2020 Spring OOP Assignment Report

과제 번호 : ASSN4
학번 : 20190439
이름 : 오승훈
Povis ID : sho0927

명예서약 (Honor Code)

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.
I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.
독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.
제출 기한이 지나면 LMS를 통하여 과제를 제출할 수 없습니다. 제출 기한이 지난 후에는 담당 조교에게 email로 보내야 합니다. 늦게 제출된 과제에 대한 감점은 담당 조교에게 email이 도착한 시간을 기준으로 합니다.
문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

각 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. 프로그램 개요

a. ASSN4

A. ASSN4는 template을 이용한 Shared 포인터를 구현하는 것을 목표로 한다.

B. 이번 shared 포인터에서는 특징으로 하는 것이 어떤 물체를 가르키고 있는 물체에 대해서 포인터의 개수를 세어서 만약 포인터의 개수가 0개가 될 시 그를 deallocation해주는 것이다.

C. 스마트 포인터는 어떤 자료형이든 가르칠 수 있어야 하기 때문에 template으로 구현하는 것을 기본으로 한다.

2. 프로그램의 구조 및 알고리즘

a. SmartPtr 클래스

SmartPtr private 변	`CountedObjectContainer* m_ref_object;` : 이 포인터가 가르치고 있는 물체의 정보를 저장해둔다. CountedObjectContainer 내부에는 어

수	떤 물체인지를 알려주는 정보와 이 물체가 현재 몇 개의 포인터로 가르쳐지고 있는지를 알려주는 정보가 들어가 있다.
SmartPtr public 함수	SmartPtr() SmartPtr(ObjectType* object) SmartPtr(const SmartPtr& pointer) ~SmartPtr() SmartPtr& operator=(ObjectType* object) SmartPtr& operator=(const SmartPtr& ref_pointer)

A. 이번 클래스 설명에서는 CountedObjectContainer에 대한 설명은 제외했습니다. 그 이유는 CountedObjectContainer는 저희가 구현을 해야하는 부분이 없었고, Container에 대한 설명이 필요한 부분에 있어서는 몇 개를 설명을 넣을 예정입니다.

B. 먼저 SmartPtr(ObjectType* object)는 object가 매개변수로 들어오고, 그렇다면 그 매개변수를 가지고 있는 포인터를 생성하면 될 문제이다. 그렇기 때문에 m_ref_object에 동적할당을 해서 넣어준다.

C. SmartPtr(const SmartPtr& pointer) 함수 역시 똑같이 생성자인데 이번엔 SmartPtr을 매개변수로 받았기 때문에 pointer의 m_ref_object 값을 m_ref_object에 대입을 하고, 포인터의 ref_object 값을 늘려주면 된다.

D. 소멸자의 경우 m_ref_object의 ref_count값을 하나 낮춰준다.

E. 대입 연산자의 경우 object를 받을 경우 m_ref_object == NULL이면 아무것도 가르키고 있지 않았기 때문에 동적 할당을 해서 m_ref_object 값에 넣어주면 되고, 뭔가를 가르키고 있었을 때는 m_ref_object의 ref_count값을 하나 감소시켜주고, object 에 대입을 해주면 된다.

F. SmartPtr을 매개변수로 받을 경우 NULL이면 매개변수의 m_ref_object 값을 현재 포인터의 m_ref_object에 대입을 하면 되고, 또한 m_ref_object의 ref_count 값을 1 늘려준다. 아닐 경우에는 현재 가르키고 있는 object의 ref_count값을 하나 감소시키고, 그 후 위와 동일한 행동을 하면 된다.

b. SmartMatrix

Member 변수	int m_rows, m_cols; row의 수와 col의 수를 저장해둔다. SmartArray<T> m_values; 값을 배열로 저장해둔다.
Member 함수	SmartMatrix(int rows, int cols) SmartMatrix(const SmartMatrix<T>& mtx) SmartMatrix(int rows, int cols, const T* values) void AddRow(const T* values) void AddCol(const T* values) const SmartMatrix<T> Inverse() const SmartMatrix<T> operator+(const SmartMatrix<T>& a, const SmartMatrix<T>& b)

	const SmartMatrix<T> operator-(const SmartMatrix<T>& a, const SmartMatrix<T>& b) const SmartMatrix<T> operator*(const SmartMatrix<T>& a, T s) inline const SmartMatrix<T> operator*(const SmartMatrix<T>& a, const SmartMatrix<T>& b)

A. 생성자들의 경우 여러가지 생성자들이 존재했는데 먼저, rows, cols 만 받아온경우 m_value 배열에 templete으로 만들어진 rows * cols 만큼의 동적할당을 해주면 된다.

B. SmartMatrix를 매개변수로 받아온경우 SmartMatrix의 row, col 값을 대입해주고, m_value에 row * col만큼 동적할당을 해서 매개변수로 받아온 matrix의 값들을 대입해주면된다.

C. Rows, cols, values를 받아온 경우 모든 값들을 대입해주는 것은 위와 동일하나 matrix의 m_value값을 넣어주는 것 대신 매개변수의 values에 값을 생성하려는 matrix의 m_value 값을 넣어주면된다.

D. AddRow 의 경우 row 값을 1 추가를 해주고, row * col 만큼의 새로운 배열을 동적할당 받아 새로운 배열에 기존의 배열에 존재했던 모든 값들을 대입해주면 된다. 그 후 원래 배열보다 col 만큼 길어진 곳에 매개변수로 받아온 values 들을 넣어주면된다.

E. AddCol의 경우 위와 동일한 방식으로 진행되나 이중 for문에서 두번째 for 문의 변수 값이 cols – 1 값과 동일해지면 매개변수로 받아온 values 값을 넣어주고, 나머지 값들은 동일하게 대입해주면 된다.

F. Inverse의 경우 현재 2*2 배열인지 확인을 하고, determinant 값인 ad - bc값이 0인지를 확인한 후 코드가 진행되게 구현을 해두었다. 그 후 공식인 $\frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$ 값을 원래 값에 넣어줄 수 있도록 대입을 했다.

G. 더하기와 빼기는 동일한 연산자라고 봐도 되는데 매개변수로 받아온 두 개의 배열에 row와 col이 동일해야지만 연산이 진행되도록 코드를 짜두었다.

H. 곱하기 기능의 경우 constant를 곱하는 곱셈 말고, 행렬과 행렬의 곱의 경우 앞의 행렬의 cols값과 뒤의 행렬의 row 값이 동일해야 연산이 진행되므로, 예외 처리를 해두었다. 그 후 연산은 행렬 연산과 동일하다.

3. 토론 및 개선

□ 이번 코드에서는 뭔가 C++을 제대로 공부하는 느낌이 났다. 다른 것보다 template 이라는 기능이 얼마나 유용하게 사용이 될 수 있는지에 대해서 공부를 할 수 있는 것 같아서 좋은 어싸인이라고 생각이 되었다.

- 가장 어려웠던 부분은 아무래도 skeleton 코드를 해석하는 문제였는데 그 코드를 해석함에 있어서 template이라는 개념이 생소한 문제가 존재했고, 이 코드가 언제 실행되는지에 대한 문제가 발생했던 것 같다. 또한 template을 이용한 함수와 변수를 구분을 하면서 사용을 했어야했는데 그러한 점을 제대로 공부하지 않고 시작을 하느라 어려웠던 것 같다.
- 그리고 비주얼 스튜디오에서 코드를 사용을 하고 있었는데 포인터를 이용하다 보니 ->로 접근을 시도했는데 비쥬얼 스튜디오 내부에서 접근 할려는 값들이 보이지 않았떤 문제가 발생했었다. 그래서 이 값이 실제로 접근되고 있는지 안되는지에 대한 정보를 컴파일을 통해서 알아내야 했던 부분이 답답하게 느껴졌던 것 같다.