C++ 핵심 정리노트 (7장 ~ 13장, 예제 포함)

7장: 프렌드와 연산자 중복

◇ friend 함수

- 외부 함수가 클래스의 private/protected 멤버에 접근 가능하게 함.

```
class Rect {
private:
    int width, height;
    friend bool isEqual(const Rect& a, const Rect& b);
public:
    Rect(int w, int h) : width(w), height(h) {}
};

bool isEqual(const Rect& a, const Rect& b) {
    return a.width == b.width && a.height == b.height;
}
```

◇ 연산자 중복

이항 연산자

```
class Point {
    int x, y;
public:
    Point(int x = 0, int y = 0) : x(x), y(y) {}
    Point operator+(const Point& p) {
        return Point(x + p.x, y + p.y);
    }
};
```

단항 연산자 (전위/후위)

```
class Counter {
    int value;
public:
    Counter(int v = 0) : value(v) {}
    Counter operator++() { value++; return *this; }       // 전위
    Counter operator++(int) { Counter temp = *this; value++; return temp; } // 후
위
};
```

8장: 상속 (Inheritance)

◇ 기본 상속 구조

```
class Shape {
protected:
    string color;
public:
    void setColor(string c) { color = c; }
};

class Circle : public Shape {
    int radius;
public:
    Circle(int r) : radius(r) {}
};
```

◇ 생성자/소멸자 호출 순서

```
class Base {
public:
    Base() { cout << "Base 생성자" << endl; }
    ~Base() { cout << "Base 소멸자" << endl; }
};

class Derived : public Base {
public:
    Derived() { cout << "Derived 생성자" << endl; }
    ~Derived() { cout << "Derived 소멸자" << endl; }
};
```

9장: 가상 함수와 오버라이딩

◇ 가상 함수 및 다형성

```
class Animal {
public:
    virtual void speak() { cout << "동물 소리" << endl; }
};

class Dog : public Animal {
public:
    void speak() override { cout << "멍멍" << endl; }
};

Animal* a = new Dog();
a->speak(); // 멍멍
```

◇ 추상 클래스

```
class Shape {
public:
    virtual void draw() = 0; // 순수가상함수
};

class Rectangle : public Shape {
public:
    void draw() override { cout << "사각형 그리기" << endl; }
};
```

10장: 템플릿과 STL

◇ 함수 템플릿

```
template <typename T>
T add(T a, T b) {
    return a + b;
}
```

◇ 클래스 템플릿

```
template <typename T>
class MyArray {
    T data[100];
public:
    void set(int index, T value) { data[index] = value; }
    T get(int index) { return data[index]; }
};
```

◇ STL 컨테이너 사용 예시

```
#include <vector>
#include <algorithm>
vector<int> v = {5, 2, 9, 1};
sort(v.begin(), v.end());
```

11장: 입출력 스트림과 포맷

◇ put(), get(), getline()

```
char ch;
cout.put('A');
cin.get(ch);
char line[100];
cin.getline(line, 100);
```

◇ 포맷 조작자

```
#include <iomanip>
cout << setw(10) << setfill('.') << 25 << endl;
cout << hex << showbase << 255 << endl; // 0xff
```

13장: 예외 처리

◇ try-throw-catch 기본 구조

```
try {
    if (n == 0) throw "0으로 나눌 수 없습니다.";
    int avg = sum / n;
} catch(const char* msg) {
    cout << msg << endl;
}
```

◇ 사용자 정의 예외 클래스

```
class MyException {
    string msg;
public:
    MyException(string m) : msg(m) {}
    void print() { cout << msg << endl; }
};

class DivideByZero : public MyException {
public:
    DivideByZero() : MyException("0으로 나눌 수 없음") {}
};

try {
    throw DivideByZero();
} catch(MyException& e) {
    e.print();
}
```