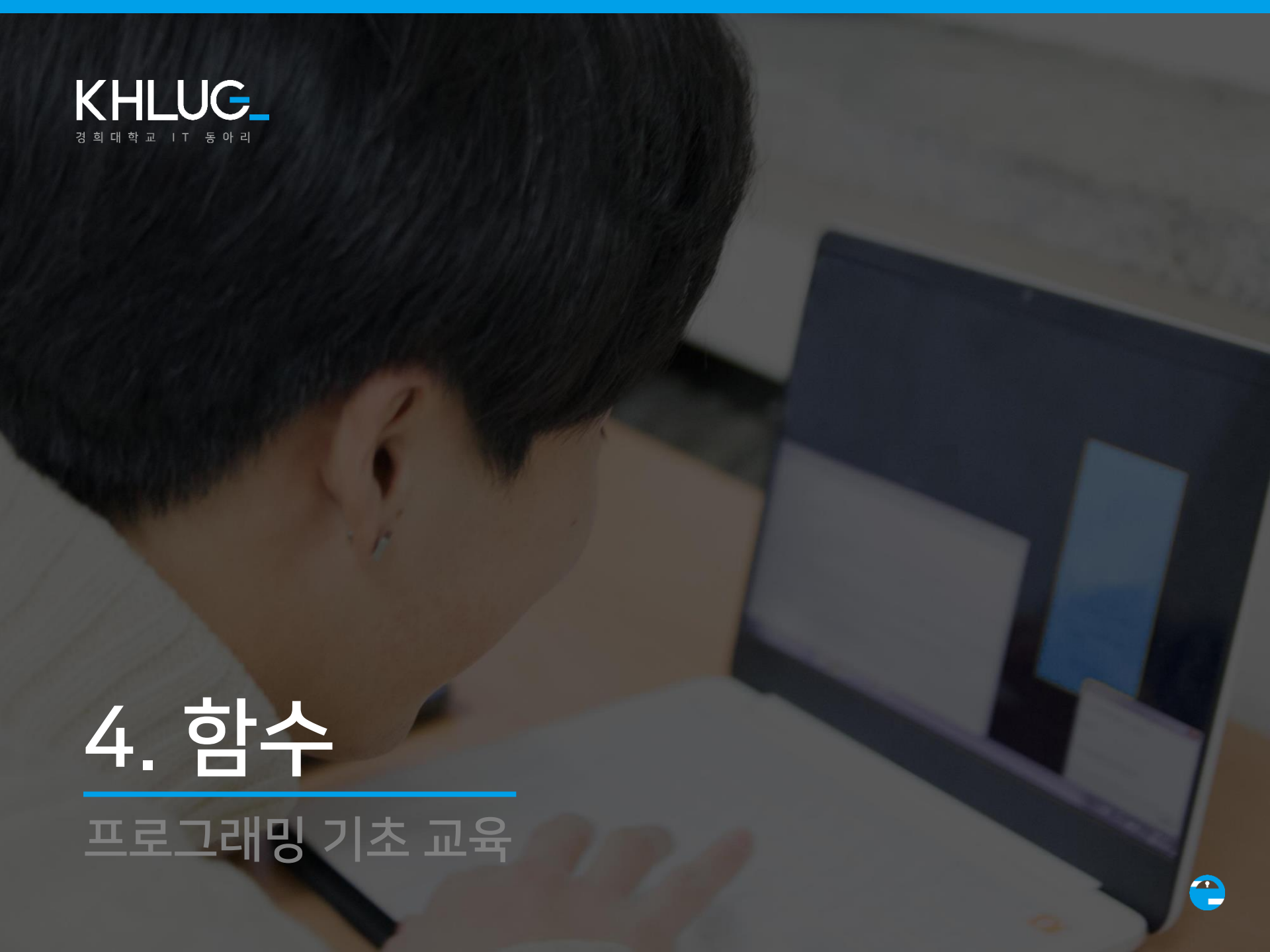
KHLUG

경희대학교 IT 동아리

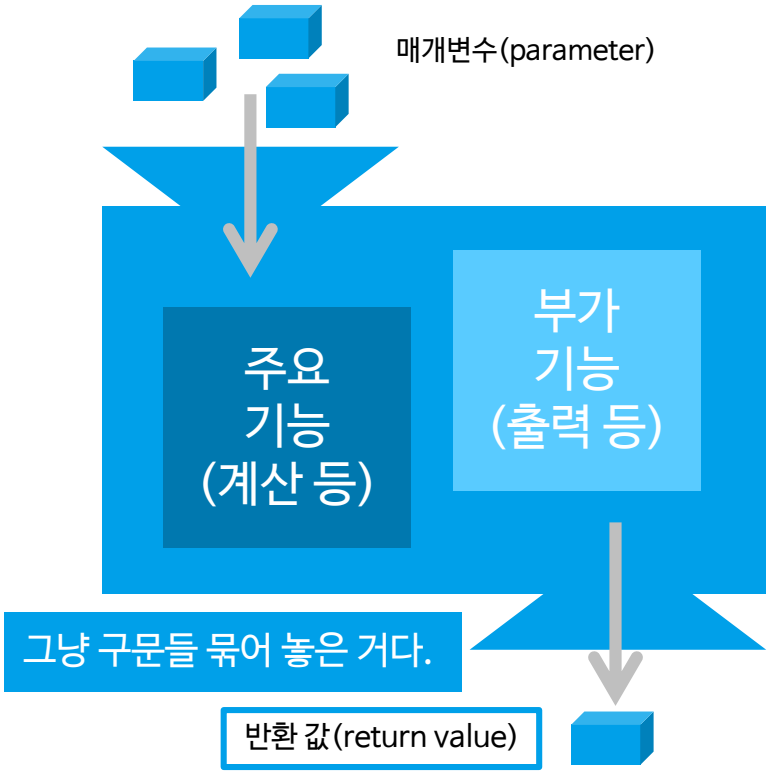


4. 함수

프로그래밍 기초 교육

함수

- 함수(function)는 몇 개의 변수(매개변수, parameter)를 넣어 적당히 계산해서 돌려주는(반환, return) 마법상자이다.



컴퓨터가 만든 함수가 아니라서 사용자 정의 함수라고 부른다.

```
int add(int a, int b) {
    int result = a + b;
    return result;
}

int main() {
    int n, m;
    cin >> n >> m;
    int sum = add(n, m);
    cout << sum << endl;

    return 0;
}
```

반환 값의 자료형은 반환형과 같아야 한다.

int main()도 함수이다. 마지막에 return 0;를 하면 프로그램이 끝날 때 운영체제에게 0(정상종료)으로 끝났음을 알린다.

반환_값의_자료형 함수_식별자 (매개변수_목록) { }

함수 호출

- 일반적으로 main() 위에서 함수의 프로토타입을 선언하고 아래에서 정의한다.

함수를 사용하기 전에 함수를 선언해야 한다. main() 아래에서 함수의 내용을 구현하기 위해 여기에 함수의 프로토타입을 선언했다.

함수이름(전달해줄 매개변수 목록) 구문을 통해 함수를 호출(call)한다.

여기에서 함수가 정의(구현)된다.

매개변수는 없을 수도 있다. (여긴 2개)

```
int add(int a, int b);

int main() {
    int n, m;
    cin >> n >> m;
    int sum = add(n, m);


    cout << sum << endl;
    return 0;
}

int add(int a, int b) {
    int result = a + b;
    return result;
}
```

return을 만나면 함수가 끝나고 반환 값을 돌려준다.

함수 호출

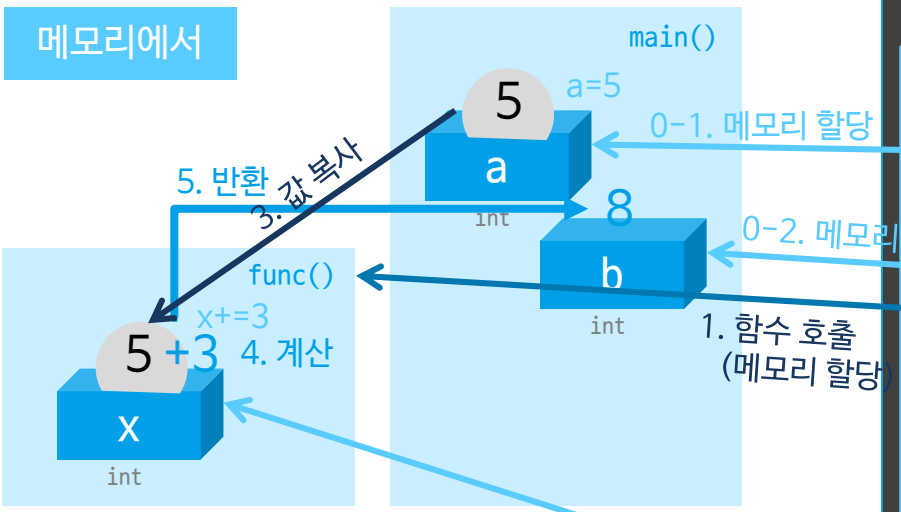
- 매개변수에는 변수나 값만 들어가는 것이 아니라 식, 함수 등 모든 종류의 표현이 들어갈 수 있다. (자료형만 맞춰주면 된다.)

```
8
10
11
```

```
int add(int a, int b);

int main() {
    int n = 3, m = 5;

    cout << add(3, m) << endl;
    cout << add(n + 2, m) << endl;
    cout << add(n, add(n,m)) << endl;

    return 0;
}

int add(int a, int b) {
    int result = a + b;
    return result;
}
```

Pass by Value (값으로 전달)

- 함수는 아래와 같이 작동한다.




```
int func(int x);

int main() {
    int a;
    a = 5;
    int b;
    b = func(a);
    // a==5
    // b==8
    return 0;
}

int func(int x) {
    x += 3;
    return x;
}
```

1. 함수 호출
2. 매개변수 생성 (메모리 할당)
3. 매개변수 값 복사
   (main()의 a의 값이 func()의 x에 복사된다.)
4. 함수의 기능 수행
5. 값 반환 (func()의 x 값이 main() b에 복사된다.)

함수 안의 x는 main()에서의 a의 값만 복사된
다른 변수이므로, x가 바뀌어도 a는 바뀌지 않는다.

void 함수

- 별도의 반환 값이 필요 없을 땐 반환형이 void인 함수를 사용할 수 있다.

주로 출력만 해야 하는 경우에 사용한다.

```
출력: 1
출력: 2
출력: 3
출력: 4
출력: 5
출력: 6
출력: 7
출력: 8
출력: 9
```

```
void printResult(int x) {
    cout << "출력: " << x << endl;
    return;
}

int main(){

    for(int i = 0; i <10; i++) {
        printResult(i);
    }

    return 0;
}
```

반환 값이 없으므로 return 뒤에 아무 것도 붙이지 않는다.

함수 호출을 반드시 변수에 대입할 필요는 없다. 대입 없이 변수만 사용하면 반환 값이 있더라도 그 결과값을 사용하지 않게 된다.

사용자 정의 함수 사용해보기

정수 n을 입력 받아 구구단 n단을 출력하는 함수를 작성하시오

숫자 3개를 입력 받아 가장 큰 값을 구하는 함수와 가장 작은 값을 구하는 함수를 작성하시오. 결과 값은 main()에서 출력되어야 한다.

숫자 3개를 입력 받아 평균을 구하는 함수를 작성하고, 받은 값을 출력하는 함수를 작성한다. 단, 두 함수는 main()에서만 호출되어야 한다.
두 함수를 사용하여 결과가 출력된 뒤에는 프로그램이 종료되지 않고 계속해서 새로운 입력을 받게끔 하는 프로그램을 작성하시오.

n의 제곱을 구하는 함수를 작성하시오. main()에서는 출력만 한다.

사용자 정의 함수 사용해보기

대문자를 입력하면 소문자를 출력하는 함수를 작성하시오.
입력 값은 대문자만 있는 것으로 한다. (다른 경우 고려할 필요 없음)

섭씨 온도를 입력하면 화씨 온도를 반환하는 함수 CelToFah()와,
화씨 온도를 입력하면 섭씨 온도를 반환하는 함수 FahToCel()를
정의하고 이 두 함수를 호출하여 출력하는 프로그램을 작성하시오.
이 때, 프로그램은 섭씨 한 번, 화씨 한 번 씩 입력을 받아야 하며,
입력(cin)과 출력(cout)은 각 함수가 아닌 main()에서 이루어져야 한다.

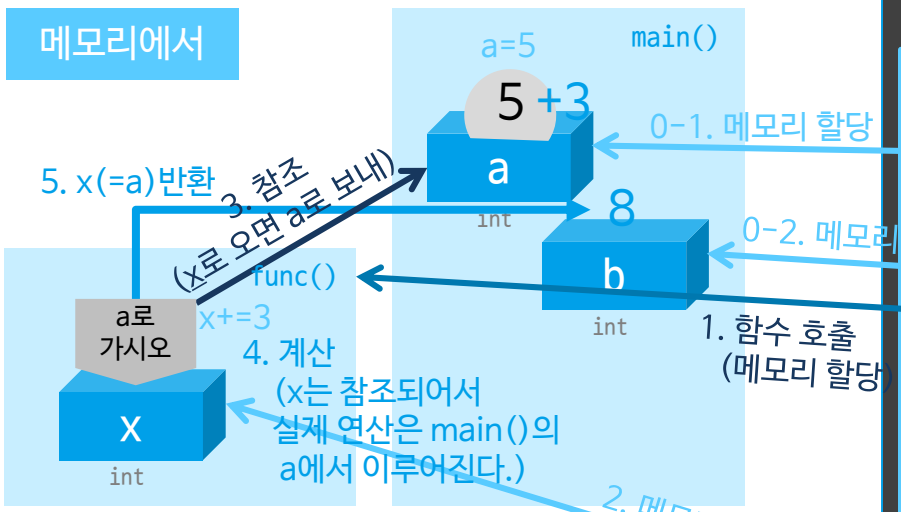
참고로 섭씨와 화씨간 온도변환의 공식은 다음과 같다.

$$화씨온도 = 1.8 \times 섭씨온도 + 32$$

Pass by Reference (참조 값으로 전달)

매개변수 자료형 뒤에 참조연산자 & (앰퍼샌드)를 붙이면 참조하여 가져온다.

- 매개변수를 원본처럼 사용해보자.




```
int func(int &x);

int main() {
    int a;
    a = 5;
    int b;
    b = func(a);
    // a==8
    // b==8
    return 0;
}

int func(int &x) {
    x += 3;
    return x;
}
```

1. 함수 호출
2. 매개변수 생성 (메모리 할당)
3. 매개변수가 원본을 참조하도록 만듦
4. 함수의 기능 수행
   이 때, 매개변수에서 이루어지는 연산은
   참조 중인 원본에 가서 이루어진다
5. 값 반환 (func()의 x 값이 main() b에 복사된다.)

함수의 x는 main()에서의 a와 같은 주소(address)를 갖기 때문에 x가 바뀌면 a도 동일하게 바뀐다.

자리 바꾸기

- 두 변수의 값을 서로 바꾸는 함수를 만들어보자.

```
3 5
3 5
```

안 바뀐다.

앞에서 말했지만…
함수로 전달해준 매개변수는 따로 복사되어 사용되기 때문에 함수 안에서 매개변수에게 일어난 일은 원본 변수에 영향을 미치지 않는다.

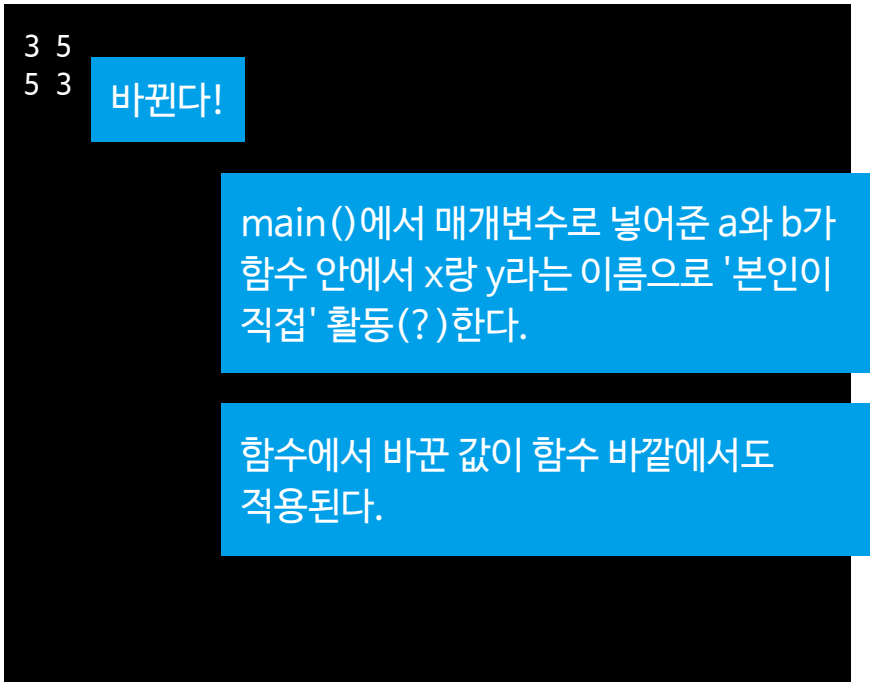
아니 근데 값 바꿀 변수는 두 개인데 반환 값은 한 개까지 밖에 안 되는데 어떡하지…

```
void exchange(int x, int y);

int main(){
    int a = 3, b = 5;
    cout << a << " " << b << endl;
    exchange(a, b);
    cout << a << " " << b << endl;
    return 0;
}

void exchange(int x, int y) {
    int hand = x;
    x = y;
    y = hand;
    return;
}
```

x = y에서 x의 값이 y의 값으로 덮어씌워지기 때문에 x의 값을 미리 다른 곳에 저장해뒀다가 꺼내 써야 한다.

자리 바꾸기

- 아까 배운 걸 써먹자
- Pass by Value에서와 달리 Pass by Reference는 여러 개의 변수를 수정해서 보낼 수 있다.




```
void exchange(int &x, int &y);

int main(){
    int a = 3, b = 5;
    cout << a << " " << b << endl;
    exchange(a, b);
    cout << a << " " << b << endl;
    return 0;
}

void exchange(int &x, int &y) {
    int hand = x;
    x = y;
    y = hand;
    return;
}
```

기본값 (기본 매개변수)

- 함수를 선언할 때, 매개변수로 값이 넘어가지 않았을 때 기본적으로 사용할 값을 정해줄 수 있다.

기본 매개변수는 끝에서부터 설정할 수 있다.
(int num1, int num2 = 0, int num3) 처럼 중간에 사용하는 것은 안 된다.

```
void printInt(int num = 0);

int main(){
    cout << "지르고 본다.\n";
    printInt();

    cout << "\n뭐라도 넣어준다.\n";
    printInt(405);

    return 0;
}

void printInt(int num) {
    cout << "출력: " << endl;
    return;
}
```

'\n'은 줄바꿈 문자이다.
역슬래시는 한국어 키보드에서 '₩'으로 보인다.

```
지르고 본다.
출력: 0

뭐라도 넣어준다.
출력: 405
```

아무 것도 넣지 않았더니 기본 값인 0이 사용된다.

함수 만들어보기

피연산자 두 개를 받아 나누어 그 몫과 나머지를 보내주는 함수를 작성하고, 그 결과를 출력하는 프로그램을 작성하시오.

힌트: 아래와 같은 프로토타입을 사용해보자.

```
void divide(int divnd, int divsr, int &quot, int &rem);
```

두 개의 정수와 연산자(+, -, *, /, %)를 입력 받아서 계산하는 프로그램을 작성하시오.
지정된 연산자 외의 글자가 들어오면 프로그램을 종료시킨다.
계산기의 기능을 하는 함수와 각 연산은 별개의 함수로 분리되어 있어야 하며, 결과값을 출력한 뒤에도 반복해서 입력을 받을 수 있어야 한다. (0으로 나누는 등 연산이 되지 않는 경우에 대해서는 오류메세지를 출력한다.)

힌트: main()의 끝은 return 0이다. 함수 안에서도 함수를 호출할 수 있다.

```
void calculator();                          double add(double num1, double num2);
double subtract(double num1, double num2); long divide(long num1, long num2);
double multiply(double num1, double num2); long modular(long num1, long num2);
```

라이브러리

- 미리 만들어놓은 기능들의 집합이다.
- #include <iostream> 과 같이 미리 만들어진 라이브러리를 첨부할 수 있다.

```
#include <iostream>
using namespace std;

int main(){
    int a;
    cin >> a;
    cout << a << endl;
    return 0;
}
```

(cin과 cout만 본다면)
컴파일 될 때 링커(linker)에 의해 이렇게 추가된다.

```
/* cin 선언 */
/* cout 선언 */
/* 기타 여러 함수들 선언 */

int main(){
    int a;
    cin >> a;
    cout << a << endl;
    return 0;
}

/* cin 정의 (구현) */
/* cout 정의 (구현) */
/* 기타 함수들 정의 (구현) */
```

수학 <cstdlib> <cmath>

#include ⟨cstdlib⟩

```
int abs(int number);          // 절대값(absolute number)
long labs(long number);       // long을 위한 abs 함수
double fabs(double number);   // 실수(float, double)을 위한 abs 함수
```

#include ⟨cmath⟩

```
double ceil(double number);  // 1 단위 올림
double floor(double number); // 1 단위 내림
double pow(double x, double y); // x의 y제곱 (x^y)
doudle sqrt(double number);  // 제곱근
```

더 많은 함수는 http://www.cplusplus.com/reference/cmath/에서 확인할 수 있다.

수학 <cstdlib> <cmath>

```
ceil    : 3
floor   : 5
abs     : 120
pow     : 1024
sqrt    : 1.73205
```

```
#include <iostream>
#include <cstdlib>
#include <cmath>
using namespace std;

int main(){
    cout << "ceil \t: "
        << ceil(2.4) << endl;
    cout << "floor\t: "
        << floor(5.6) << endl;
    cout << "abs  \t: "
        << abs(-120) << endl;
    cout << "pow  \t: "
        << pow(2.0, 10) << endl;
    cout << "sqrt \t: "
        << sqrt(3.0) << endl;

    return 0;
}
```

랜덤(random, 임의)

srand(int)

- 랜덤 나무 열매에 씨를 뿌린다. (시드 값 설정)
- 뿌려진 씨가 같으면 똑같은 순서로 랜덤 값이 출력된다.
- 매 초마다 다른 씨를 뿌리고 싶으면 매개변수로 time(NULL)을 넣어보자.

rand()

- 0부터 RAND_MAX(Visual C++에서는 32767, GCC 등에서는 $2^{32}-1$) 사이의 정수 중 하나를 임의로 반환한다.

```
17710
14870
14692

6
11
```

10으로 나눈 나머지는 0~9이다.
즉, 0~9 사이의 임의의 수를 출력한다.

0~9에 3을 더하면 3~12이다.
즉, 3~12 사이의 임의의 수를 출력한다.

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main(){
    srand(time(NULL));

    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << endl;
    cout << rand() % 10 << endl;
    cout << rand() % 10 + 3 << endl;
    return 0;
}
```

srand()와 rand()는 〈cstdlib〉에,
time()은 〈ctime〉에 있다.

라이브러리와 함수 사용해보기

여태까지 만든 계산기 프로그램에 절대값, 올림, 내림, 제곱, 제곱근 연산도 추가한다.

컴퓨터와 가위바위보 하는 프로그램을 작성하시오.
사용자는 가위, 바위, 보를 낼 수 있으며 입력 값은 정수, 문자 등 어느 것이든 상관 없다.
컴퓨터 역시 임의의 수를 내며, 사용자와 컴퓨터가 낸 수에 따라 결과가 정해진다.
가위는 보에 이기며, 보는 바위에 이기고, 바위는 가위에 이긴다.
같은 수를 낸 경우 무승부로 처리하며, 무승부일 경우 한 번 더 입력을 받는다.

문방구 프로그램을 작성하시오.
문방구에는 다음과 같은 물품을 판다
: 연필, 볼펜, 지우개, 자, 풀, 테이프
물품의 가격은 자유롭게 설정하며,
손님의 초기 자금은 1만 원이다.
손님은 물품을 선택하고 구입할 개수를
정하며, 돈이 부족한 경우 구입할 수 없다.

결과 예시

```
구매할 물건을 입력하세요.
> 연필
연필은 하나에 500원입니다. 몇 개 구입하시겠습니까?
> 12
합계 6000원입니다. 4000원 남았습니다.
구매할 물건을 입력하세요.
> 자
연필은 하나에 1000원입니다. 몇 개 구입하시겠습니까?
> 5
합계 5000원입니다. 1000원 부족합니다. 4000원 남았습니다.
구매할 물건을 입력하세요.
> 나가기
없는 물품입니다. 오늘은 여기까지 영업합니다. 안녕히 가세요!
```

2017.03.30. 프로그래밍 기초 (2017-1)
with D.com