항목 24: 타입 변환이 모든 매개변수에 대해 적용되어야 한다면 non-member 함수를 선언하자.

유리수를 나타내는 클래스를 만들었습니다. 이때 덧셈이나 곱셈 등의 수치 연산을 기본으로 지원하는 함수를 만들고자 합니다. 멤버 함수로 만들까요? non-member non-friend 함수가 좋을 까요?

우선 멤버 함수로 만들어 봅니다.

class Rational {

public:

const Rational operator\*(const Rational& rhs) const;

};

Rational oneHalf(1, 2);

Rational result = oneHalf \* 2; // 가능

Rational result = 2 \* oneHalf; // 에러!

교환 법칙이 성립하지 않습니다. 원인은 간단합니다. 2.operator \* (oneHalf)은 가능하지 않기 때문입니다. 클래스도 아닌 숫자에 불과한 2가 .operator\*을 호출 할 수 있을 리가 없습니다.

그렇다면 어째서 oneHalf.operator \* (2); 는 가능했을까요? 이유는 컴파일러의 암시적 타입 변환 때문 입니다. **컴파일러는 int를 이용하면 Rational 생성자를 통해 Rational객체를 만들 수 있다는 것을 이미 알고 있습니다.**

oneHalf.operator \* (2);을 컴파일러 관점에서 풀어서 다시 쓰면 oneHalf.operator \* (Rational temp(2)) 라는 것 입니다. (물론 생성자가 명시호출이 아니어야 합니다.)

그런데 2 \* oneHalf도 비명시호출 생성자를 통해 (Rational temp(2)) \* oneHalf가 되어 잘 되었으면 좋겠지만 왜 안 되는 걸까요? 암시적 타입 변환에 대해 매개변수가 사용되려면 **매개변수 리스트**에 들어 있어야 합니다. 호출되는 멤버 함수를 갖고 있는 객체에 해당하는 암시적 매개변수에는 암시적 변환이 먹히지 않습니다.

그럼 어떻게 해야 교환법칙이 성립하게 만들 수 있을까요? **바로 operator\*를 non-member 함수로 만들어서, 컴파일러 쪽에서 모든 인자에 대해 암시적 타입 변환을 수행하도록 내버려 두는 것 입니다.**

다시 처음으로 돌아와서 이제 non-member 함수로 만들어 봅니다.

class Rational {}; // 클래스

const Rational operator \* (const Rational& lhs, const Rational& rhs) { … }

이제 result = 2 \* oneFourth; 이것도 됩니다.