The Matrix and his happy fellows

HOMEWORK 6

DUE DATE: 6/20

請根據6/10上課的詳解以及範例程式碼來完成工作。請注意P.8-9的更動

Vector and Matrix

3 3x3 4 4x4 …

1x3 * 3x3 = 1x3

3x3 * 3 = 3

3 x 1

one column with 3 elements.

http://en.wikipedia.org/wiki/Matrix_multiplication

Matrix / Vector multiplication

4 * 3 = 4x3

4x3 = 3 column, each column has 4 elements

4x3 * 3x4 = 4x4

http://en.wikipedia.org/wiki/Matrix_multiplication

Implement following classes

- Base template class:
  - Column vector class for Nx1
    - One column with N elements
  - Matrix class for N x M
    - M columns with N elements per column
- And derived template classes
  - Vec1, Vec2, Vec3, Vec4
  - Mat1x1, Mat1x3, Mat1x4, Mat3x1, Mat4x1, Mat3x3, Mat3x4, Mat4x3, Mat4x4

Implement following functions

- Vector
  - Constructors
  - dotProduct(Vector &),
  - crossProduct(Vector &)

- Matrix
  - Constructors
  - Transpose()
  - Inverse()

Column
都是 行向量

Implement following operators

- Operation*
  - Matrix * Matrix
    - For example:
    - Mat1x1 = Mat1x4 * Mat4x1
    - Mat1x3 = Mat1x3 * Mat3x3
    - ...
  - Vector (+ -) Vector, Matrix (+ -) Matrix
    - Element-wise operation
  - Vector (* /) Scalar, Matrix (*/) Scalar
- Operation<<

cout << vector << endl
(5, 3, 4)
cout << matrix43 << endl
Matrix 4x3:
col[0]: (2, 5, 6, 4)
col[1]: (4, 3, 8, 2)
col[2]: (3 ,2, 4, 1)

2	4	3
5	3	2
6	8	4
4	2	1

A = [1 0 3], B = [2 3 7]
A+B = [3 3 10]
A-B = [-1 -3 -4]
A*B = [2 0 21]
A/B = [1/2 0/3 3/7]

http://en.wikipedia.org/wiki/Scalar_multiplication

不考慮向量為行向量或是列向量

Vector.dotProduct() 內積

Algebraic definition [edit]

The dot product of two vectors **A** = [A_1, A_2, ..., A_n] and **B** = [B_1, B_2, ..., B_n] is defined as:[1]

$$\mathbf{A}\cdot\mathbf{B}=\sum_{i=1}^{n}A_iB_i=A_1B_1+A_2B_2+\cdots+A_nB_n$$

where Σ denotes summation notation and n is the dimension of the vector space. For instance, in three-dimensional space, the dot product of vectors [1, 3, −5] and [4, −2, −1] is:

$$\begin{aligned}[1,3,-5]\cdot[4,-2,-1]&=(1)(4)+(3)(-2)+(-5)(-1)\\&=4-6+5\\&=3.\end{aligned}$$

T dotProduct(Vector& in)

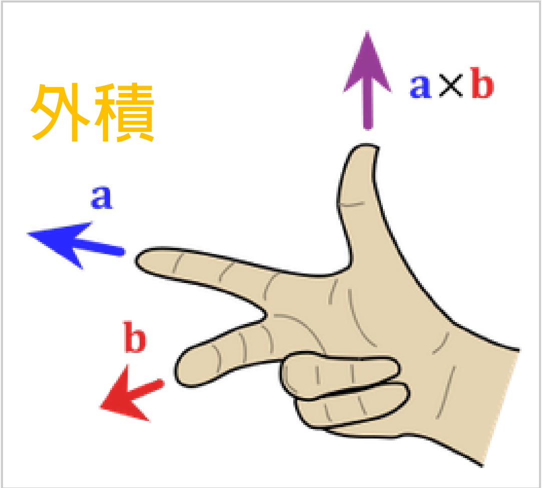
A.dotProduct(B) means A • B

這個運算不影響自己

http://en.wikipedia.org/wiki/Dot_product

不考慮向量為行向量或是列向量

Vector.crossProduct()



Finding the direction of the cross product by the right-hand rule

The cross product is defined by the formula[3][4]

$$\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \; \mathbf{n}$$

where θ is the angle between **a** and **b** in the plane containing them (hence, it is between 0° and 180°), ‖**a**‖ and ‖**b**‖ are the magnitudes of vectors **a** and **b**, and **n** is a unit vector perpendicular to the plane containing **a** and **b** in the direction given by the right-hand rule (illustrated). If the vectors **a** and **b** are parallel (i.e., the angle θ between them is either 0° or 180°), by the above formula, the cross product of **a** and **b** is the zero vector **0**.

Vector crossProduct(Vector& in)

A.crossProduct(B) means A x B

這個運算不影響自己，傳回新物件的實體

http://en.wikipedia.org/wiki/Cross_product

Matrix Transpose() 轉置矩陣

- $\begin{bmatrix}1 & 2\end{bmatrix}^T = \begin{bmatrix}1 \\ 2\end{bmatrix}$
- $\begin{bmatrix}1 & 2 \\ 3 & 4\end{bmatrix}^T = \begin{bmatrix}1 & 3 \\ 2 & 4\end{bmatrix}$
- $\begin{bmatrix}1 & 2 \\ 3 & 4 \\ 5 & 6\end{bmatrix}^T = \begin{bmatrix}1 & 3 & 5 \\ 2 & 4 & 6\end{bmatrix}$

1. $(\mathbf{A}^T)^T = \mathbf{A}$

   The operation of taking the transpose is an involution (self-inverse).

2. $(\mathbf{A}+\mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$

   The transpose respects addition.

3. $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$

Matrix Inverse() only for N x N matrix 反矩陣

In linear algebra, an n-by-n square matrix **A** is called **invertible** (also **nonsingular** or **nondegenerate**) if there exists an n-by-n square matrix **B** such that

$$\mathbf{AB} = \mathbf{BA} = \mathbf{I}_n$$

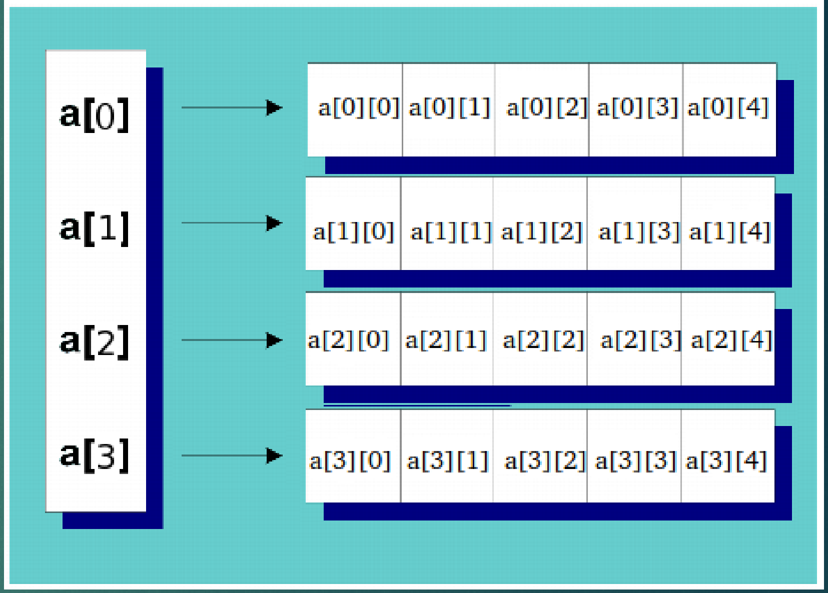
where $\mathbf{I}_n$ denotes the n-by-n identity matrix and the multiplication used is ordinary matrix multiplication. If this is the case, then the matrix **B** is uniquely determined by **A** and is called the ***inverse*** of **A**, denoted by $\mathbf{A}^{-1}$.

http://en.wikipedia.org/wiki/Transpose
http://en.wikipedia.org/wiki/Invertible_matrix

這兩個矩陣運算都不影響自己，並回傳新物件的實體

NOTE: How to create a dynamic 2D array?

```
int** ary = new int*[sizeY];
for(int i = 0; i < sizeY; ++i)
    ary[i] = new int[sizeX];
…
for(int i = 0; i < sizeY; ++i)
    delete [] ary[i];
delete [] ary;
```



sizeX = 5 and sizeY = 4

NOTE: How to create a Matrix from column Vec

```
template <int Row_Size, int Column_Size, class
Data_Type, class Column_Type, class
Matrix_Type>

class Matrix_Base

{

public:

    Column_Type _[Row_Size] ;

}
```

cout << matrix43 << endl

Matrix 4x3:
col[0]: (2, 5, 6, 4)
col[1]: (4, 3, 8, 2)
col[2]: (3 ,2, 4, 1)

2	4	3
5	3	2
6	8	4
4	2	1

END