myMatrix.h

```
#include "../../include/myMatrix.h"
```

자신의 폴더에 위치에 있는 myMatrix.h라는 헤더파일 가져오는 방법

```
#include <iostream>
#include <string>
#include <fstream>
```

```
typedef struct {
    double** at;//2d array, 더블 포인터
    int rows, cols;//dimension 정보 저장
}Matrix;
```

행렬을 표현할 구조체 선언해주기, 앞으로 이런 형식으로 행렬을 선언해 줄 것이다.

행렬을 가져올 때 기본적으로 지켜야 될 규칙들

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    return 0;
}
```

using Matrix

- addMat()
- zeros()
- ones()
- initMat()
- multMat()
- eye()
- transpose()

- copyMat()
- copyVal()
- augMat()

addMat()

두 행렬을 합할 때 사용을 한다. 행렬을 더할 때 두 행렬의 크기는 같아야 한다.

```
Matrix  addMat(Matrix _A, Matrix _B);
```

Parameters

> _A: A행렬을 의미
>
> _B: B행렬을 의미

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix matU = txt2Mat(path, "prob1_matU");
    Matrix matAdd = addMat(matA, matU);
    printMat(matA, "matA");
    printMat(matU, "matU");
    printMat(matAdd, "matA + matU");
    return 0;
}
```

output

```
matA =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

matU =
       4.000000       -2.000000       -3.000000        6.000000
       0.000000        4.000000        2.000000        3.000000
       0.000000        0.000000        3.000000       -2.000000
       0.000000        0.000000        0.000000        4.000000
```

```
matA + matU =
       8.000000       -4.000000       -6.000000       12.000000
      -6.000000       11.000000        8.500000       -3.000000
       1.000000        7.500000        9.250000        3.500000
     -12.000000       22.000000       15.500000        3.000000
```

Warning

- A와 B의 행렬의 크기는 같아야 한다.

Error Handling

- 두 행렬의 열의 총 개수가 같아야 하고 행의 총 개수도 같아야 한다.

zeros()

행렬의 값을 0으로 초기화한 행렬을 만들고 싶을 때 사용이 된다.

```
Matrix  zeros(int _rows, int _cols)
```

Parameters

> _rows: 만들고 싶은 행렬의 행의 전체 개수
>
> _cols: 만들고 싶은 행렬의 열의 전체 개수

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matTemp = zeros(4, 4);
    printMat(matTemp, "zeros result")
    return 0;
}
```

output

```
zeros result =
       0.000000        0.000000        0.000000        0.000000
       0.000000        0.000000        0.000000        0.000000
       0.000000        0.000000        0.000000        0.000000
       0.000000        0.000000        0.000000        0.000000
```

Warning

- _rows, _cols는 int의 형식이어야 한다.

ones()

행렬의 값을 1로 초기화한 행렬을 만들고 싶을 때 사용이 된다.

```
Matrix  ones(int _rows, int _cols);
```

Parameters

> _rows: 만들고 싶은 행렬의 행의 전체 개수
>
> _cols: 만들고 싶은 행렬의 열의 전체 개수

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matTemp = ones(4, 4);
    printMat(matTemp, "ones result");
    return 0;
}
```

output

```
ones result =
       1.000000        1.000000        1.000000        1.000000
       1.000000        1.000000        1.000000        1.000000
       1.000000        1.000000        1.000000        1.000000
       1.000000        1.000000        1.000000        1.000000
```

Warning

- _rows, _cols는 int의 형식이어야 한다.

initMat()

행렬에 특정 값으로 초기화하고 싶을 때 사용이 된다.

```
void    initMat(Matrix _A, double _val);
```

Parameters

> _A: 초기화를 할 행렬
>
> _val: 초기화를 하고 싶은 값

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matTemp = createMat(4, 4);
    initMat(matTemp, 3.141592);
    printMat(matTemp, "initMat result")
    return 0;
}
```

output

```
initMat result =
       3.141592        3.141592        3.141592        3.141592
       3.141592        3.141592        3.141592        3.141592
       3.141592        3.141592        3.141592        3.141592
       3.141592        3.141592        3.141592        3.141592
```

Warning

- _A는 행렬, _val은 double의 형식이어야 한다.

multMat()

행렬 곱을 하고 싶을 때 사용을 한다.

A는 m x n행렬, B는 n x r행렬일 때 AB행렬의 크기는 m x r 이다.

```
Matrix  multMat(Matrix _A, Matrix _B);
```

Parameters

> _A: 행렬 곱 시 왼쪽에 위치해있는 행렬
>
> _B: 행렬 곱 시 오른쪽에 위치해있는 행렬

Example code

```
#define ASGN        999     // enter your assignment number
```

```
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix matU = txt2Mat(path, "prob1_matU");
    Matrix matTemp = multMat(matA, matU);
    printMat(matA, "matA");
    printMat(matU, "matU");
    printMat(matTemp, "A times U result");
    return 0;
}
```

output

```
matA =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

matU =
       4.000000       -2.000000       -3.000000        6.000000
       0.000000        4.000000        2.000000        3.000000
       0.000000        0.000000        3.000000       -2.000000
       0.000000        0.000000        0.000000        4.000000

A times U result =
      16.000000      -16.000000      -25.000000       48.000000
     -24.000000       40.000000       51.500000      -52.000000
       4.000000       28.000000       30.750000       38.000000
     -48.000000      112.000000      126.500000      -41.000000
```

Warning

- A의 열의 총 개수와 b의 행의 총 개수는 같아야 한다.
- 출력 시 사이즈는 (A의 행의 총 개수 x B의 열의 총 개수)인 행렬이다.
- _A, _B는 행렬이어야 한다.

Error Handling

- A의 열의 총 개수와 b의 행의 총 개수는 같지 않으면 오류가 발생한다.

eye()

단위 행렬을 만들고 싶을 때 사용을 한다. 주대각선에 있는 값은 1이고 나머지 값들은 0인 행렬이다.

```
Matrix  eye(int _rows, int _cols);
```

Parameters

> _rows: 만들고 싶은 행렬의 전체 행의 개수
>
> _cols: 만들고 싶은 행렬의 전체 열의 개수

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matTemp = eye(4, 4);
    printMat(matTemp, "eye result");
    return 0;
}
```

output

```
eye result =
       1.000000        0.000000        0.000000        0.000000
       0.000000        1.000000        0.000000        0.000000
       0.000000        0.000000        1.000000        0.000000
       0.000000        0.000000        0.000000        1.000000
```

Warning

- 주 대각선에 있는 값들은 1어야 한다.
- 나머지 위치에 있는 값들은 0이어야 한다.

transpose()

전치행렬을 만들고 싶을 때 사용을 한다.

1. (n x m)의 행렬을 만든다.
2. 각 열에 있는 값들은 행으로 각 행에 있던 값들은 열로 이동을 한다.

```
Matrix  transpose(Matrix _A);
```

Parameters

> _A: 전치행렬로 만들고 싶은 행렬

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix matTemp = transpose(matA);
    printMat(matA, "matA");
    printMat(matTemp, "A trasnpose result");
    return 0;
}
```

output

```
madA is =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

A trasnpose result =
       4.000000       -6.000000        1.000000      -12.000000
      -2.000000        7.000000        7.500000       22.000000
      -3.000000        6.500000        6.250000       15.500000
       6.000000       -6.000000        5.500000       -1.000000
```

Warning

- 각 열에 있는 값들은 행으로 옮기고 행에 있는 값들을 열로 옮긴다.
- _A는 행렬이어야 한다.

copyMat()

새로운 행렬을 만들고 만들 행렬의 값으로 원하는 행렬의 전체 값을 그대로 가져오기 위해 사용을 한다.

```
Matrix  copyMat(Matrix _A);
```

Parameters

> _A: 복사하고 싶은 행렬

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]
```

```
#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix matTemp = copyMat(matA)
    printMat(matA, "matA");
    printMat(matTemp, "Copy result");
    return 0;
}
```

output

```
madA is =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

Copy result =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000
```

Warning

- _A는 행렬이어야 한다.
- 두 행렬의 크기는 같아야 한다.

copyVal()

두 행렬을 가져온다. B행렬에 A행렬의 값을 그대로 대입한다.

```
void    copyVal(Matrix _A, Matrix _B);
```

Parameters

> _A: 복사하고 싶은 행렬
>
> _B: 붙여넣기를 당할 행렬

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
```

```
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix matTemp = createMat(matA.rows, matA.cols);
    copyVal(matA, matTemp);
    printMat(matA, "matA");
    printMat(matTemp, "CopyVal result");
    return 0;
}
```

output

```
madA is =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

CopyVal result =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000
```

Warning

- _A, _B는 행렬이어야 한다.
- 두 행렬의 크기는 같아야 한다.

eye()

단위 행렬을 만들고 싶을 때 사용을 한다. 주대각선에 있는 값은 1이고 나머지 값들은 0인 행렬이다.

```
Matrix  eye(int _rows, int _cols);
```

Parameters

> _rows: 만들고 싶은 행렬의 전체 행의 개수
>
> _cols: 만들고 싶은 행렬의 전체 열의 개수

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
```

```
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
    path += "eval/";
#endif
    Matrix matTemp = eye(4, 4);
    printMat(matTemp, "eye result");
    return 0;
}
```

output

```
eye result =
       1.000000        0.000000        0.000000        0.000000
       0.000000        1.000000        0.000000        0.000000
       0.000000        0.000000        1.000000        0.000000
       0.000000        0.000000        0.000000        1.000000
```

Warning

- 주 대각선에 있는 값들은 1어야 한다.
- 나머지 위치에 있는 값들은 0이어야 한다.

Error Handling

- A의 열의 총 개수와 b의 행의 총 개수는 같지 않으면 오류가 발생한다.

augMat()

augmented matrix를 만들고 싶을 때 사용을 한다. 이때 두 행렬의 행의 총 개수는 같아야 한다.

A행렬에 B행렬을 추가적으로 첨가한 것처럼 보인다.

```
Matrix  augMat(Matrix _A, Matrix _B);
```

Parameters

> _A: augmented matrix를 만들 때 왼쪽에 위치해 있는 행렬
>
> _B: augmented matrix를 만들 때 오른쪽에 위치해 있는 행렬

Example code

```
#define ASGN        999     // enter your assignment number
#define EVAL        0       // [¡Ø DO NOT EDIT !!!]

#include "../../include/myMatrix.h"
int main(int argc, char* argv[])
{
    /*   [¡Ø DO NOT EDIT !!!]   Resources file path setting for evaluation  */
    std::string path = "C:/NP_data/Assignment" + std::to_string(ASGN) + "/";


#if EVAL
```

```
    path += "eval/";
#endif
    Matrix matA = txt2Mat(path, "prob1_matA");
    Matrix vecb = txt2Mat(path, "prob1_vecb");
    Matrix matAug = augMat(matA, vecb);
    printMat(matA, "matA");
    printMat(vecb, "vecb");
    printMat(matAug, "matAug is");
    return 0;
}
```

output

```
matA =
       4.000000       -2.000000       -3.000000        6.000000
      -6.000000        7.000000        6.500000       -6.000000
       1.000000        7.500000        6.250000        5.500000
     -12.000000       22.000000       15.500000       -1.000000

vecb =
      12.000000
      -6.500000
      16.000000
      17.000000

matAug is =
       4.000000       -2.000000       -3.000000        6.000000       12.000000
      -6.000000        7.000000        6.500000       -6.000000       -6.500000
       1.000000        7.500000        6.250000        5.500000       16.000000
     -12.000000       22.000000       15.500000       -1.000000       17.000000
```

Warning

- 두 행렬의 전체 행의 개수는 같아야 한다.

Error Handling

- A행렬과 B행렬의 행의 개수가 같지 않으면 오류가 발생한다.