항목 23. 멤버 함수보다는 **non-member non-friend** 함수와 더 가까워지자

클래스 안에 여러 가지 함수가 있습니다. 이 함수들 각각 호출 하는 것이 불편해서 한번에 호출 해주는 편의 함수를 만들었습니다.

이때 한번에 호출 해주는 함수는 멤버 함수로 만들까요? 아니면 non-member 함수로 만들까요?

객체 지향 적으로 생각해 봅시다. 가능한 데이터를 캡슐화 하는 것 입니다.

**캡슐화**

1. **외부에서 볼 수 없게 합니다. 따라서 밖에서 볼 수 있는 것들이 줄어듭니다. 그러므로 외부에서 변경이 되어도 코드가 유지 됩니다.**
2. **어떤 데이터를 접근하는 함수가 많으면 그 데이터의 캡슐화 정도는 낮습니다.**

다시 질문으로 돌아와서 어느 쪽이 더 캡슐화에 기여하는지에 대한 답은 non-member 함수 입니다. non-member 함수는 class의 private 멤버 부분을 접근할 수 있는 함수의 개수를 늘리지 않기 때문입니다.

주의 할 점은 non-friend 함수에만 적용 된다는 것 입니다. friend 함수는 private 멤버에 대한 접근권한이 있기 때문입니다.

편의 함수를 확장하기 쉽고 컴파일 의존성을 고민할 이유도 없으며 캡슐화까지 챙길 수 있는 방법이 있습니다.

* **하나의 헤더를 만들고 핵심 기능들이 있는 클래스를 네임스페이스로 감싸 줍니다.**
* **한 가지 편의 기능을 제공하는 헤더를 하나 만들고 위와 동일한 네임스페이스를 사용해서 편의 함수를 만듭니다.**
* **또 다른 편의 함수를 만들 때는 헤더를 새롭게 만들고 위와 동일한 네임스페이스를 감싸 편의 함수를 만들어 줍니다.**

핵심헤더.h

namespace { class 핵심 기능 함수 들 }

편의헤더.h

namespace{ 편의 함수 }

… 새로 편의 함수가 필요할 때마다 추가.