객체지향프로그래밍 - 과제 #1

- 202184021 소프트웨어전공 박지민

1. 문제 정의

1. Add(더하기), Sub(빼기), Mul(곱하기), Div(나누기)를 수행하는 클래스를 정의한다
2. 각 클래스는 다음과 같은 멤버를 가진다.
    - **멤버 변수** : `a(int)`, `b(int)`
    - **멤버 함수** : `void setValue(int x, int y)`, `int calculate()`
3. `main()` : 프로그램 동작 과정
    - `Add`, `Sub`, `Mul`, `Div` 클래스 타입의 객체 `a`, `s`, `m`, `d`를 생성
    - 키보드로부터 두 개의 정수와 연산자 입력 받음 (cin)
    - 입력받은 연산자에 따른 객체의 `setValue` 함수 호출
    - 입력받은 연산자에 따른 객체의 `calculate` 함수 호출 후, 결과 출력
    - 프로그램 무한 반복

2. 문제 해결 방법

1. 클래스 정의
    - `Add`, `Sub`, `Mul`, `Div` 클래스를 정의
    - 각 클래스는 `a`, `b` 멤버 변수와 `setValue`, `calculate` 멤버 함수를 가짐
2. 각 클래스의 함수 정의
    - `setValue` : 멤버 변수 `a`, `b`에 인자로 받은 값을 저장
    - `calculate` : `a`, `b` 멤버 변수에 저장된 값을 이용하여 각 클래스의 역할에 맞는 기능 구현
3. `main()` 함수 구현
    - `Add`, `Sub`, `Mul`, `Div` 클래스 타입의 객체 `a`, `s`, `m`, `d`를 생성
    - **while**문을 이용하여 무한 반복
        - 키보드로부터 두 개의 정수와 연산자 입력 받음 (cin)
        - 입력받은 연산자에 따른 객체의 `setValue` 함수 호출
        - 입력받은 연산자에 따른 객체의 `calculate` 함수 호출 후 `result`에 저장
        - cout을 이용하여 `result` 출력
        - 다시 반복

3. 아이디어 평가

- 객체지향 프로그래밍의 특징인 **캡슐화**를 적용하여 멤버 변수(a, b)를 외부로 노출하지 않고 멤버 함수를 통해 접근할 수 있도록 구현.
- 단, 객체지향 프로그래밍의 특징인 **상속**과 **다형성**을 적용하지 않아, 중복된 코드가 발생할 수 있음.
    - 이 문제는 추후 `Operator`라는 부모 클래스를 정의하고, 이를 상속받아 더 간결한 코드로 구현할 수 있음.
    - 이때, `Operator` 클래스는 `a`, `b` 멤버 변수의 접근 제어자를 `protected`로 설정하여 상속받은 클래스에서 접근할 수 있도록 함.
    - 또한, `setValue` 함수의 내용은 모두 동일하므로 `Operator` 클래스에서 정의하여 중복을 제거할 수 있음.
    - 다만, `calculate` 함수는 내용이 각 클래스마다 다르므로, `virtual` 키워드를 이용하여 가상 함수 선언 후, 각 클래스에서 `override`하여 구현할 수 있음.

> TODO: C++의 **상속**과 **다형성**을 학습 후 `Operator` 클래스를 정의하여 구현해보기

4. 문제를 해결한 키 아이디어 또는 알고리즘 설명

- **캡슐화** : 멤버 변수(a, b)를 외부로 노출하지 않고 **멤버 함수**를 통해 접근할 수 있도록 구현
- **함수 분리** : `setValue`, `calculate` 함수를 정의하여 코드를 분리
- **무한 루프** : **while**문을 이용하여 무한 반복
- **입력 처리** : cin을 이용하여 키보드로부터 입력받은 연산자를 **switch**문을 통해 처리