문제 정의와 문제 해결 방법 및 알고리즘 설명

이 코드는 사용자가 선(Line), 원(Circle), 사각형(Rect)과 같은 도형을 삽입, 삭제, 출력할 수 있도록 만들어야 합니다. 각 도형은 연결 리스트로 관리할 수 있게 만들었습니다. 사용자가 선, 원, 사각형 중 하나를 선택하여 리스트에 삽입할 수 있게 하고 삽입된 도형은 연결 리스트의 노드로 추가되어야 합니다. 사용자는 삭제하고자 하는 도형의 인덱스를 입력하여 특정 도형을 리스트에서 삭제할 수 있습니다. 삭제된 도형의 메모리는 해제되며, 리스트가 올바르게 연결되도록 갱신됩니다. 리스트에 저장된 모든 도형은 인덱스와 함께 출력합니다. Shape 클래스와 이를 상속받은 자식 클래스(Line, Circle, Rect)로 구현할 것입니다. 리스트는 pStart와 pLast 포인터를 사용해 관리합니다. 도형을 삽입하면 add() 메서드를 통해 리스트에 노드가 추가되고, 삭제 시 해당 노드는 올바르게 메모리에서 해제되도록 처리됩니다. 동적 메모리 할당과 해제를 통해 도형 객체를 생성 및 삭제하며 사용자가 메뉴에서 "종료"를 선택하면 프로그램이 종료되고, 종료 시 delete를 사용해 할당된 메모리를 해제합니다.

이 문제는 다형성과 추상화를 잘 활용해야하는 문제입니다. Shape 클래스를 기본으로 Line, Circle, Rect 도형을 정의하고 연결 리스트로 관리하면서, 가상함수인 virtual과 추상메서드를 통하여 문제를 해결하였습니다. 그렇기 때문에 클래스를 총 6개로 나누고 관리하게 하였습니다. 먼저 UI 클래스는 사용자 인터페이스에 관련된 작업을 처리하고 Shape 클래스는 도형들이 공통적으로 상속받는 기본 클래스로 정의했습니다. Shape 클래스의 draw() 메서드는 virtual 가상함수로 선언하여 Shape는 추상 클래스가 되었고 상속받는 파생 클래스에서 메서드가 구현되어야 합니다.
그리고 Line, Circle, Rect 클래스는 Shape 클래스를 상속받은 파생 클래스입니다. 추상 메서드인 draw() 메서드를 구현해야만 프로그램이 실행됩니다. 마지막으로 GrapicEditor 클래스입니다. 이 클래스는 도형을 추가, 삭제, 표시하는 기능을 관리하는 핵심 클래스 입니다. GrapicEditor는 Shape 객체들을 관리하는 연결 리스트로 동작합니다.

프로그램의 전체적인 실행 흐름은 main 함수에서 GrapicEditor 객체를 생성하고 run() 메서드를 호출하여 프로그램을 실행합니다. run() 메서드는 while(true) 무한루프를 사용하여 메뉴를 계속 사용자에게 표시하고 사용자의 입력에 따라 작업을 수행합니다. 사용자가 종료할 때까지 반복적으로 실행합니다. 사용자가 도형 삽입을 선택하면 input_new() 메서드를 호출합니다. 사용자가 선택한 도형 객체를 생성하고 생성된 객체를 연결 리스트에 추가합니다. 사용자가 도형을 삭제하면 del() 메서드가 호출되어 지정한 도형을 삭제합니다. del() 메서드는 연결 리스트에서 해당 도형을 제거하고, pStart와 pLast 포인터를 사용해 리스트를 다시 연결합니다. 사용자가 모두보기를 섵낵하면 show() 메서드가 호출되어 현재 리스트에 있는 모든 도형을 출력합니다. 각 객체는 paint() 메서드를 통해 출력합니다. 사용자가 종료를 선택하면 delete를 사용하여 동적으로 할당된 GraphicEditor 객체를 삭제합니다.