

10주차 강의

- 10주 화상강의
 - 8장 함수 연습문제 풀이
- 10주 동영상 강의
 - 9장 함수와 변수

01 함수에 대한 설명 중 잘못된 것은?

- ① 함수를 정의하기 전에 반드시 함수 원형 정의가 있어야 한다.
- ② 함수의 매개 변수는 전혀 없을 수도 있다.
- ③ 함수는 값을 반환하지 않을 수도 있다.
- ④ 함수 안에 문장이 하나도 없을 수도 있다.

02 다음 함수 원형 정의 중에서 int형을 반환하는 함수는?

- ① void func(int); ② double func(int, int); ③ int func(double x); ④ void func(void);

03 다음의 함수 원형 정의 중에서 잘못된 것은?

- ① int func(int x, y); ② void func(x, y); ③ char func(double); ④ void func(void, void);

04 int func(int);의 원형을 가지는 함수의 호출로 옳은 것은?

- ① func(10); ② int func(10); ③ func(int); ④ func();

05 다음 라이브러리 함수 호출의 반환값을 쓰라.

- (a) fabs(-1.72) (b) floor(1.72) (c) ceil(1.72)

06 다음 수식의 반환값의 범위는?

- (a) rand()%10 (b) rand()%5 + 2

07 다음의 수학식을 계산하는 문장을 작성해보자.

- (a) $y = \log_{10}x + e^x$
- (b) $y = \sin(x) + \sqrt{x^2 - 2a} + 2^{10}$

08 다음과 같은 기술에 부합하는 함수 원형을 작성하라.

- (a) int형 매개 변수 n을 받아서 아무것도 반환하지 않는 print_error() 함수
- (b) double형 매개 변수 x, y를 받아서 double 형을 반환하는 larger_of() 함수
- (c) 어떠한 매개 변수도 갖지 않고 아무것도 반환하지 않는 side_effect() 함수

1. (1)

2. (3)

3. (1), (2), (4)

4. (1)

5. (a) 1.720000 (b) 1.000000 (c) 2.000000

6.(a) 0에서 9 (b) 2에서 6

7.

(a) $y = \log_{10}(x) + \exp(x);$

(b) $y = \sin(x) + \sqrt{x^2 - 2.0*a} + \text{pow}(2.0, 10);$

8.

(a) void print_error(int n);

(b) double larger_of(double x, double y);

(c) void side_effect(void);

09 왼쪽 박스의 함수 원형 정의에 적합한 return 문을 오른쪽 박스에서 찾아서 서로 연결하라.

- | | |
|---------------------------------------|--------------------------------|
| • <code>int f(void);</code> | • <code>return;</code> |
| • <code>void g(int, int);</code> | • <code>return 1.0;</code> |
| • <code>double h(double, int);</code> | • <code>return 10 + 20;</code> |

9. `int f(void) ----- return 10 + 20;`
`void g(int, int) ----- return;`
`double h(double, int); ----- return 'a' + 1.0;`

10 다음은 1부터 10까지의 합을 계산하는 프로그램이다. 프로그램의 빈칸을 채워라.

```
#include <stdio.h>

__ int f(int n); _____ // 함수 f()의 원형 정의

int main(void)
{
    __ f(10); _____ // f()를 인수 10으로 호출
    return 0;
}

int f(int n)
{
    int i, result = 0;

    for(i = 0; i <= n; i++)
        result += i;

    return result; // 변수 result의 값을 반환
}
```

11 다음의 함수 원형 정의가 올바른지를 결정하고 만약 잘못된 점이 있다면 이유를 설명하고 바르게 수정하라.

- (d) `double f(double x, y);`
(e) `(int) f(double x, double y);`
(f) `int f((int)x, (int)y);`
(g) `float get_area(radius, pi);`

- (a) `double f(double x, double y);`
(b) `int f(double x, double y);`
(c) `int f(int x, int y);`
(d) `float get_area(float radius, float pi);`

12 다음의 프로그램에서 붉은 색으로 표시된 부분을 함수로 작성하여 프로그램을 수정하시오. 함수는 인수와 반환값을 갖도록 설계하라.

```
#include <stdio.h>

int main(void)
{
    int i, n, sum = 0;

    printf("정수를 입력하시오: ");
    scanf("%d", &n);

    for(i = 0; i <= n; i++)
        sum += i;

    printf("0부터 %d까지의 합은%d입니다.\n", n, sum);
    return 0;
}
```

이 부분을 함수로 작성

13 다음 코드에서 잘못된 곳(컴파일 오류 및 경고)을 지적하라. 논리적인 오류도 지적하라.

(a) <pre>int half_of(int x); { return x / 2; }</pre>	(b) <pre>void print_message(void); int main(void) { print_message(3); }</pre>
(c) <pre>double half_of(int); int main(void) { printf("%f", half_of(10.0)); return 0; } double half_of(double x) { return x / 2.0; }</pre>	(d) <pre>int sum(int x, int y) { sum = x+y; }</pre>
	(e) <pre>void sum(void) { int x=1, y=2, z=3; return x + y + z; }</pre>

12.

```
#include <stdio.h>
int f(int x);
```

```
int f(int x)
{
    int i, sum=0;
    for(i = 0; i <= x; i++)
        sum += i;

    return sum;
}
```


```
int main(void)
{
    int n;
    printf("정수를 입력하시오:\n");
    scanf("%d", &n);
    printf("0부터 %d까지의 합은 %d입니다.\n", n, f(n));
}
```

13.

- (a) `int half_of(int x);` -> `int half_of(int x)`
- (b) 함수 원형의 매개 변수 개수와 함수 호출시의 인수 개수가 다르다.
- (c) 함수 원형의 매개 변수 타입과 함수 정의의 매개 변수 타입이 서로 다르다.
- (d) `sum = x + y` -> `return x + y;`
- (e) 반환형이 정의되지 않았는데 값을 반환하였다.

01 주어진 실수를 제공하여 반환하는 함수 `double square(double)`을 작성한다. `square()` 함수를 테스트하는 프로그램을 작성하라.

실행결과



```
C:\WINDOWS\system32\cmd.exe
정수를 입력하시오: 2.0
주어진 정수 2.000000의 제곱은 4.000000입니다.
```

HINT double형을 입력받을 때는 `scanf("%lf", &x);`를 사용한다.

```
#include<stdio.h>
double square(double x);

int main()
{
    double m, n;
    printf("정수를 입력하시오: ");
    scanf("%lf", &m);
    n = square(m);
    printf("주어진 정수 %f의 제곱은 %f입니다. \n", m, n);
}

double square(double x)
{
    double p;
    p = x * x;
    return (p);
}
```

02 전달된 문자가 알파벳 문자인지 아닌지를 검사하는 함수 `check_alpha()`를 작성하고 이것을 호출하여서 사용자가 입력한 문자가 알파벳('a'에서 'z'까지)인지를 판단하여 출력하는 프로그램을 작성하라.

실행결과



```
C:\WINDOWS\system32\cmd.exe
문자를 입력하시오: k
k는 알파벳 문자입니다.
```

HINT 문자를 입력받을 때는 `ch = getchar()`를 사용해도 좋다, `scanf("%c", &ch);`를 사용해도 된다.

```
int check_alpha(char c)
{
    if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
        return 1;
    else
        return 0;
}

int main(void)
{
    char c;
    printf("문자를 입력하시오: ");
    scanf("%c", &c);
    if (check_alpha(c))
        printf("%c는 알파벳 문자입니다. \n", c);
    else
        printf("%c는 알파벳 문자가 아닙니다. \n", c);
    return 0;
}
```

03 원의 면적을 구하는 문제를 함수로 작성하여 보자. 다음과 같은 식을 이용하라. 원의 면적을 구하는 함수 `cal_area(double radius)`를 작성하고 함수를 호출하여 원의 면적을 출력하는 전체 프로그램을 완성하라.



면적 = 원주율 * 반지름 * 반지름

실행결과

```
C:\WINDOWS\system32\cmd.exe
원의 반지름을 입력하시오: 10.0
원의 면적은 314.159200입니다.
```

HINT 원주율은 기호 상수로 표현하여 보자. 함수 원형도 정의한다.

```
double get_radius()
{
    double r;
    printf("원의 반지름을 입력하시오:");
    scanf("%lf", &r);
    return r;
}

double cal_area(double r)
{
    return 3.141592 * r * r;
}

int main(void)
{
    double r;
    r = get_radius();
    printf("원의 면적은 %f입니다.\n", cal_area(r));

    return 0;
}
```

04 우리는 앞에서 윤년을 구하는 알고리즘을 학습하였다. 이것을 함수 `is_leap(int year)` 함수로 작성하고 이 함수를 사용하여 사용자가 입력한 연도가 윤년인지를 출력하는 프로그램을 작성하라.

실행결과

```
C:\WINDOWS\system32\cmd.exe
연도를 입력하시오: 2012
2012년은 366일입니다.
```

HINT 윤년은 4의 배수이지만 100의 배수는 제외하고 400의 배수는 무조건 추가하면 구할 수 있다. 윤년이면 366이고 평년이면 365일이 된다.

```
int is_leap(int y);
int main(void)
{
    int year;

    printf("연도를 입력하시오: ");
    scanf("%d", &year);

    if (is_leap(year) == 1)
        printf("%d년은 366일입니다.\n", year);
    else
        printf("%d년은 365일입니다.\n", year);

    return 0;
}

int is_leap(int y)
{
    return (y % 4 == 0) && (y % 100 != 0) || (y % 400 == 0);
}
```


- 05 실수를 정수로 변환하면 소수점 이하는 잘려서 없어지게 된다. 예를 들어서 6.999를 정수로 변환하면 6이 된다. 실수에 0.5을 더하여 소수점 이하를 버리는 반올림 연산을 수행하는 함수 `round(double f)`를 작성하고 테스트하라. 단 `f`는 양수라고 가정하라.

실행결과

```
C:\WINDOWS\system32\cmd.exe
실수를 입력하시오: 3.141592
반올림한 값은 3입니다.
```

HINT 반올림시키는 가장 간단한 방법은 `(int)(x+0.5)`이다.

```
int round(double a);
int main(void)
{
    double a;

    printf("실수를 입력하시오: ");
    scanf("%lf", &a);

    printf("반올림한 값은 %d입니다.\n", round(a));

    return 0;
}

int round(double a)
{
    return (int)(a + 0.5);
}
```

- 06 다음과 같은 간단한 기능을 하는 함수들을 작성하고, 사용자로부터 임의의 값을 입력받은 후에 작성한 함수들을 테스트하여 보자.

- (a) 주어진 정수가 짝수이면 1을 반환하고 홀수이면 0을 반환하는 함수 `int even(int n)`
- (b) 주어진 정수의 절대값을 구하는 함수 `int absolute(int n)`
- (c) 주어진 정수가 음수이면 -1을, 양수이면 1을 0이면 0을 반환하는 함수 `int sign(int n)`

실행결과

```
C:\WINDOWS\system32\cmd.exe
정수를 입력하시오: 12
even()의 결과: 짝수
absolute()의 결과: 12
sign()의 결과: 양수
```

HINT 함수 원형을 먼저 정의한 후에 함수를 정의하도록 하자. 함수들을 구현할 때, 조건 연산자 `?:`를 사용하여도 된다.

```
int even(int n);
int absolute(int n);
int sign(int n);

int main(void)
{
    int n;
    printf("정수를 입력하시오:");
    scanf("%d", &n);
    printf("even()의 결과: ");
    if (even(n) == 1)
        printf("짝수\n");
    else
        printf("홀수\n");
    printf("absolute()의 결과: %d\n", absolute(n));
    printf("sign()의 결과: ");
    if (sign(n) == 1)
        printf("양수\n");
    else
        printf("음수\n");
    return 0;
}
```

```
int even(int n)
{
    return (n % 2 == 0) ? 1 : 0;
}

int absolute(int n)
{
    return (n > 0) ? n : -n;
}

int sign(int n)
{
    return (n > 0) ? 1 : -1;
}
```

07 월급에 붙는 소득세를 계산하는 함수 `get_tax(int income)`를 작성하고 테스트하여 보자. 과표 구간은 1000만 원 이하 8%, 1000만 원 초과는 10%로 되어 있다고 가정한다. 사용자로부터 소득을 입력받아서 세금을 계산하는 프로그램을 작성하라.

실행결과

```
C:\WINDOWS\system32\cmd.exe
소득을 입력하시오(만원):2500
소득세는 230입니다.
```

HINT 소득이 1000만 원 초과이면 소득 중에서 1000만 원 미만은 8%를 적용하고 1000만 원이 넘는 부분만 10%를 적용한다.

```
int get_tax(int income);
```

```
int main(void)
{
    int income;
    printf("소득을 입력하시오(만원):");
    scanf("%d", &income);
    printf("소득세는 %d입니다.\n", get_tax(income));
    return 0;
}
```

```
int get_tax(int income)
{
    if (income > 1000) return (int)(1000 * 0.08 + (income - 1000) * 0.1);
    else return (int)(income * 0.08);
}
```

08 `sin()` 라이브러리 함수를 호출하여서 0도부터 180도까지 10도 단위로 사인 함수 값을 출력하여 보자. 추가적으로 아예 각도를 받아서 사인값을 반환하는 함수 `sin_degree(double degree)`를 작성하여 문제를 해결할 수 있는가?

실행결과

```
C:\WINDOWS\system32\cmd.exe
sin(0.000000)의 값은 0.000000
sin(10.000000)의 값은 0.173648
sin(20.000000)의 값은 0.342020
sin(30.000000)의 값은 0.500000
sin(40.000000)의 값은 0.642787
sin(50.000000)의 값은 0.766044
sin(60.000000)의 값은 0.866025
```

HINT `sin()` 함수는 라디안으로 인수받는다. 따라서 라디안을 각도로 변환하여야 한다. 변환식은 $(\pi * \text{각도}) / 180.00$ 이다. 또 `sin()` 함수를 이용하려면 `<math.h>`가 필요하다.

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

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

```
double sin_degree(double degree);
```

```
int main(void)
{
    double degree;

    for (degree = 0.0; degree <= 180.0; degree += 10.0)
        printf("sin(%f)의 값은 %f\n", degree, sin_degree(degree));
    return 0;
}
```

```
double sin_degree(double degree)
{
    return sin((3.141592 * degree) / 180.0);
}
```


09 난수(random number)는 컴퓨터를 이용한 문제 해결에서 많이 사용된다. 특히 수학적인 분석이 너무 복잡한 경우에 시뮬레이션을 사용하면 실제로 제품을 제작하지 않고서도 많은 실험을 할 수 있다. Visual Studio의 경우, rand()가 한번 호출될 때마다 0에서 32767까지의 정수를 같은 확률로 선택하여 반환한다. rand() 함수를 이용하여 0 또는 1 값을 무작위로 반환하는 함수 b_rand()를 작성하고 5번 호출하여 보자.

실행결과

```
C:\WINDOWS\system32\cmd.exe
1 1 0 0 1 0 0 0 0 0
```

HINT 난수의 시드도 현재 시각으로 설정하여 보자. <stdlib.h>를 포함시켜야 한다. % 연산자를 이용하라.

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

int b_rand();
int main(void)
{
    for (int i = 0; i < 10; i++)
        printf("%d ", b_rand());
    printf("\n");
    return 0;
}

int b_rand()
{
    return rand() % 2;
}
```

10 앞에서 작성한 b_rand() 함수를 이용하여 간단한 동전 던지기 게임을 시뮬레이션하여 보자. 컴퓨터가 동전을 던지고 사용자는 앞뒤를 말한다. 컴퓨터는 b_rand()를 이용하여 생성된 난수가 1이면 동전의 앞면으로 간주하고 0이면 동전의 뒷면으로 간주한다.

실행결과

```
C:\WINDOWS\system32\cmd.exe
앞면 또는 뒷면(1 또는 0):1
맞았습니다.
계속하시겠습니까?(y 또는 n):y
앞면 또는 뒷면(1 또는 0):0
틀렸습니다.
계속하시겠습니까?(y 또는 n):
```

HINT 반복 루프를 만들어서 사용자가 n을 입력할 때까지 반복한다.

```
int b_rand();
int main(void)
{
    int answer, coin;
    char c;

    while (1) {
        printf("앞면 또는 뒷면(1 또는 0):");
        scanf("%d", &answer);
        coin = b_rand();
        if (coin == answer)
            printf("맞았습니다.\n");
        else
            printf("틀렸습니다.\n");
        printf("계속하시겠습니까?(y 또는 n):");
        scanf(" %c", &c);
        if (c == 'n')
            break;
    }
    return 0;
}

int b_rand()
{
    return rand() % 2;
}
```

11 0.0부터 1.0까지의 난수를 반환하는 함수 f_rand()를 작성하고 5번 호출하여 본다.

실행결과



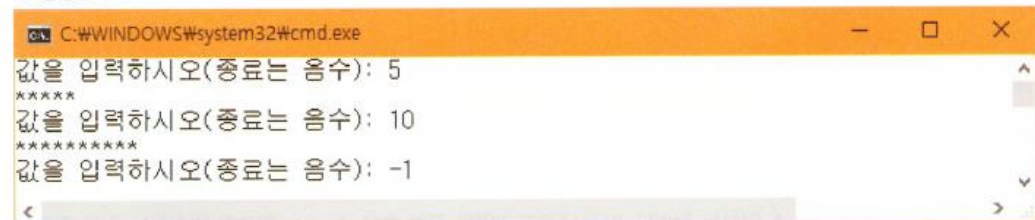
HINT 어떻게 하면 되는가? rand()/(double)RAND_MAX 수식을 분석하여 보자.

```
double f_rand();
int main(void)
{
    for (int i = 0; i < 5; i++)
        printf("%f ", f_rand());
    printf("\n");
    return 0;
}

double f_rand()
{
    return rand() / (double)RAND_MAX;
}
```

12 화면에 세로로 막대 그래프를 그리는 프로그램을 작성하여 보자. 인수의 값만큼의 별표 문자를 출력하는 함수 print_value(int n)을 작성하라. 반복적으로 사용자로부터 값을 입력받아서 print_value()를 호출하여 입력값만큼 막대를 그리는 프로그램을 완성하라. 사용자가 음수를 입력하면 반복을 중단하라.

실행결과



```
int main(void)
{
    int n;
    while (1) {
        printf("값을 입력하시오(종료는 음수): ");
        scanf("%d", &n);
        if (n < 0) break;
        print_value(n);
    }
    return 0;
}

void print_value(int n)
{
    for (int i = 0; i < n; i++)
        printf("*");
    printf("\n");
}
```

13 두 개의 정수 n, m을 입력받아서 n이 m의 배수이면 1을 반환하고 그렇지 않으면 0을 반환하는 함수 is_multiple (int n, int m)를 작성하고 테스트하여 보자.

실행결과

HINT 만약 n이 m의 배수이면 (n % m)이 0임을 이용하라.

```
int main(void)
{
    int x, y;

    printf("첫번째 정수를 입력하시오:");
    scanf("%d", &x);
    printf("두번째 정수를 입력하시오:");
    scanf("%d", &y);
    if (is_multiple(x, y) == 1)
        printf("%d는 %d의 배수입니다.\n", x, y);
    else
        printf("%d는 %d의 배수가 아닙니다.\n", x, y);
    return 0;
}

int is_multiple(int n, int m)
{
    if (n % m == 0) return 1;
    else return 0;
}
```

14 두 점사이의 거리를 계산하는 함수를 작성하여 보자. 2차원 공간에서 두 점 와 사이의 거리를 계산하는 get_distance(double x1, double y1, double x2, double y2)를 작성하시오. 다음과 같은 두 점 사이의 거리를 계산하는 공식을 사용하라. 제공근은 sqrt() 라이브러리 함수를 사용하라.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

실행결과

HINT sqrt() 함수를 이용하려면 <math.h>가 필요하다. 또 모든 수학 함수는 float가 아닌 double 형의 값을 받는 다는 점에 유의한다.

```
double get_distance(double x1, double y1, double x2, double y2)
{
    return sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
}

int main(void)
{
    double x1, y1, x2, y2;
    printf("첫번째 점의 좌표를 입력하시오:(x, y)");
    scanf("%lf %lf", &x1, &y1);
    printf("두번째 점의 좌표를 입력하시오:(x, y)");
    scanf("%lf %lf", &x2, &y2);

    printf("두점 사이의 거리는 %f입니다.\n", get_distance(x1, y1, x2, y2));
    return 0;
}
```

- 15 주어진 정수가 소수인지를 검사하는 함수 is_prime()을 작성하라. 이 함수를 이용하여 2부터 100 사이의 모든 소수를 출력하라.

실행결과

```
C:\WINDOWS\system32\cmd.exe
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

HINT 소수를 계산하는 알고리즘은 앞에서 많이 다룬바 있다. 2부터 자기 자신 사이에 약수가 하나라도 있으면 소수가 아니다.

```
int main(void)
{
    int i, j;

    for (i = 2; i < 100; i++) {
        if (is_prime(i) == 1)
            printf("%d ", i);
    }
    printf("\n");

    return 0;
}

int is_prime(int x)
{
    int i;
    for (i = 2; i < x; i++) {
        if (x % i == 0) return 0;
    }
    return 1;
}
```

- 16 오일러의 수 e는 자연 로그의 밑수로 사용된다. 이 값은 다음과 같은 식에 의하여 근사치를 구할 수 있다. 본문에 있는 팩토리얼 값을 계산하는 함수 factorial()을 호출하여서 오일러의 수를 계산하는 프로그램을 작성하라.

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \dots + \frac{1}{n!}$$

실행결과

```
C:\WINDOWS\system32\cmd.exe
어디까지 계산할까요: 30
오일러의 수는 2.718282입니다.
```

HINT 주의할 점은 팩토리얼 값은 매우 커질 수 있다는 점이다. 따라서 팩토리얼을 구한 후에 double 형으로 변환하는 것도 좋겠다. 또 팩토리얼 함수 안에서 result 변수의 자료형을 64비트 정수형인 long long으로 하는 것도 좋다.

```
long long factorial(int n)
{
    int i;
    long long result = 1;

    for (i = 1; i <= n; i++)
        result *= i;        // result = result * i

    return result;
}

int main(void)
{
    int i;
    double sum = 1.0;
    int n;

    printf("어디까지 계산할까요: ");
    scanf("%d", &n);

    for (i = 1; i <= n; i++)
        sum += 1.0 / factorial(i);
    printf("오일러의 수는 %f입니다. \n", sum);
    return 0;
}
```


- 17 두개의 부동 소수점 수가 근사적으로 같은 값이면 1를 반환하고 근사적으로 같지 않으면 0를 넘겨주는 함수 `f_equal(a, b)`을 작성하시오. 근사적으로 같다고 하는 것은 다음의 부등식을 만족하는 경우라고 가정한다.

$$\frac{|a-b|}{\min(|a|, |b|)} < e$$

부동 소수점 수의 절대값을 구하는 함수와 두수 중에서 최소값을 찾는 함수는 스스로 제작하여 사용하라. `e`는 상수로서 0.000001로 정의된다.

실행결과



```
C:\WINDOWS\system32\cmd.exe
실수를 입력하시오: 1.2345
실수를 입력하시오: 1.2346
두 개의 실수는 서로 다름
```

HINT `f_abs(double x)`, `f_min(double x, double y)`도 스스로 제작하여 보자.

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

int f_equal(double x, double y);
int main(void)
{
    double a, b;

    printf("실수를 입력하시오: ");
    scanf("%lf", &a);
    printf("실수를 입력하시오: ");
    scanf("%lf", &b);

    if (f_equal(a, b) == 1)
        printf("두 개의 실수는 서로 같음\n");
    else
        printf("두 개의 실수는 서로 다름\n");

    return 0;
}
```

```
double f_abs(double x)
{
    if (x > 0) return x;
    else return -x;
}

double f_min(double x, double y)
{
    if (x > y) return y;
    else return x;
}

int f_equal(double x, double y)
{
    double value;
    value = f_abs(x - y) / f_min(f_abs(x), f_abs(y));
    if (value < 0.000001) return 1;
    else return 0;
}
```

18 사용자로부터 2개의 숫자를 받아서 덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 연산의 결과를 계산해주는 프로그램을 작성해 보자. 다음과 같은 메뉴를 화면에 표시한다. 프로그램 작성시에 최대한 함수를 많이 사용해보자.

실행결과

```
C:\WINDOWS\system32\cmd.exe
=====
MENU
=====
1. 덧셈
2. 뺄셈
3. 곱셈
4. 나눗셈
5. 나머지
원하는 메뉴를 선택하시오(1-5):1
숫자 2개를 입력하시오: 10 20
연산결과 :30
계속하려면 y를 누르시오:
<
```

HINT 메뉴를 화면에 표시하는 함수도 만들어서 사용해보자.

```
void displaymenu() {
    printf("=====\n");
    printf("MENU\n");
    printf("=====\n");
    printf("1. 덧셈\n");
    printf("2. 뺄셈\n");
    printf("3. 곱셈\n");
    printf("4. 나눗셈\n");
    printf("5. 나머지\n");
}

int Add(int a, int b) {
    return(a + b);
}

int Subtract(int a, int b) {
    return(a - b);
}

int Multiply(int a, int b) {
    return(a * b);
}

float Divide(int a, int b) {
    return(a / b);
}

int Modulus(int a, int b) {
    return(a % b);
}
```

```
int main(int argc, char* argv[])
{
    //show menu
    displaymenu();
    int yourchoice;
    int a;
    int b;
    char confirm;
    do
    {
        printf("원하는 메뉴를 선택하시오(1-5):");
        scanf("%d:", &yourchoice);
        printf("숫자 2개를 입력하시오: ");
        scanf("%d %d", &a, &b);
        switch (yourchoice) {
            case 1:printf("연산결과 :%d", Add(a, b)); break;
            case 2:printf("연산결과 :%d", Subtract(a, b)); break;
            case 3:printf("연산결과 :%d", Multiply(a, b)); break;
            case 4:printf("연산결과 :%.2f", Divide(a, b)); break;
            case 5:printf("연산결과 :%d", Modulus(a, b)); break;
            default:printf("잘못된 연산\n");
        }

        printf("\n");
        printf("계속하려면 y를 누르시오: ");
        scanf("%s", &confirm);
    } while (confirm == 'y' || confirm == 'Y');
    system("PAUSE");
    return EXIT_SUCCESS;
}
```