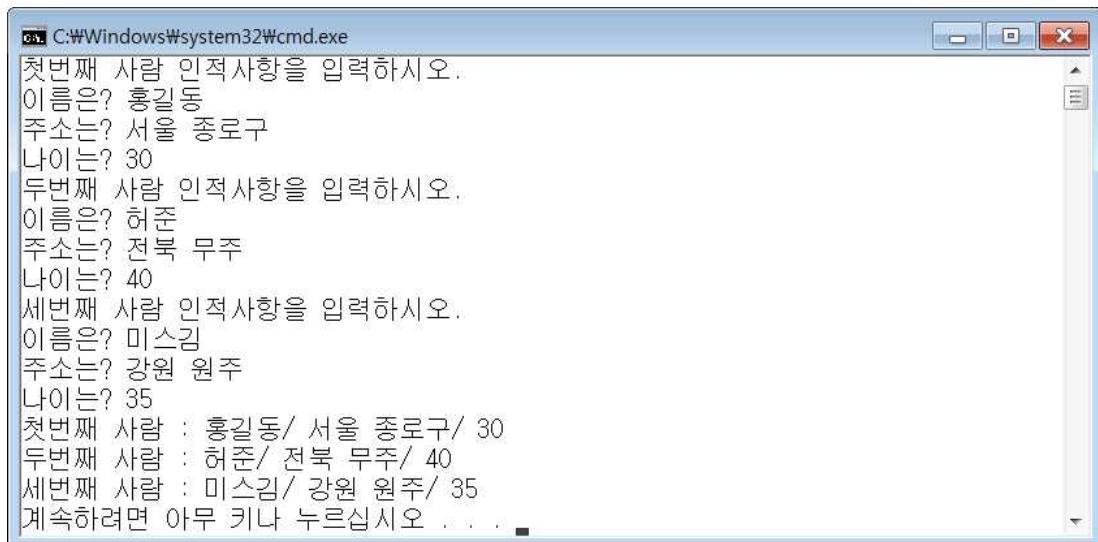
○ 실습 문제

[M01] 인적사항 데이터 입력

다음 구조체를 이용하여 3명의 데이터를 입력받아 이를 한꺼번에 출력하라.

```
struct person{  
    char name[20];  
    char address[80];  
    int age;  
};
```

단, 3명의 데이터를 입력받기 위해 구조체 변수 3개를 임의로 선언하여 사용하라.



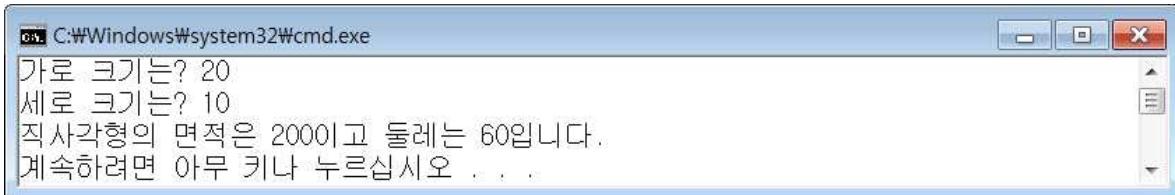
[M02] 사각형 면적 계산

다음 구조체를 이용하여 직사각형의 가로, 세로 값을 입력받아 둘레와 면적을 계산하여 출력하라.

```
struct rectangle{
    int width;      // 세로
    int height;     // 가로
};
```

이 때 직사각형의 면적과 둘레를 계산하는 함수를 별도로 만들어 사용하라.

```
int calc_area(struct rectangle rect);           // 직사각형 면적 구하는 함수
int calc_boundary(struct rectangle rect);        // 직사각형 둘레 구하는 함수
```



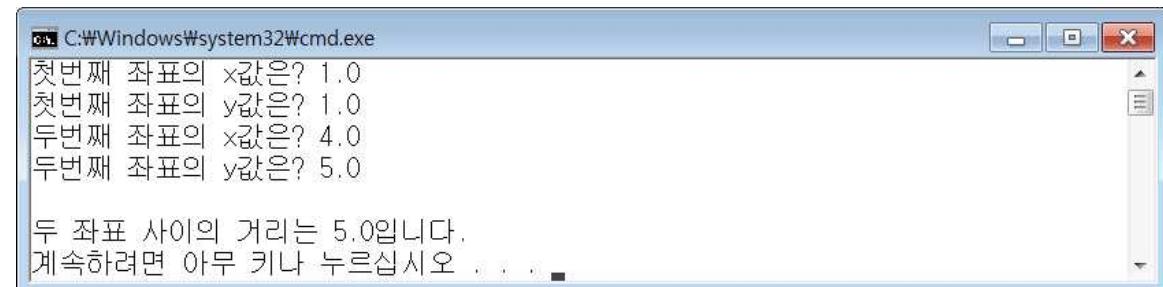
[M03] 좌표 거리 계산

다음 구조체를 이용하여 2개의 좌표 데이터를 입력받아 이를 출력하고, 두 좌표 사이의 거리를 출력하라.

```
struct point{
    float x; // x좌표 값
    float y; // y좌표 값
};
```

단, 두 좌표 사이의 거리는 별도의 함수를 만들어 사용하되, 계산 방법은 $(x\text{값의 차이})^2 + (y\text{값의 차이})^2$ 의 제곱근이다. 계산할 때에 `sqrt()`와 `pow()`함수를 사용하라.

```
float calc_distance(struct point p1, struct point p2);           // 두 점의 거리 구하는
함수
```



[M04] 사각형 크기 비교

다음 구조체를 이용하여 직사각형 2개의 좌상점의 좌표값(x, y)와 우하점의 좌표값(x, y)을 입력받아 면적을 계산하고 어떤 직사각형이 큰지 출력하라. 이 때 직사각형의 면적을 계산하는 함수를 만들어 사용하라.

```
struct rectangle{  
    int x1, y1;      // 좌상점 좌표값 (왼쪽 위 모서리)  
    int x2, y2;      // 우하점 좌표값 (오른쪽 아래 모서리)  
};  
int calc_area(struct rectangle rect);           // 직사각형 면적 구하는 함수
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following text:

```
첫번째 직사각형의 좌상점 좌표값(x, y)은? 1 3  
첫번째 직사각형의 우하점 좌표값(x, y)은? 4 1  
두번째 직사각형의 좌상점 좌표값(x, y)은? 2 5  
두번째 직사각형의 우하점 좌표값(x, y)은? 5 1  
  
첫번째 직사각형의 크기 : 6  
두번째 직사각형의 크기 : 12  
두번째 직사각형의 면적이 더 큽니다.  
계속하려면 아무 키나 누르십시오 . . .
```

[M05] 직사각형 겹침 검사

다음 구조체를 이용하여 직사각형 2개의 좌상점의 좌표값(x, y)과 우하점의 좌표값(x, y)을 입력받은 후에 2개의 직사각형이 평면좌표 상에서 겹치는 부분이 있는지의 여부를 출력하라.

```
struct rectangle{
    int x1, y1;      // 좌상점 좌표값 (왼쪽 위 모서리)
    int x2, y2;      // 우하점 좌표값 (오른쪽 아래 모서리)
};
```

단, 2개의 직사각형이 겹치는지를 검사하는 다음 함수를 별도로 만들어 사용하라.

```
int check_overlap(struct rectangle r1, struct rectangle r2); // 겹치면 1리턴, 아니면 0리턴
```

```
C:\Windows\system32\cmd.exe
첫번째 직사각형의 좌상점 좌표값(x, y)은? 1 3
첫번째 직사각형의 우하점 좌표값(x, y)은? 2 2
두번째 직사각형의 좌상점 좌표값(x, y)은? 3 5
두번째 직사각형의 우하점 좌표값(x, y)은? 4 2
두 직사각형은 겹치지 않습니다.
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
첫번째 직사각형의 좌상점 좌표값(x, y)은? 1 4
첫번째 직사각형의 우하점 좌표값(x, y)은? 3 2
두번째 직사각형의 좌상점 좌표값(x, y)은? 2 5
두번째 직사각형의 우하점 좌표값(x, y)은? 4 1
두 직사각형은 겹칩니다.
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
첫번째 직사각형의 좌상점 좌표값(x, y)은? 3 3
첫번째 직사각형의 우하점 좌표값(x, y)은? 4 2
두번째 직사각형의 좌상점 좌표값(x, y)은? 1 4
두번째 직사각형의 우하점 좌표값(x, y)은? 5 0
두 직사각형은 겹칩니다.
계속하려면 아무 키나 누르십시오 . . .
```

[M06] 메뉴판 만들기

어떤 식당의 메뉴판이 다음과 같이 구성되어 있을 때, 다음 5개의 메뉴를 5개의 구조체 변수에 저장한 후, 이를 화면에 출력하라.

메뉴번호	메뉴이름	원산지	1인분 가격
1	삼겹살	국내산	9000
2	갈비살	미국산	15000
3	꽃등심	국내산	30000
4	양념갈비	호주산	25000
5	차돌박이	국내산	28000

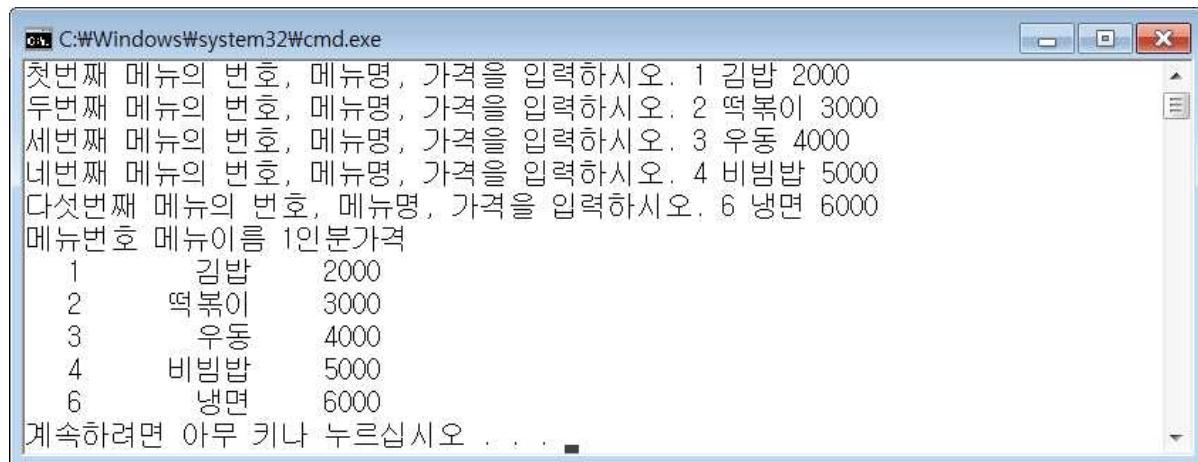
```
struct menu{
    int no;                  // 메뉴번호
    char name[20];           // 메뉴명
    char madein[20];          // 원산지
    int price;                // 가격
}
```



[M07] 사용자 정의형 메뉴판 만들기

어떤 식당의 메뉴판의 구조가 다음과 같이 구성되어 있을 때, 이 메뉴판의 메뉴 정보를 구조체로 선언하고, 5개의 메뉴 정보를 입력받아 이를 5개의 구조체 변수에 저장한 후, 한꺼번에 화면에 출력하라.

```
struct menu{  
    int no;           // 메뉴번호  
    char name[20];    // 메뉴명  
    int price;        // 가격  
}
```



[Step N] 포인터 사용하기

```
int i; //4바이트 //int의 포인터
char ch; //1바이트 //캐릭터의 포인터
float f; //4바이트 //float의 포인터
```

```
int * p1 = %i;
char * p2 = %ch;
char * p3 = %f;
```

이번 단계에서는 포인터를 다루어 보겠다. 포인터란 주소값을 갖도록 하는 특별한 형식의 변수를 말한다. 프로그램에서 선언된 변수들은 각각 특정한 위치와 크기의 메모리 공간을 할당받게 되는데, 변수들이 위치한 메모리 내에서의 위치인 주소값을 알고 있으면 이 값을 통해 해당 변수를 다룰 수 있다. 포인터는 바로 이런 기능을 제공하도록 만들어진 것이다.

포인터를 선언할 때에는 변수 형식 뒤에 '*' 표시를 달아 선언한다. 그리고 포인터를 사용할 때에는 특정 변수의 주소값을 알아내는 '&' 연산자와, 특정 포인터가 가리키는 변수를 찾아내는 '**' 연산자를 사용한다.

```
// 예제 ex_N_1.c
#include <stdio.h>

void main()
{
    int num=100;
    int * ptr; // 정수형 변수를 가리키는 포인터 ptr을 선언한다.

    ptr = &num; // 포인터 ptr에 num변수의 주소값을 할당한다.
    printf("ptr의 값은 %u이며, ptr이 가리키고 있는 변수의 값은 %d이다.\n", ptr, *ptr);
}
```

포인터를 사용하는 주요 이유 중에 하나는 서브 함수를 실행할 때 호출하는 구문(예를 들어 main 함수) 내의 변수 값을 변경하도록 할 수 있다는 것이다. 예를 들어 두 변수의 값을 맞바꾸는 다음과 같은 코드를 생각해보자.

```
void main()
{
    int num1, num2, temp;

    num1 = 100; num2 = 200;
    printf("num1 = %d, num2 = %d\n", num1, num2);

    temp = num1;
    num1 = num2;
    num2 = temp;
    printf("num1 = %d, num2 = %d\n", num1, num2);
}
```

위의 코드에서 숫자 2개의 값을 맞바꾸는 동작을 별도의 함수를 통해 하게 하려면 다음과 같이 포인터를 활용해야 한다.

```
// 예제 ex_N_2.c
#include <stdio.h>
void exchange(int * n1, int * n2);

void main()
{
    int num1, num2, temp;

    num1 = 100; num2 = 200;
    printf("num1 = %d, num2 = %d\n", num1, num2);
    exchange(&num1, &num2);
    printf("num1 = %d, num2 = %d\n", num1, num2);
}

void exchange(int * n1, int * n2)
{
    int temp;
    temp = *n1;
    *n1 = *n2;
    *n2 = temp;
}
```

이번에는 구조체 변수를 가리키는 포인터를 연습하도록 하자. 이전 스텝에서 언급한 바와 같이 함수 호출 시에 파라미터로 구조체 변수의 포인터를 사용하게 되면 호출된 함수에서 호출한 함수에서 선언한 구조체 변수들을 직접 다룰 수 있다.

앞의 예제 ex_M_1.c에서 선언한 구조체를 사용하여 구조체 포인터의 사용 방법을 설명하겠다.

```
struct person{
    char name[20];
    char address[80];
    int age;
};
```

메인 함수에서 구조체 변수를 하나 선언한 후에 이 변수 내의 멤버 내용을 입력받는 함수를 만들어 보도록 하자.

```
void add_person(struct person * ptr);
```

C Workbook

```
void main()
{
    struct person employee;
    struct person * p = &employee;

    add_person(p);
    printf("%s/ %s/ %d\n", employee.name,employee.address,employee.age);
}

void add_person(struct person * ptr);
{
    printf("이름은? ");
    gets(ptr->name);
    printf("주소는? ");
    gets(ptr->address);
    printf("나이는? ");
    scanf("%d", &(ptr->age));
}
```

위 프로그램에서 보는 바와 같이 구조체 포인터를 통해 구조체 멤버를 접근할 때에는 ‘.’ 연산자 대신 ‘->’ 연산자를 사용해야 함을 유의하라.

○ 실습 문제

[N01] 숫자 3개 정렬하는 함수 만들기

메인 함수에서 정수 3개(num1, num2, num3)를 입력받은 후, 이 숫자들의 주소값을 넘겨주면 크기가 큰 순서대로 num1, num2, num3에 저장되도록 변경하는 함수를 제작하여 테스트하라. 단, 함수의 원형은 다음과 같다.

```
void sort3num(int * n1, int * n2, int * n3);
```

메인함수에서는 다음과 같이 호출하면 된다.

```
sort3num(&num1, &num2, &num3);
```



[N02] 최댓값과 최솟값 구하기

10개의 정수 배열을 사용하여 숫자를 입력받은 후에, 포인터를 이용하여 이 배열 중에서 가장 큰 수와 가장 작은 수를 찾아 출력하라.

다음 변수들은 반드시 사용할 것.

```
int num[10];      // 숫자 배열  
int *p;          // 포인터
```

The screenshot shows a Windows Command Prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1번째 숫자를 입력하시오. 11  
2번째 숫자를 입력하시오. 22  
3번째 숫자를 입력하시오. 55  
4번째 숫자를 입력하시오. 66  
5번째 숫자를 입력하시오. 77  
6번째 숫자를 입력하시오. 99  
7번째 숫자를 입력하시오. 88  
8번째 숫자를 입력하시오. 44  
9번째 숫자를 입력하시오. 33  
10번째 숫자를 입력하시오. 54  
최댓값 99, 최솟값 11 입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

[N03] 문자열 길이 구하기

문자열을 입력받은 후에, 포인터를 이용하여 이 문자열의 문자 개수가 몇 개인지 알아내는 함수를 만들고 이를 이용하여 문자열의 개수를 출력하라. 단, 문자열의 길이를 알아내는 `strlen()`함수를 사용하지 말 것. 함수 원형은 다음과 같다.

```
int string_length(char * str);
```



[N04] 문자열 비교하기

2개의 문자열을 입력받은 후에, 포인터를 이용하여 이 문자열 중에서 영어사전 순서에 따를 때 먼저 나타날 문자열을 알아내는 함수를 만들고, 이를 사용하여 결과를 출력하라. 단, 각각의 문자를 비교할 때에 대소문자는 무시하라. 함수 원형은 다음과 같다.

```
int which_first(char * str1, char * str2); // 리턴 값 : 사전상 먼저 나타날 문자열의 순서, 1(첫 번째 문자열) 또는 2(두번째 문자열)
```

The screenshot shows two separate command-line windows. Both windows have the title 'C:\Windows\system32\cmd.exe'. The top window displays the following text:
첫번째 문자열을 입력하시오. -->Hello
두번째 문자열을 입력하시오. -->World
첫번째 문자열이 사전에 먼저 나옵니다.
계속하려면 아무 키나 누르십시오 . . .

The bottom window displays the following text:
첫번째 문자열을 입력하시오. -->boy
두번째 문자열을 입력하시오. -->atom
두번째 문자열이 사전에 먼저 나옵니다.
계속하려면 아무 키나 누르십시오 . . .

[N05] 문자열 역순으로 만들기

하나의 문자열을 입력받은 후에, 포인터를 이용하여 이 문자열을 역순으로 한 문자열을 리턴하는 함수를 만들고 테스트하라. 함수 원형은 다음과 같다.

```
char * reverse_string(char * str);
```

The screenshot shows a single Windows cmd.exe window with the title 'C:\Windows\system32\cmd.exe'. It displays the following text:
문자열을 입력하시오. --> Hello World!
역순으로 변환한 문자열은 --> !d!rrow olleH
계속하려면 아무 키나 누르십시오 . . .

[N06] 로그인 데이터 입력

다음 구조체를 이용하여 한 사용자의 데이터를 입력받는 함수를 만들고, 테스트하라.

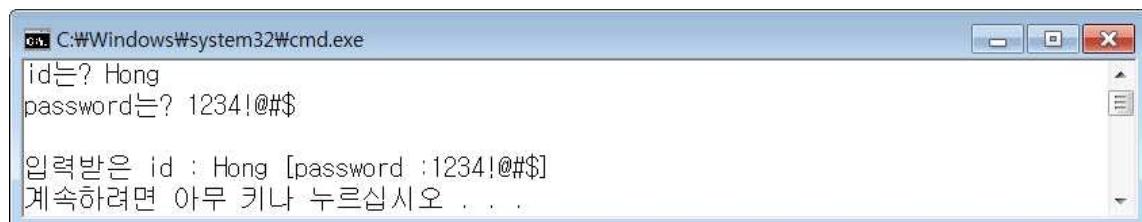
```
struct login_data{  
    char id[20];  
    char password[20];  
};
```

메인에서는 다음과 같이 구조체 포인터를 사용하라.

```
struct login_data * user1;
```

단, 입력받은 함수의 원형은 다음과 같다.

```
void add_user(struct login_data * user);
```



[N07] 좌표 거리 계산

다음 좌표 구조체의 포인터 2개를 파라미터로 입력받아 이 두 좌표 사이의 거리를 리턴하는 함수를 제작하고, 이를 사용하여 입력받은 2개의 좌표 거리를 출력하라.

```
struct point{  
    float x; // x좌표 값  
    float y; // y좌표 값  
};  
메인에서는 다음과 같이 구조체 포인터를 사용하라.  
struct point *point1, *point2;
```

단, 두 좌표 사이의 거리는 $(x\text{값의 차이})^2 + (y\text{값의 차이})^2$ 의 제곱근이다. 계산할 때에 `sqrt()`와 `pow()`함수를 사용하라.

```
float calc_distance(struct point * p1, struct point * p2); // 두 점의 거리 구하는 함수
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32'. The window contains the following text:

```
첫번째 좌표의 x값은? 1
첫번째 좌표의 y값은? 1
두번째 좌표의 x값은? 4
두번째 좌표의 y값은? 5

두 좌표 사이의 거리는 5.0입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[N08] 사용자 정의형 메뉴판 만들기

어떤 식당의 메뉴판의 구조가 다음과 같이 구성되어 있을 때, 이 메뉴판의 메뉴 정보를 구조체로 선언하고, 구조체 포인터를 파라미터로 넘겨받아 메뉴 정보를 입력받는 함수를 구현하라. 이를 이용하여 5개의 메뉴를 입력받은 후 한꺼번에 화면에 출력하라.

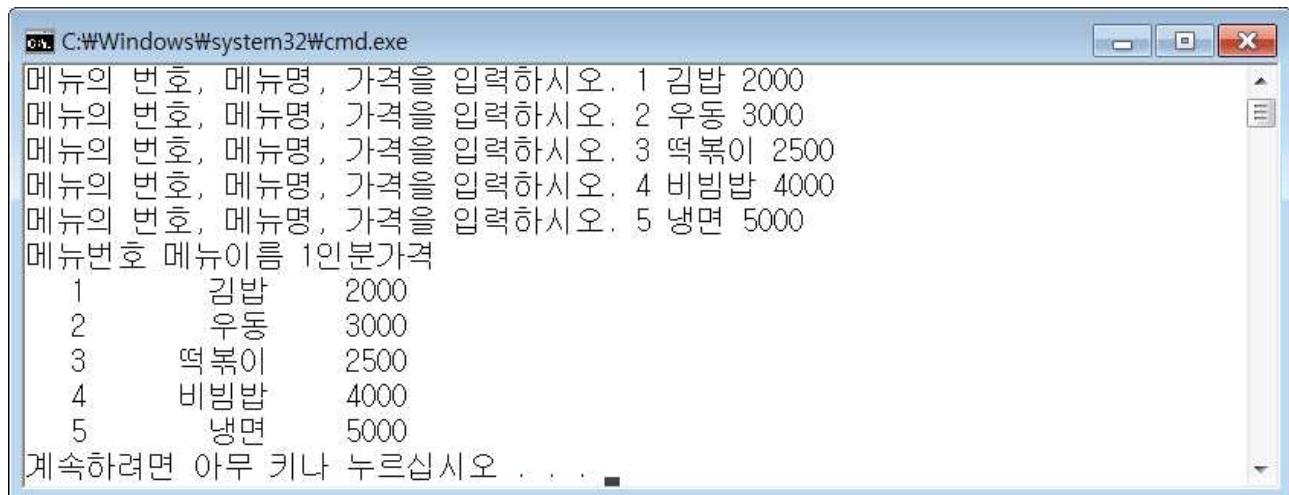
```
struct menu{
    int no;           // 메뉴번호
    char name[20];   // 메뉴명
    int price;        // 가격
}
```

메인에서는 다음과 같은 구조체 포인터를 사용하라.

```
struct menu * menu1;
struct menu * menu2;
struct menu * menu3;
struct menu * menu4;
struct menu * menu5;
```

함수 원형은 다음과 같다.

```
void ask_menu(struct menu * m);
```



[Step 0] 구조체와 배열 함께 사용하기

이번 단계에서는 여러 개의 포인터를 배열로 선언하여 사용하는 방법을 연습하도록 하겠다. 예를 들어 앞의 예제 ex_M_1.c에서 선언한 구조체를 사용하여 10명의 인적 사항을 다루려면 어떻게 해야 할까?

```
struct person{
    char name[20];
    char address[80];
    int age;
};

struct person plist[10];
```

이렇게 구조체 배열로 선언하여 사용할 수가 있다. 하지만 이런 경우에는 구조체 10개에 해당하는 메모리 공간을 미리 확보해야 하는 비효율성을 감수해야 한다. 따라서 구조체 포인터 배열을 선언하여 메모리 공간을 최소화하고 필요한 경우에만 구조체를 만들어 연결하게 함으로써 효율적인 프로그래밍이 될 수 있다. 다음 프로그램을 참조하라.

```
struct person * plist[10]; // 구조체 포인터의 배열 선언
plist[0] = (struct person *)malloc(sizeof(struct person)); // 첫번째 포인터에 구조체 할당
add_person(plist[0]); // Step N 에서 제작한 함수
printf("%s/ %s/ %d\n", plist[0]->name, plist[0]->address, plist[0]->age);
```

○ 실습 문제

[001] 로그인 데이터 다루기

다음 구조체를 이용하여 10명의 로그인 데이터를 보관하기 위한 구조체 포인터 배열을 선언한 후, 데이터를 입력받는 함수를 통해 로그인 데이터를 구축하라. 입력하기 전에 malloc()이 필요하며, 입력 후에는 user_count가 증가해야 한다.

```
struct login_data{
    char id[20];
    char password[20];
};
```

단, 입력받는 함수의 원형은 다음과 같다.

`int add_user(struct login_data * userlist[], int count); // 사용자목록과 사용자 수`
 동작 : 현재 사용자 목록에 새 사용자를 추가한다.
 리턴값 : 추가작업 후에 변경된 사용자 수

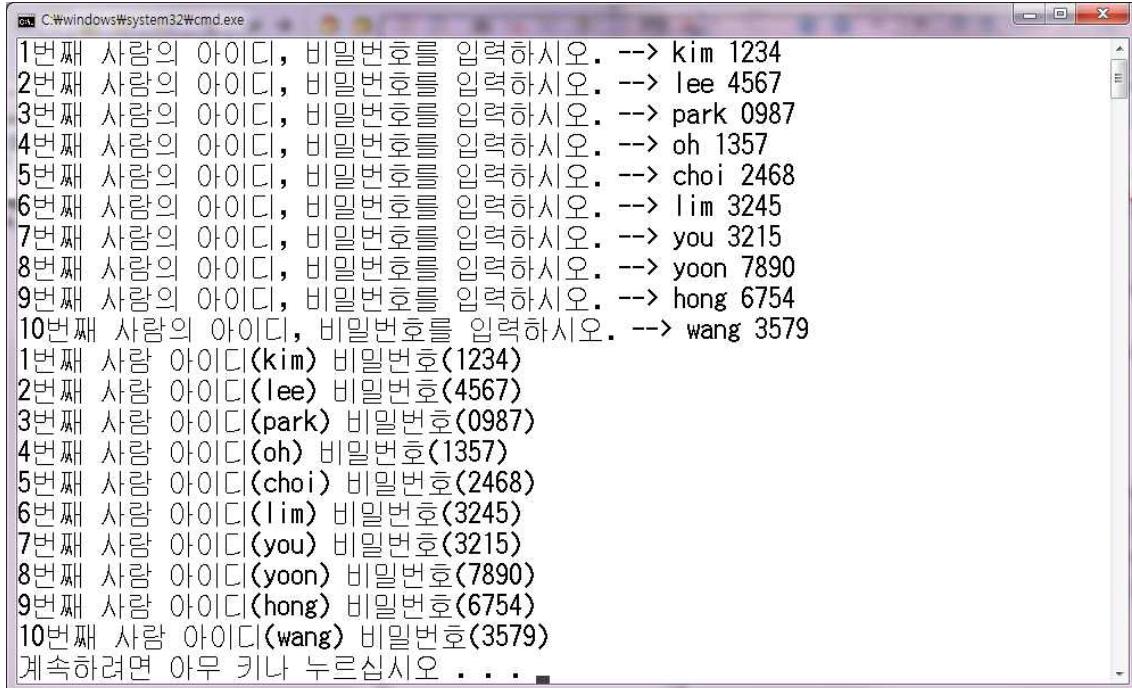
메인 함수에서 선언할 구조체 포인터 배열은 다음과 같다.

```
struct login_data * userlist[10]; // 10명의 로그인 데이터 보관용 포인터 배열
int user_count = 0; // 등록된 사용자 수
```

사용자 추가작업 구문

```
user_count = add_user(userlist, user_count);
```

C Workbook



```
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kim 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> lee 4567
3번째 사람의 아이디, 비밀번호를 입력하시오. --> park 0987
4번째 사람의 아이디, 비밀번호를 입력하시오. --> oh 1357
5번째 사람의 아이디, 비밀번호를 입력하시오. --> choi 2468
6번째 사람의 아이디, 비밀번호를 입력하시오. --> lim 3245
7번째 사람의 아이디, 비밀번호를 입력하시오. --> you 3215
8번째 사람의 아이디, 비밀번호를 입력하시오. --> yoon 7890
9번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 6754
10번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 3579
1번째 사람 아이디(kim) 비밀번호(1234)
2번째 사람 아이디(lee) 비밀번호(4567)
3번째 사람 아이디(park) 비밀번호(0987)
4번째 사람 아이디(oh) 비밀번호(1357)
5번째 사람 아이디(choi) 비밀번호(2468)
6번째 사람 아이디(lim) 비밀번호(3245)
7번째 사람 아이디(you) 비밀번호(3215)
8번째 사람 아이디(yoon) 비밀번호(7890)
9번째 사람 아이디(hong) 비밀번호(6754)
10번째 사람 아이디(wang) 비밀번호(3579)
계속하려면 아무 키나 누르십시오 . . .
```

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

struct login_data{
    char id[20];
    char password[20];
};

int add_user(struct login_data * userlist[], int count); // 사용자목록과 사용자 수

void main()
{
    struct login_data * userlist[10]; // 10명의 로그인 데이터 보관용 포인터 배열
    int user_count = 0; // 등록된 사용자 수
    int i;

    for (i=0;i<10 ;i++ )
    {
        userlist[user_count] = (struct login_data *)malloc(sizeof(struct login_data));
        user_count = add_user(userlist, user_count);
    }
    for (i=0;i<user_count ;i++ )
    {
        printf("%d번째 사람 아이디(%s) 비밀번호(%s)\n", i+1, userlist[i]->id,
        userlist[i]->password);
    }
}
```

[O02] 사용자 추가 함수 변경하기

위 [O01] 프로그램에서 만든 add_user()함수에 다음과 같은 기능을 추가하라.

- 사용자 ID를 입력받을 때에 이미 등록된 사용자 목록에 동일한 ID가 있으면 다시 입력받도록 한다.
- 사용자 ID와 비밀번호는 반드시 4글자 이상이어야 한다.

```

C:\Windows\system32\cmd.exe
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kim 1234
아이디는 4글자 이상으로 입력해주세요.
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kkim 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> slee 234
비밀번호는 4글자 이상으로 입력해주세요.
2번째 사람의 아이디, 비밀번호를 입력하시오. --> slee 2345
3번째 사람의 아이디, 비밀번호를 입력하시오. --> kkim 5678
동일한 아이디가 이미 등록되어 있습니다.
3번째 사람의 아이디, 비밀번호를 입력하시오. --> slee 5678
동일한 아이디가 이미 등록되어 있습니다.
3번째 사람의 아이디, 비밀번호를 입력하시오. --> sslee 5678
4번째 사람의 아이디, 비밀번호를 입력하시오. --> park 0987
5번째 사람의 아이디, 비밀번호를 입력하시오. --> choi 1357
6번째 사람의 아이디, 비밀번호를 입력하시오. --> you 2468
아이디는 4글자 이상으로 입력해주세요.
6번째 사람의 아이디, 비밀번호를 입력하시오. --> ryou 2468
7번째 사람의 아이디, 비밀번호를 입력하시오. --> yoon 3456
8번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 0987
9번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 1313
10번째 사람의 아이디, 비밀번호를 입력하시오. --> kmin 1569
1번째 사람 아이디(kkim) 비밀번호(1234)
2번째 사람 아이디(slee) 비밀번호(2345)
3번째 사람 아이디(sslee) 비밀번호(5678)
4번째 사람 아이디(park) 비밀번호(0987)
5번째 사람 아이디(choi) 비밀번호(1357)
6번째 사람 아이디(ryou) 비밀번호(2468)
7번째 사람 아이디(yoon) 비밀번호(3456)
8번째 사람 아이디(wang) 비밀번호(0987)
9번째 사람 아이디(hong) 비밀번호(1313)
10번째 사람 아이디(kmin) 비밀번호(1569)
계속하려면 아무 키나 누르십시오 . . .

```

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

struct login_data{
    char id[20];
    char password[20];
};

int add_user(struct login_data * userlist[], int count); // 사용자목록과 사용자 수

void main()
{
    struct login_data * userlist[10]; // 10명의 로그인 데이터 보관용 포인터 배열
    int user_count = 0; // 등록된 사용자 수
    int i;

    for (i=0;i<10 ;i++ )
    {
        userlist[user_count] = (struct login_data *)malloc(sizeof(struct login_data));
        user_count = add_user(userlist, user_count);
    }
    for (i=0;i<user_count ;i++ )
    {
        printf("%d번째 사람 아이디(%s) 비밀번호(%s)\n", i+1, userlist[i]->id,
userlist[i]->password);
    }
}
```

[O03] 사용자 검사 기능 만들기

위 [O02] 프로그램에 사용자 정보를 확인하는 함수를 추가하고, 테스트하라.

- 함수 원형은 다음과 같다.

```
int check_user(struct login_data * userlist[], int count, struct login_data * tempuser);
```

파라미터 : 사용자 목록, 사용자 수, 검사하려는 사용자 임시정보

리턴값 : 1 아이디와 비밀번호 둘다 맞음, 0 아이디는 맞는데 비밀번호 틀림, -1 맞는 아이디 없음

```
C:\Windows\system32\cmd.exe
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kkim 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 3456
3번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 0987
1번째 사람 아이디(kkim) 비밀번호(1234)
2번째 사람 아이디(wang) 비밀번호(3456)
3번째 사람 아이디(hong) 비밀번호(0987)
로그인할 아이디는? --> kkim
로그인할 비밀번호? --> 12345
아이디는 맞는데 비밀번호가 틀립니다.
로그인할 아이디는? --> slee
로그인할 비밀번호? --> 1234
맞는 아이디가 없습니다.
로그인할 아이디는? --> kkim
로그인할 비밀번호? --> 1234
아이디와 비밀번호 둘다 맞습니다.
계속하려면 아무 키나 누르십시오 . . .
```

```

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

struct login_data{
    char id[20];
    char password[20];
};

int add_user(struct login_data * userlist[], int count); // 사용자목록과 사용자 수
int check_user(struct login_data * userlist[], int count, struct login_data * tempuser);

void main()
{
    struct login_data * userlist[10]; // 10명의 로그인 데이터 보관용 포인터 배열
    struct login_data * user; // 로그인하기 위해 입력되는 사용자 정보
    int user_count = 0; // 등록된 사용자 수
    int i, check;

    for (i=0;i<3 ;i++ )
    {
        userlist[user_count] = (struct login_data *)malloc(sizeof(struct login_data));
        user_count = add_user(userlist, user_count);
    }
    for (i=0;i<user_count ;i++ )
    {
        printf("%d번째 사람 아이디(%s) 비밀번호(%s)\n", i+1, userlist[i]->id,
userlist[i]->password);
    }

    user = (struct login_data *)malloc(sizeof(struct login_data));
    while(1){
        printf("로그인할 아이디는? --> ");
        scanf("%s", user->id);
        printf("로그인할 비밀번호? --> ");
        scanf("%s", user->password);
        check = check_user(userlist, user_count, user);
        if (check == 0)
            printf("아이디는 맞는데 비밀번호가 틀립니다.\n");
        else if (check == -1)
            printf(" 맞는 아이디가 없습니다.\n");
        else {
            printf("아이디와 비밀번호 둘다 맞습니다.\n");
            break;
        }
    }
}

```

[O04] 사용자 목록 기능 만들기

위 [O03] 프로그램에 모든 사용자 정보를 출력하는 함수를 추가하고, 테스트하라.

- 함수 원형은 다음과 같다.

```
void all_user(struct login_data * userlist[], int count);
```

파라미터 : 사용자 목록, 사용자 수

리턴값 : 없음

```

C:\Windows\system32\cmd.exe
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kkim 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 2345
3번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 0987
4번째 사람의 아이디, 비밀번호를 입력하시오. --> slee 8796
5번째 사람의 아이디, 비밀번호를 입력하시오. --> choi 1357
6번째 사람의 아이디, 비밀번호를 입력하시오. --> park 9753
7번째 사람의 아이디, 비밀번호를 입력하시오. --> yoon 1357
8번째 사람의 아이디, 비밀번호를 입력하시오. --> ming 9807
9번째 사람의 아이디, 비밀번호를 입력하시오. --> edge 3456
10번째 사람의 아이디, 비밀번호를 입력하시오. --> silk 3214

```

등록된 사용자 목록은 다음과 같습니다.

No	아이디	비밀번호
1	kkim	1234
2	wang	2345
3	hong	0987
4	slee	8796
5	choi	1357
6	park	9753
7	yoon	1357
8	ming	9807
9	edge	3456
10	silk	3214

계속하려면 아무 키나 누르십시오 . . .

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

struct login_data{
    char id[20];
    char password[20];
};

int add_user(struct login_data * userlist[], int count); // 사용자목록과 사용자 수
int check_user(struct login_data * userlist[], int count, struct login_data * tempuser);
void all_user(struct login_data * userlist[], int count);

void main()
{
    struct login_data * userlist[10]; // 10명의 로그인 데이터 보관용 포인터 배열
    struct login_data * user; // 로그인하기 위해 입력되는 사용자 정보
    int user_count = 0; // 등록된 사용자 수
    int i, check;

    for (i=0;i<10 ;i++ )
    {
        userlist[user_count] = (struct login_data *)malloc(sizeof(struct login_data));
        user_count = add_user(userlist, user_count);
    }
    all_user(userlist, user_count);
}
```

[O05] 좌표 위치 계산

다음 구조체를 이용하여 10개의 좌표 데이터를 입력받은 후, 각각의 좌표가 1,2,3,4분면 중에 어디에 위치하고 있는지를 출력하고, 각 사분면 별로 좌표의 갯수를 출력하라.

단, 10개의 좌표는 구조체포인터 배열을 선언하여 사용하라.

```
struct point{
    float x; // x좌표 값
    float y; // y좌표 값
};
```

- 메인에서 사용할 구조체 포인터 배열

```
struct point * mypoint[10];
```

- 위치 검사 방법

단, 1사분면은 x값, y값이 모두 양수, 2사분면은 x값은 음수, y값은 양수, 3사분면은 x값, y값이 모두 음수, 4사분면은 x값은 양수, y값은 음수인 경우이다.

```
C:\Windows\system32\cmd.exe
1번째 좌표의 x, y값을 입력하시오. --> 1.0 2.0
2번째 좌표의 x, y값을 입력하시오. --> -1.0 2.0
3번째 좌표의 x, y값을 입력하시오. --> -1.0 -2.0
4번째 좌표의 x, y값을 입력하시오. --> 1.0 -2.0
5번째 좌표의 x, y값을 입력하시오. --> 3.5 4.7
6번째 좌표의 x, y값을 입력하시오. --> 10.1 45.0
7번째 좌표의 x, y값을 입력하시오. --> -10.1 100.0
8번째 좌표의 x, y값을 입력하시오. --> -5.5 -9.0
9번째 좌표의 x, y값을 입력하시오. --> 3.0 -1.0
10번째 좌표의 x, y값을 입력하시오. --> 2.0 -7.0
1번째 좌표는 1사분면에 위치합니다.
2번째 좌표는 2사분면에 위치합니다.
3번째 좌표는 3사분면에 위치합니다.
4번째 좌표는 4사분면에 위치합니다.
5번째 좌표는 1사분면에 위치합니다.
6번째 좌표는 1사분면에 위치합니다.
7번째 좌표는 2사분면에 위치합니다.
8번째 좌표는 3사분면에 위치합니다.
9번째 좌표는 4사분면에 위치합니다.
10번째 좌표는 4사분면에 위치합니다.
1사분면의 좌표는 모두 3개입니다.
2사분면의 좌표는 모두 2개입니다.
3사분면의 좌표는 모두 2개입니다.
4사분면의 좌표는 모두 3개입니다.
계속하려면 아무 키나 누르십시오 . . .
```

[006] 메뉴판 만들기

어떤 식당의 메뉴판이 다음과 같이 구성되어 있을 때, 다음 5개의 메뉴를 구조체 포인터 배열에 저장한 후, 이를 화면에 출력하라.

메뉴번호	메뉴이름	원산지	1인분 가격
1	삼겹살	국내산	9000
2	갈비살	미국산	15000
3	꽃등심	국내산	30000
4	양념갈비	호주산	25000
5	차돌박이	국내산	28000

```
struct menu{
    int no;           // 메뉴번호
    char name[20];   // 메뉴명
    char madein[20]; // 원산지
    int price;        // 가격
};
```

- 메인에서 사용할 구조체 포인터 배열
- ```
struct menu * mymenu[5];
```

| 메뉴번호 | 메뉴이름 | 원산지 | 1인분가격 |
|------|------|-----|-------|
| 1    | 삼겹살  | 국내산 | 9000  |
| 2    | 갈비살  | 미국산 | 15000 |
| 3    | 꽃등심  | 국내산 | 30000 |
| 4    | 양념갈비 | 호주산 | 25000 |
| 5    | 차돌박이 | 국내산 | 28000 |

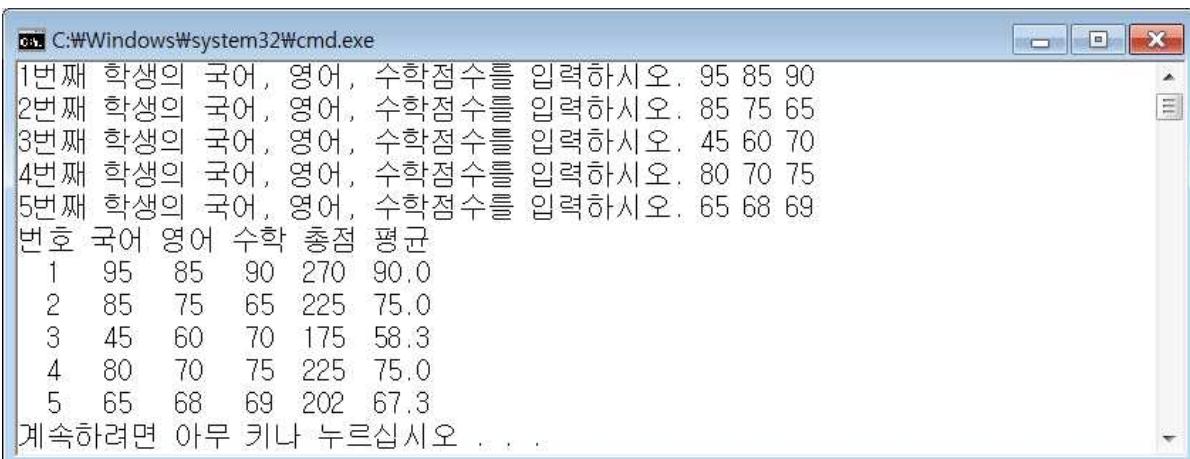
### [007] 학생성적 계산 프로그램

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 각 과목별 총점과 평균 점수를 계산하여 출력하라. 단, 학생들의 점수를 다음과 같은 구조체 포인터 배열을 사용하라.

```
struct jumsu{
 int kor; // 국어점수
 int eng; // 영어점수
 int mat; // 수학점수
 int sum; // 총점
 float average; // 평균
};
```

- 메인에서 사용할 구조체 포인터 배열

```
struct jumsu * student[5];
```



```
C:\Windows\system32\cmd.exe
1번째 학생의 국어, 영어, 수학점수를 입력하시오. 95 85 90
2번째 학생의 국어, 영어, 수학점수를 입력하시오. 85 75 65
3번째 학생의 국어, 영어, 수학점수를 입력하시오. 45 60 70
4번째 학생의 국어, 영어, 수학점수를 입력하시오. 80 70 75
5번째 학생의 국어, 영어, 수학점수를 입력하시오. 65 68 69
번호 국어 영어 수학 총점 평균
 1 95 85 90 270 90.0
 2 85 75 65 225 75.0
 3 45 60 70 175 58.3
 4 80 70 75 225 75.0
 5 65 68 69 202 67.3
계속하려면 아무 키나 누르십시오 . . .
```

## [Step P] 파일 사용하기

이번 단계에서는 텍스트 파일을 사용하여 데이터를 읽고 쓰는 방법을 연습하도록 하겠다. 파일을 사용하기 위해서는 파일 핸들이라는 특별한 형식의 포인터를 사용해야 한다. 그리고 특정 텍스트 파일을 읽거나 쓰기 위해서 먼저 파일을 열어야 한다. 파일로부터 읽어오는 구문을 살펴보자. 데이터 파일인 data.txt에는 다음과 같은 내용이 들어 있다고 가정한다.

```
홍길동
강원도 원주시 흥업면
25
```

위 파일로부터 데이터를 읽어들여 구조체에 담는 프로그램은 다음과 같다.

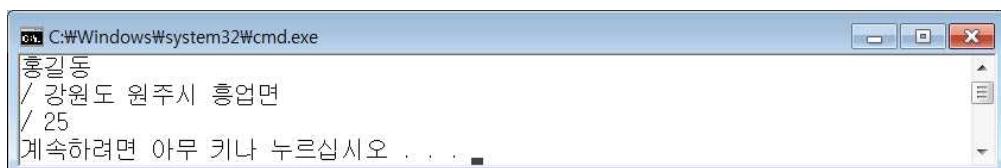
```
// 예제 ex_P_1.c
#include <stdio.h>

struct person{
 char name[20];
 char address[80];
 int age;
};

void main()
{
 FILE * fp; // 파일 핸들 선언
 struct person employee;
 fp = fopen("data.txt", "rt"); // 텍스트파일인 data.txt를 읽는 용도로 연다.

 fgets(employee.name, sizeof(employee.name), fp); // 파일에서 문자열을 읽어온다.
 fgets(employee.address, sizeof(employee.address), fp); // 파일에서 문자열을 읽어온다.
 fscanf(fp, "%d", &employee.age); // 파일에서 숫자를 읽어온다.

 fclose(fp); // 파일을 닫는다.
 printf("%s/ %s/ %d\n", employee.name, employee.address, employee.age);
}
```



파일에 데이터를 넣는 방법은 파일을 쓰기 용도로 연 다음에 적절한 파일 쓰기 함수들을 사용하면 되는데, 다음 프로그램을 참고하라.

```
// 예제 ex_P_2.c
#include <stdio.h>

struct person{
 char name[20];
 char address[80];
 int age;
};

void main()
{
 FILE * fp; // 파일 핸들 선언
 struct person employee = {"홍길동", "강원도 원주시 흥업면", 25}; // 초기화
 fp = fopen("data.txt", "wt"); // 텍스트파일인 data.txt를 쓰는 용도로 연다.

 fprintf(fp, "%s\n", employee.name); // 파일에서 한 줄을 쓴다.
 fprintf(fp, "%s\n", employee.address); // 파일에서 한 줄을 쓴다.
 fprintf(fp, "%d", employee.age); // 파일에서 숫자 하나를 쓴다.

 fclose(fp); // 파일을 닫는다.
}
```

## ○ 실습 문제

### [P01] 메뉴판 저장하기

위의 [O06] 프로그램에서 입력받은 5개의 메뉴들을 텍스트 파일인 menu.txt 파일에 저장하는 함수를 제작하고 테스트하라.

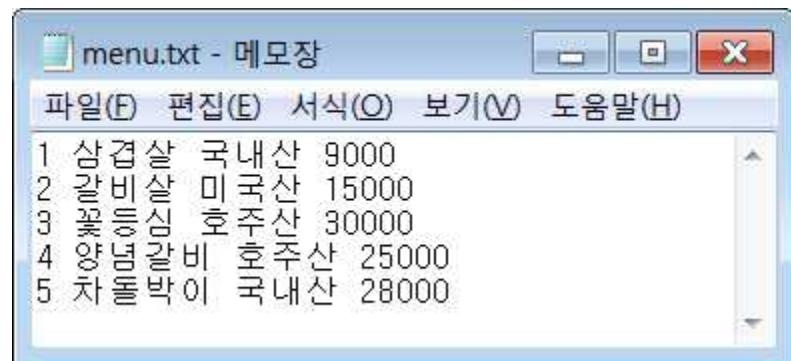
- 함수 원형은 다음과 같다.

```
void save_menu(struct menu * list[]);
```

파라미터 : 메뉴 목록

리턴값 : 없음

```
C:\Windows\system32\cmd.exe
1번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 1 삼겹살 국내산 9000
2번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 2 갈비살 미국산 15000
3번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 3 꽃등심 호주산 30000
4번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 4 양념갈비 호주산 25000
5번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 5 차돌박이 국내산 28000
메뉴번호 메뉴이름 원산지 1인분가격
 1 삼겹살 국내산 9000
 2 갈비살 미국산 15000
 3 꽃등심 호주산 30000
 4 양념갈비 호주산 25000
 5 차돌박이 국내산 28000
menu.txt에 저장하였습니다. 계속하려면 아무 키나 누르십시오 . . .
```



**[P02] 메뉴판 읽어오기**

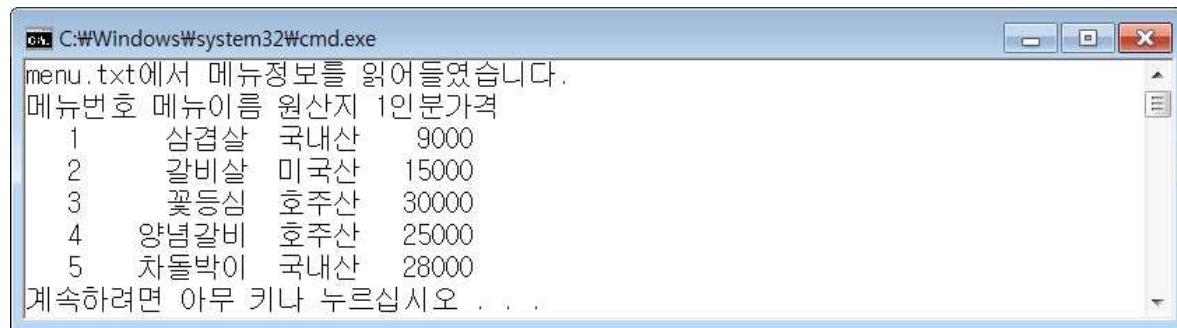
위의 [P01]문제에서 저장한 menu.txt 파일에서 메뉴 정보를 읽어오는 함수를 제작한 후, 이를 테스트하고, 읽어온 메뉴를 화면에 출력하라.

- 함수 원형은 다음과 같다.

```
void load_menu(struct menu * list[]);
```

파라미터 : 메뉴 목록

리턴값 : 없음



### [P03] 좌표 저장하기

위의 [O05]문제에서 입력받은 10개의 좌표 값을 point.txt 파일에 저장하는 함수를 제작하고, 테스트하라.

- 함수 원형은 다음과 같다.

```
void save_point(struct point * list[]);
```

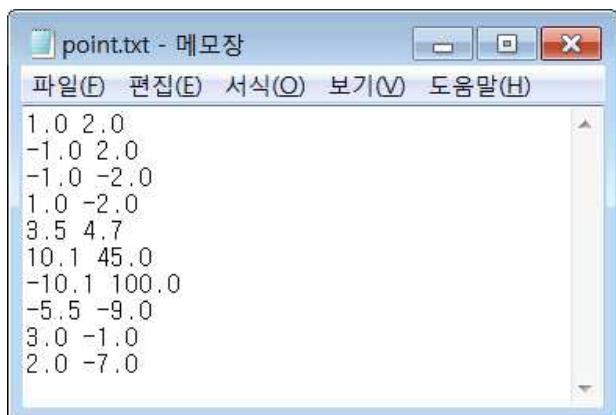
파라미터 : 좌표 목록

리턴값 : 없음

```

1번째 좌표의 x, y값을 입력하시오. --> 1.0 2.0
2번째 좌표의 x, y값을 입력하시오. --> -1.0 2.0
3번째 좌표의 x, y값을 입력하시오. --> -1.0 -2.0
4번째 좌표의 x, y값을 입력하시오. --> 1.0 -2.0
5번째 좌표의 x, y값을 입력하시오. --> 3.5 4.7
6번째 좌표의 x, y값을 입력하시오. --> 10.1 45.0
7번째 좌표의 x, y값을 입력하시오. --> -10.1 100.0
8번째 좌표의 x, y값을 입력하시오. --> -5.5 -9.0
9번째 좌표의 x, y값을 입력하시오. --> 3.0 -1.0
10번째 좌표의 x, y값을 입력하시오. --> 2.0 -7.0
1번째 좌표는 1사분면에 위치합니다.
2번째 좌표는 2사분면에 위치합니다.
3번째 좌표는 3사분면에 위치합니다.
4번째 좌표는 4사분면에 위치합니다.
5번째 좌표는 1사분면에 위치합니다.
6번째 좌표는 1사분면에 위치합니다.
7번째 좌표는 2사분면에 위치합니다.
8번째 좌표는 3사분면에 위치합니다.
9번째 좌표는 4사분면에 위치합니다.
10번째 좌표는 4사분면에 위치합니다.
1사분면의 좌표는 모두 3개입니다.
2사분면의 좌표는 모두 2개입니다.
3사분면의 좌표는 모두 2개입니다.
4사분면의 좌표는 모두 3개입니다.
point.txt에 저장하였습니다.
계속하려면 아무 키나 누르십시오 . .

```



## [P04] 좌표 불러오기

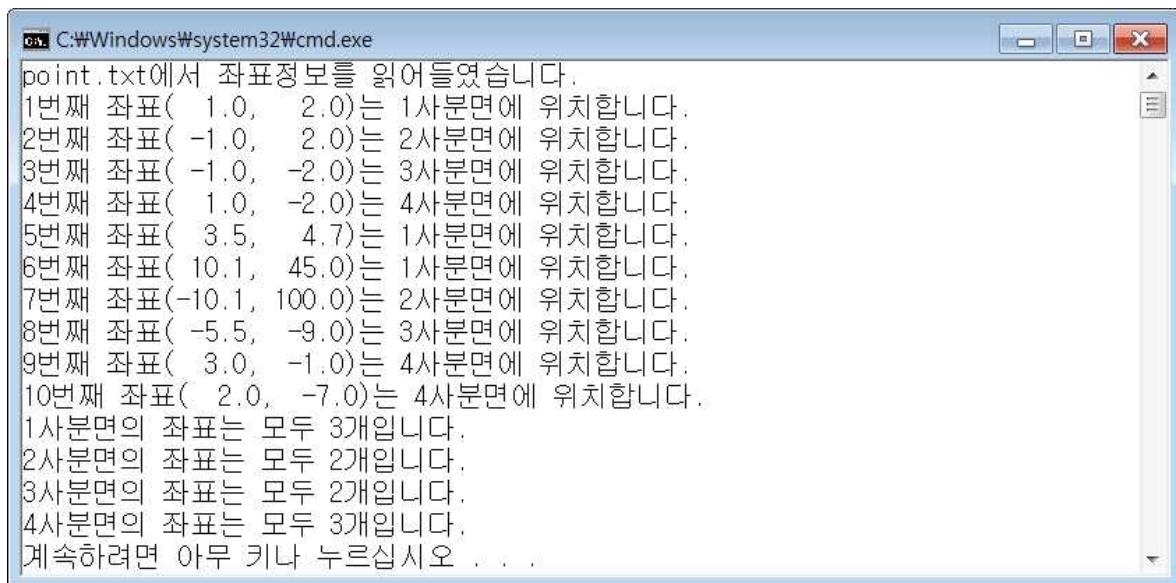
위의 [P03]문제에서 저장한 point.txt 파일로부터 10개의 좌표를 읽어오는 함수를 제작하고, 테스트하라.

- 함수 원형은 다음과 같다.

```
void load_point(struct point * list[]);
```

파라미터 : 좌표 목록

리턴값 : 없음



## [P05] 사용자 목록 저장하기

위의 [O04]문제에 다음과 같은 함수를 추가하라.

기능 : 현재의 모든 사용자 정보를 텍스트 파일인 user.txt 파일에 저장한다.

- 함수 원형은 다음과 같다.

```
void save_list(struct login_data * userlist[], int count);
```

파라미터 : 사용자 목록, 사용자 수

리턴값 : 없음

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kk im 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 2123
3번째 사람의 아이디, 비밀번호를 입력하시오. --> cho i 0099
4번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 7324
5번째 사람의 아이디, 비밀번호를 입력하시오. --> yoon 0000
6번째 사람의 아이디, 비밀번호를 입력하시오. --> mary 1212
7번째 사람의 아이디, 비밀번호를 입력하시오. --> david 0101
8번째 사람의 아이디, 비밀번호를 입력하시오. --> mathew 1211
9번째 사람의 아이디, 비밀번호를 입력하시오. --> james 0000
10번째 사람의 아이디, 비밀번호를 입력하시오. --> smith 3333
```

등록된 사용자 목록은 다음과 같습니다.

| No | 아이디    | 비밀번호 |
|----|--------|------|
| 1  | kk im  | 1234 |
| 2  | hong   | 2123 |
| 3  | cho i  | 0099 |
| 4  | wang   | 7324 |
| 5  | yoon   | 0000 |
| 6  | mary   | 1212 |
| 7  | david  | 0101 |
| 8  | mathew | 1211 |
| 9  | james  | 0000 |
| 10 | smith  | 3333 |

user.txt에 저장하였습니다.  
계속하려면 아무 키나 누르십시오 . . .



### [P06] 사용자 목록 불러오기

위의 [P05]문제에 다음과 같은 함수를 추가하라.

기능 : user.txt 파일로부터 사용자 정보를 읽어온다.

- 함수 원형은 다음과 같다.

```
int load_list(struct login_data * userlist[]);
```

파라미터 : 읽어와서 저장할 사용자 목록

리턴값 : 읽어들인 사용자 수

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32'. The window displays the following text:

```
C:\Windows\system32\cmd.exe
user.txt에서 10명의 사용자 정보를 읽어들였습니다.

등록된 사용자 목록은 다음과 같습니다.

No 아이디 비밀번호
1 kkim 1234
2 hong 2123
3 choi 0099
4 wang 7324
5 yoon 0000
6 mary 1212
7 david 0101
8 matthew 1211
9 james 0000
10 smith 3333

계속하려면 아무 키나 누르십시오 . . .
```

### [P07] 학생 점수 결과 저장하기

위의 [O07]문제에 다음과 같은 함수를 추가하라.

기능 : 현재의 모든 학생들의 점수, 합계, 평균 정보를 텍스트 파일인 jumsu.txt 파일에 저장한다.

- 함수 원형은 다음과 같다.

```
void save_jumsu(struct jumsu * list[], int count);
```

파라미터 : 점수 목록, 학생 수

리턴값 : 없음

```
C:\Windows\system32\cmd.exe
1번째 학생의 국어, 영어, 수학점수를 입력하시오. 95 85 75
2번째 학생의 국어, 영어, 수학점수를 입력하시오. 90 80 70
3번째 학생의 국어, 영어, 수학점수를 입력하시오. 60 85 44
4번째 학생의 국어, 영어, 수학점수를 입력하시오. 77 88 55
5번째 학생의 국어, 영어, 수학점수를 입력하시오. 80 70 40
번호 국어 영어 수학 총점 평균
 1 95 85 75 255 85.0
 2 90 80 70 240 80.0
 3 60 85 44 189 63.0
 4 77 88 55 220 73.3
 5 80 70 40 190 63.3
jumsu.txt에 저장하였습니다.
계속하려면 아무 키나 누르십시오 . . .
```

