자료구조 HW3

B935394 컴퓨터공학과 장준희

October 3, 2020

1 개념 설명

1.1 matrix.h와 matrix.cpp파일의 차이점

헤더파일은 대게 클래스의 선언부 등을 담고있다. **cpp 파일**은 대게 구현부, main()함수 등을 담고있다. 선언부와 구현부를 분리함으로써 관리의 편리, 클래스의 재사용의 편리 등을 추구할 수 있다. 과제파일에서도, .h 파일에는 선언부를, .cpp파일에서는 구현부를 작성했다.

1.2 연산자 오버로딩

연산자 오버로딩은 피연산자에 따라 다른 연산을 하도록 동일한 연산자를 중복해서 작성하는 것이다.
returnType operator operatorSymbol(list of variable) 의 형태로 선언한다.

1.2.1 +연산자

```
Matrix Matrix::operator+(const Matrix &a) {
        Matrix temp;
        for (int i = 0; i < 3; ++i) {
                for (int j = 0; j < 3; ++j) {
                        temp.m[i][j] = m[i][j] + a.m[i][j];
                }
        }
        return temp;
}
```

과제a와 과제b에서의 차이점은 범위밖에 없으므로 한 코드만을 가져와서 작성하였습니다.//대입 연산자를 오버로딩 해놓았다는 가정하에, *this의 행렬과 a의 행렬을 모든 원소를 읽어 더한 후 temp의 행렬에 대입하여 temp를 반환하였다.

1.2.2 −연산자

```
Matrix Matrix::operator-(const Matrix &a) {
        //덧셈과 동일
        Matrix temp;
        for (int i = 0; i < 3; ++i) {
                for (int j = 0; j < 3; ++j) {
                        temp.m[i][j] = m[i][j] - a.m[i][j];
                }
        }
        return temp;
}
```

1.2.3 ∗연산자

```
Matrix Matrix::operator*(const Matrix &a) {
        Matrix temp;
        for (int i = 0; i < 3; ++i) {
```

```
                for (int j = 0; j < 3; ++j) {
                        for (int h = 0; h < 3; ++h) {
                                temp.m[i][j] += m[i][h] * a.m[h][i];
                        }
                }
        }
        return temp;
}
```

행렬의 곱 공식을 보면서 +나 −연산자와는 다르게 임시 행렬의 한 원소에 고정시켜 놓은 후 그 행과 열을 각각 읽어야하기때문에 삼중 반복문을 돌려야겠다고 생각하였다. 고정을 한 후, 행과 열을 읽는 것이기 때문에 반복문의 순서를 위와 같이 작성하였다.

1.2.4 =연산자

```
void Matrix::operator=(const Matrix &a) {
        for (int i = 0; i < 3; ++i) {
                for (int j = 0; j < 3; ++j) {
                        m[i][j] = a.m[i][j];
                }
        }
}
```

+연산자에서와 마찬가지로 이중반복문을 이용해 a의 모든 원소를 *this에 대입하였다.

2 기타

2.1 기타 코드

```
void Matrix::Transpose() {
        Matrix temp;
        //temp에 전달-->=연산자 오버로딩으로 대체
        temp = *this;
        //리턴 타입이 void 이므로 m을 수정해야함
        for (int i = 0; i < 3; ++i) {
                for (int j = 0; j < 3; ++j) {
                        m[i][j] = temp.m[j][i];
                }
        }
}
void Matrix::ShowMatrix() {
        for (int i = 0; i < 3; ++i) {
                for (int j = 0; j < 3; ++j) {
                        cout << m[i][j] << " ";
                }
                cout << endl;
```

```
        }
}
int Matrix::GetDet() {
        return m[0][0] * (m[1][1] * m[2][2] - m[2][1] * m[1][2])
        - m[0][1] * (m[1][0] * m[2][2] - m[2][0] * m[1][2])
        + m[0][2] * (m[1][0] * m[2][1] - m[2][0] * m[1][1]);
}
```

클래스 자체가 행렬을 일반적으로 표현하지는 않기때문에, 행렬식에서도 굳이 포괄적인 코드를 작성하지는 않았다.

2.2 어려웠던 점

어려웠던 점은 아니지만, 과제a,b를 전부 작성한 후, 매크로의 존재를 생각해냈다. 다음 과제에는 활용할 수 있는지 체크해보는 것이 좋을 것 같다.