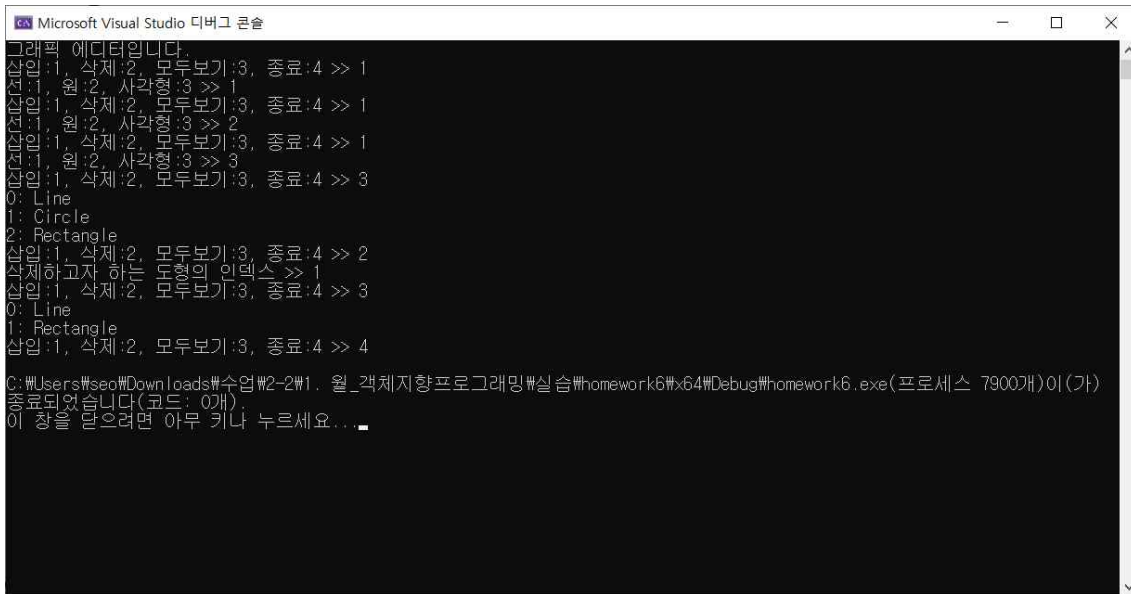


1. 소스 파일

깃허브 업로드 완료

2. 소스 수행 화면



```
Microsoft Visual Studio 디버그 콘솔
그래픽 에디터입니다.
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 1
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 2
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 1
선:1, 원:2, 사각형:3 >> 3
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 3
0: Line
1: Circle
2: Rectangle
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 2
삭제하고자 하는 도형의 인덱스 >> 1
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 3
0: Line
1: Rectangle
삽입:1, 삭제:2, 모두보기:3, 종료:4 >> 4
C:\Users\seo\Downloads\수업\2-2\1. 원_객체지향프로그래밍\실습\homework6\x64\Debug\homework6.exe(프로세스 7900개)이 (가)
종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

3. 소스 구현 설명

a. 문제 정의

해당 문제는 벡터 기능을 이용한 그래픽 에디터 프로그램을 통해 선택한 메뉴에 따라 도형의 삽입, 삭제, 보기, 프로그램 종료의 작업을 진행하는 문제이다.

b. 문제 해결 방법

문제	해결 방법
삽입	UI::seleteShape() 메서드를 사용해 사용자가 선택한 도형 확인. new 연산자를 사용하여 동적 생성 후 vector<Shape*>에 저장.
삭제	UI::seleteDelIndex() 메서드를 사용해 삭제할 도형의 인덱스 확인. vector<Shape*>에서 해당 도형 삭제 후 delete 사용하여 메모리에서 해제.
보기	UI::showAll() 메서드를 사용해 벡터 내의 모든 도형에 대한 paint() 메서드 호출.

c. 아이디어 평가

문제	아이디어 평가
삽입	코드 수정 및 추가 시 새로운 도형을 추가하여 Shape 클래스를 상속 및 draw() 메서드를 구현하는 것만으로 해결할 수 있어 코드의 유연성과 확장성이 뛰어남. 또한 각 도형이 Shape* 타입으로 관리되어 삽입 도형의 종류에 상관없이 같은 방식으로 다룰 수 있음.
삭제	벡터 vector<Shape*>에 저장된 도형을 인덱스를 통해 빠르게 접근 및 삭제 가능. 메모리를 delete로 해제하여 동적으로 할당된 객체를 메모리 누수 없이 삭제 및 반환.
보기	Shape 클래스의 paint() 메서드를 호출하면 각 도형의 draw() 메서드가 실행되므로 다양한 도형 출력 가능. 새로운 도형이 추가되어도 시스템은 그대로 유지하여 사용 가능.

d. 알고리즘 설명

순서	설명
프로그램 시작	GraphicEditor 객체 생성. 메시지 출력 후 start() 함수 사용하여 프로그램 시작.
메인 루프 반복	무한 루프. UI::seleteMenu() 함수 사용하여 메뉴 선택.
메뉴 선택	삽입:1 UI::seleteShape() 호출하여 도형 종류 선택. 선택한 도형에 맞춰 도형 객체를 v 벡터에 추가.
	삭제:2 UI::seleteDelIndex() 호출하여 삭제할 도형의 인덱스 선택. v.begin() 사용하여 삭제할 도형 포인터 호출. 삭제 후 delete tmp;로 메모리 해제.
	모두보기:3 UI::showAll(v, it) 호출하여 현재 벡터에 저장된 모든 도형 함수 출력. 도형은 paint() 함수를 통해 출력되며, 각 도형의 draw() 함수가 호출되어 도형 이름 출력.
	종료:4 start() 함수 종료 및 프로그램 종료.