문제 정의

5 번 과제와 동일한 기능을 가지는 그래픽 편집기는 vector<Shape*> v;를 사용하여 구성하기.

핵심 알고리즘

문제에 나온 조건대로 Shape, Circle, Rect, Line 의 선언부와 구현부를 구현하고, vector<Shape*> v 를 사용한다.

vector<Shape*> v 에 insert 로 생성되는 Shape 객체를 넣는다.

삭제와 모두보기 기능은 iterator 를 사용해서 구현한다.

알고리즘 평가

그 전 과제에서는 next 를 사용해서 각각 Shape 객체를 연결시켰는데, 이번에는 문제에 나온 대로 vector 를 사용해서 Shape 객체를 vector 에 넣어주었다. 그렇기에 HW#5 에서 문제에 나온 대로 Shape, Circle, Rect, Line 의 선언부와 구현부를 고쳤고, main 과 UI 의 선언부와 구현부도 그대로 사용하였다. GraphicEditor 의 구현부 내용만 변경하였는데, vector 의 push_back, erase 를 사용했고, iterator 를 사용해 반복문을 돌려 기능을 구현했다. Vector 를 사용해서 이미 구현된 기능을 사용하니 저번에 구현했던 과제보다는 더욱 깔끔한 코드가 완성되었다고 생각한다.

알고리즘 구현

앞서 설명했던 대로 그래픽 에디터의 구현부와 문제에 맞춰서 변경된 Shape, Circle, Rect, Line 클래스의 선언부 구현부를 제외하고는 모든 부분이 hw#5 와 동일하게 구성되었다.

간단하게 Shape 클래스는 부모 클래스이며, 추상 클래스로 이루어져 있고 가상함수 draw() 와 draw 를 실행시키는 paint() 함수가 존재한다.

Shape 클래스를 상속받는 Shape, Circle, Rect, Line 클래스들은 public 으로 Shape 를 상속받고 오버라이딩된 draw() 함수가 존재하며 각각 자신 클래스의 종류를 출력하는 기능을 한다.

UI 클래스는 static 전역변수로 선언된 modeNum, insertNum, delNum 변수가 존재한다. 이 변수들은 각각 입력을 받을 때 사용되는 변수들이다. 함수로는 각각 상황에 맞는 출력문이 나오도록 하는 함수들이 구성되어 있다.

마지막으로 그래픽에디터는 vector<shape*> v 와 vector<Shape*>::iterator it 를 private 로 선언하고, 기본 기능의 생성자와 그래픽 에디터 객체가 삭제될 때 vector 에 남아있는 모든 객체를 삭제하는 기능을 하는 소멸자, 객체를 추가하는 기능의 insertShape() 함수와

삭제하는 기능을 가지는 deleteShape() 함수, 그리고 그래픽 에디터의 동작을 담당하는 start() 함수로 이루어져 있다.

소멸자는 iterator 를 사용해 vector 의 begin 부터 end 까지의 반복문을 돌려 모든 객체를 erase 를 사용해 삭제하는 기능을 한다.

insertShape(Shape *shape) 함수는 vector 의 push_back 기능을 사용해 매개변수로 받는 shape 객체를 vector 에 넣는 기능을 한다.

deleteShape(int index) 함수는 반복문을 돌며 iterator 와 at 를 사용해서 it 의 값이 vector 의 index 값과 같다면 해당 iterator 의 값을 v.erase 를 사용해 삭제해 주는 기능을 한다.

showAllShapes() 함수는 소멸사와 동일한 방식으로 반복문을 돌며 iterator 에 값을 Shape 타입의 shape 포인터에 넣어주고 그 객체에 해당하는 paint() 함수를 실행시켜준다.
반복문을 돌면서 vector 내에 있는 모든 객체를 출력해주는 기능을 한다.

Start() 함수는 반복문과 조건문을 활용해 그래픽 에디터의 구동을 담당하는 기능을 한다.

Main()에서는 그래픽에디터 객체를 생성하고, 그래픽에디터의 start 함수를 실행한다.