[고급] 오픈소스 분석을 위한

고급 C++

Final Project

❑ 과제 설명

총 3개의 프로그래밍 과제 입니다. 일부 문제만 해결하시면 부분 점수가 부여 됩니다.

❑ 제출 기한

과정 종료일까지(Syllabus 참고)

1. 아래 코드는 unique_ptr 의 삭제자를 변경해야 합니다.

   "람다 표현식" 을 사용해서 삭제자를 변경해 보세요.

   단, C++17 환경에서 컴파일이 가능해야 합니다.

```
#include <memory>

int main()
{
    std::unique_ptr<int> up1(static_cast<int*>(malloc(sizeof(int)*20)));
}
```

2. 아래 코드는 "reference counting" 을 사용하는 Label 클래스 입니다.

main 함수가 실행될수 있도록, [] 연산자를 제공해 주세요

```
#include <iostream>

class Label
{
    char* text;
    std::size_t size;
    int* ref;
public:
    Label(const char* s) : ref(new int(1))
    {
        size = strlen(s);
        text = new char[size + 1];
        strcpy_s(text, size + 1, s);

        ref = new int(1);
    }
    Label(const Label& other) : text(other.text), size(other.size),
ref(other.ref)
    {
        ++(*ref);
    }
    ~Label()
    {
        if (--(*ref) == 0)
        {
            delete ref;
            delete text;
        }
    }
    void print() const { std::cout << text << std::endl; }
};

int main()
{
    Label lb1("hello");
    Label lb2 = lb1;

    // 과제 : 아래 처럼 동작하도록 [] 연산자를 재정의해 보세요.
    char c = lb1[0]; // lb1, lb2 는 계속 자원을 공유 해야 합니다.
    lb1[0] = 'A';    // lb1, lb2 의 자원을 분리 되어야 합니다.

    lb1.print(); // "Aello"
    lb2.print(); // "hello"
}
```

3. 아래 코드가 정상 실행될수 있도록 "drop_view" 를 완성해 주세요

   단, C++20에서 표준에 추가된 view_interface 로부터 상속 받아서 만들어 주세요

```
#include <vector>
#include <iostream>
#include <ranges>

int main()
{
    std::vector v = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

    std::ranges::reverse_view rv(v);
    drop_view dv(rv, 3);

    for (auto e : dv)
        std::cout << e << ", ";
    // 실행결과 : 7, 6, 5, 4, 3, 2, 1
}
```