필수

참조 (reference)

참조 :
    변수 선언 시 변수명 앞에 & 기호.
    새로 생기는 것이 아님! 새 이름을 부여하는 것!

함수 매개변수로 사용시 call by reference :
    pointer 형을 사용하지 않고도 caller와 callee가 변수 공유 가능.

```
int  CalcDevide( int *pResult, int a, int b );
int  CalcDevide( int &result, int a, int b );
```

: 코드 간결, 변수 복사 시간 아낌.

필수

연습문제 3.17 (reference)

[문제] 아래와 같은 결과가 나오도록 MaxMin() 함수를 작성하시오.

```
void MaxMin(int &x, int &y) {
    int temp;
    if (x < y) {
        temp = x;
        x = y;
        y = temp;
    }
}
int main() {
    int min, max;
    cout << "2개의 정수 입력 : ";
    cin >> min >> max;
    MaxMin(max, min);
    cout << "최대값 : " << max << endl;
    cout << "최소값 : " << min << endl;
    return 0;
}
```

[결과]
2개의 정수 입력 : 3 4
최대값 : 4
최소값 : 3

객체지향 프로그래밍의 3요소

(1)추상화(data abstraction)

- 대상을 개념화 하여 새로운 타입을 만듦. (abstract data type)
- 클래스(class) : 데이터(멤버 변수)와 이를 다루는 함수(멤버 함수)를 묶음.
- 클래스(class) - 객체 (Object) 구분하기.

(2)상속(inheritance)

- 기존 class 확장 (class 선언문에 : public baseClass 식으로 표현됨)

(3)다형성(polymorphism)

- Derived class 에서 새로 정의 한 함수는 base class pointer로 접근하더라도 derived class 함수가 호출되도록 해 줌. (가상 함수)

위를 이용하여 "일반화 프로그래밍" (template 개념. 함수나 class가 특정 타입에 구속되지 않고 다양한 타입을 위해 사용될 수 있게 하는 방법) 지원.

클래스와 객체

```
#include <iostream>
using namespace std;
class Car
{
public:
    int m_color;
    int m_cc;
    int m_speed;
    void Accelerate() { m_speed++; }
    void Stop() {}
    void TurnOn() {}
};
int main() {
    Car car1, car2, *pCar = &car1;
    car1.m_speed = car2.m_speed = 0;
    pCar->Accelerate();
    cout << car1.m_speed << endl;
    return 0;
}
```

데이터 은닉!
Class에서는 기본적으로 private 이므로
이것이 없으면 외부에서 멤버 접근 불가능.

이 class의 객체가 선언될 때마다 멤버변수가 새로 생김
.("데이터", "스택", "힙" 영역 중 한 곳에 위치.)

이 class의 객체를 여러 개 선언해도 멤버함수는 메모리 상에 1개만 존재.
("코드"부에 속하므로)

[syntax 살펴보기]
class 객체 선언.
class 멤버에 접근.
class pointer 이용한 멤버 접근.

constructor와 destructor

[Note!] 매개변수 있는 constructor를 추가하면 default constructor는 효과 없게 되므로, 이 때 매개 변수 없는 constructor가 필요하면 명시적으로 선언해 줘야 함.

- constructor
  - 객체 생성시 호출됨. (초기화 문장에 따라 선택됨)
  - 클래스 이름과 같다.
  - 반환형이 없다.
  - 매개변수를 자유롭게(default 값 사용 가능) 줄 수 있고 overloading을 통해 여러 개 만들 수 있다.
  - 매개변수 없이 빈 형태를 default constructor라 한다.
- destructor
  - 객체 소멸시 자동 호출 됨.
  - 클래스 이름 앞에 '~' 문자를 붙인다.
  - 반환형이 없다.
  - 매개변수가 없다. 그러므로 단 하나의 함수만 존재.
  - 매개변수 없이 빈 형태를 default destructor라 한다.
  - [사용 예] 객체 사용 중 new 받았던 메모리를 delete