**2022 Spring OOP Assignment Report**

과제 번호 : prob3

학번 : 20210273

이름 : 하태혁

Povis ID : 20210273

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

1. **프로그램 개요**

이 프로그램은 15X15 사용자 화이트 보드에 알파벳 값을 원소로 가지는 도형을 출력하는 기능을 한다. 도형은 점, 수직선, 수평선, 사각형의 4종류로 이루어져 있고 main.cpp와 Menu class에서 사용자의 입력을 받고 해당하는 도형을 화이트 보드에 출력한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

프로그램 디렉토리에 있는 헤더파일은 위 종류들이 있다. Menu.h는 사용자의 입력과 각 클래스와의 연결을 이어주는 역할을 수행한다. AShapeList.h는 도형의 정보를 동적할당으로 저장하여 연결 리스트 형태로 저장하는 역할을 한다. WhiteBoard.h는 사용자에게 보여지는 화이트보드의 정보를 저장하는 기능을 한다. APoint.h, AVerticalLine.h, AHorizontalLine.h, ARectangle.h는 각 도형의 정보(좌표, 크기)를 저장하고 AShape.h의 자손 클래스이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

소스 파일의 구성은 다음과 같다. 클래스에서 method의 정의가 저장되어 있고 main.cpp는 사용자가 직접 프로그램을 실행할 수 있도록 해주는 파일이다.

1. **프로그램의 구조 및 알고리즘**

AShape 클래스는 pure virture method를 가지는 Absract class이며 derived class로는 APount class, AVerticalLine class, AHorizontalLine class, ARectangle class를 가진다. 각 자손 클래스는 알파벳 도형의 정보를 저장하고 있으며 멤버 함수에 대한 설명은 assn3 설명문에 적혀있는 바를 따라 구현하였다.

AShapeList class는 추가된 도형들을 연결 리스트 형태로 관리하는 클래스로 설명문에 나와있는 바와 같이 AShapeNode 구조체를 사용한 연결 리스트 형태로 구현하였다. 멤버함수의 구현은 전체적으로 설명문을 따랐지만, popNodeByIdx()함수를 수정하여 구현하였다.



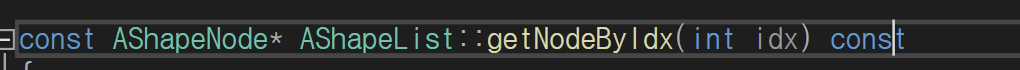
함수의 선언은 다음과 같으며 설명문에서는 idx의 노드를 연결 리스트에서 빼낸 뒤, AShapeNode의 포인터를 리턴해야 한다. 그러나 void형으로 변환한 뒤, idx 노드를 연결 리스트에서 빼낸 뒤, 함수 안에서 바로 할당해제해 줌으로서 메모리 관리를 조금 더 쉽게 구현하였다.

WhiteBoard class는 화이트 보드를 저장하는 이차원 배열과 관련된 멤버 함수를 가지고 있으며 다른 클래스에서 호출함으로서 사용된다. Menu class는 사용자의 입력에 대해 도형 연결 리스트, 화이트 보드의 정보를 직접적으로 연결시켜주는 클래스이다.

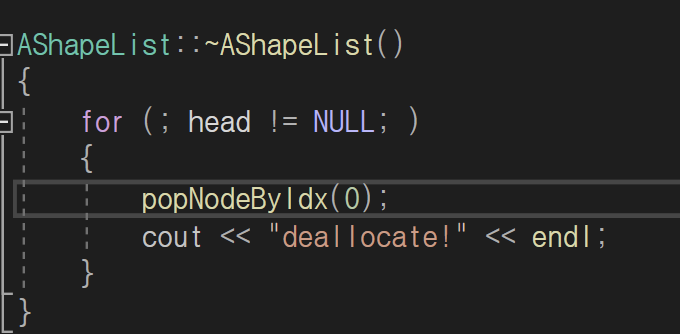
프로그램의 세부사항을 결정하는 데 있어서, 멤버 변수를 생성자에서 초기화 시켜줄 때, int와 char, string, pointer는 설명문를 따랐지만, Menu 클래스의 bool 타입은 true로 WhiteBoard 클래스에서 char형 이차원 배열은 각 요소를 ‘\*’로 초기화 시켰다.

1. **토론 및 개선**

기존 assn1, assn2에서 따로 구현하지 않았던 예외처리를 클래스 내부 수준에서 구현해보며 객체 지향 프로그램을 다루는 실력이 더 향상되었음 느꼈다. 각 기능은 모두 구현했으며 프로그램을 실형하는 데 문제가 없었다. 그러나 AShapeList class에서 아래 함수를 구현까지만 하고 실제로 사용하지는 않아서 아래 함수가 어떤 식으로 사용되는지는 더 탐구해 봐야할 것 같다.



할당된 메모리가 모두 할당해제 되었는지를 확인해 주기 위해 메모리 해제 디버깅 방식을 찾아보았지만, 잘 찾지 못했다. 그래서 결국 할당된 node 개수 만큼 자동으로 모두 할당 해제 되는지 알아보기 위해 아래와 같이 소멸자에 출력문을 사용하여 측정하였다.



1. **참고 문헌**

연결 리스트 공부 자료

<https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=kartmon&logNo=221510676043>

<https://dojang.io/mod/page/view.php?id=647>