Object Oriented Programming (IGS2130)

Lab 2

Instructor:
Choonwoo Ryu, Ph.D.

INHA UNIVERSITY

Exercise #1

- Write a program to search an element in array
  - Use the following main() function and dynamic memory allocation for the memory buffer
  - Complete two functions (inputArray(), searchIndex()) in the code so the main() function can execute as like the following execution example

```
void inputArray(int*, int);
int searchIndex(int*, int, int);
int main() {
    int size, element, index;
    int* buf = nullptr;

    cout << "Enter size of array: ";
    cin >> size;

    // 1. allocate the memory buffer

    // 2. use the allocated memory
    cout << "Enter elements of the array: ";
    inputArray(buf, size);

    cout << "Enter element to search: ";
    cin >> element;
    index = searchIndex(buf, size, element);

    if (index == -1)
        cout << element << " does not exist in the array.";
    else
        cout << element << " is found in index " << index << '.';

    // 3. free the allocated memory
    return 0;
}
```

Execution example

```
Enter size of array: 5
Enter elements of the array: 10 20 30 40 50
Enter element to search: 30
30 is found in index 2.


Enter size of array: 4
Enter elements of the array: 10 20 30 40
Enter element to search: 5
5 does not exist in the array.
```

Exercise #2

- Write a program of matrix addition
  - Use the following main() function and dynamic memory allocation for the memory buffer of the matrix
  - Complete three functions (MatInput(), MatAdd() and MatDisplay()) in the code so the main() function can execute as like the following execution example

$$\mathbf{A}+\mathbf{B}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}+\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix}$$

$$=\begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{bmatrix}$$

Exercise #2

```
void MatInput(int*, int, int);
void MatAdd(int*, int*, int*, int, int);
void MatDisplay(int*, int, int);

int main() {
    int row, col;
    int* mat1, * mat2, * matAdd;

    cout << "Enter size of matrix(row, col): ";
    cin >> row >> col;

    // 1. allocate the memory buffers for the 3 matrices

    // 2. use the buffers
    cout << "Enter elements in 1st matrix of size " << row << 'x' << col << ":\n";
    MatInput(mat1, row, col);
    cout << "Enter elements in 2nd matrix of size " << row << 'x' << col << ":\n";
    MatInput(mat2, row, col);

    MatAdd(mat1, mat2, matAdd, row, col);

    cout << "Sum of two matrices: \n";
    MatDisplay(matAdd, row, col);

    // 3. free the allocated buffers

    return 0;
}
```

```
Enter size of matrix(row, col): 3 3
Enter elements in 1st matrix of size 3x3:
10 20 30
40 50 60
70 80 90
Enter elements in 2nd matrix of size 3x3:
1 2 3
4 5 6
7 8 9
Sum of two matrices:
11  22  33
44  55  66
77  88  99
```

Execution example

Exercise #3

- Write a program of matrix multiplication
  - Use the following main() function and dynamic memory allocation for the memory buffer of the matrix
  - Use two functions(MatInput() and MatDisplay()) from Exercise #2
  - Complete matrix multiplication function (MatMul()) in the code so the main() function can execute as like the following execution example

$$\mathbf{C} = \mathbf{AB}$$

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix}$$

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^{n} a_{ik}b_{kj}$$

$$\mathbf{C} = \begin{pmatrix} a_{11}b_{11} + \cdots + a_{1n}b_{n1} & a_{11}b_{12} + \cdots + a_{1n}b_{n2} & \cdots & a_{11}b_{1p} + \cdots + a_{1n}b_{np} \\ a_{21}b_{11} + \cdots + a_{2n}b_{n1} & a_{21}b_{12} + \cdots + a_{2n}b_{n2} & \cdots & a_{21}b_{1p} + \cdots + a_{2n}b_{np} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{11} + \cdots + a_{mn}b_{n1} & a_{m1}b_{12} + \cdots + a_{mn}b_{n2} & \cdots & a_{m1}b_{1p} + \cdots + a_{mn}b_{np} \end{pmatrix}$$

Exercise #3

```
Enter elements in 1st matrix of size 3x3:
1 0 0
0 1 0
0 0 1
Enter elements in 2nd matrix of size 3x3:
10 20 30
40 50 60
70 80 90
Multiplication of two matrices:
10  20  30
40  50  60
70  80  90
```

Execution example

```
void MatInput(int*, int, int);
void MatMul(int*, int*, int*, int, int);
void MatDisplay(int*, int, int);

int main() {
    int row = 3, col = 3;
    int* mat1, * mat2, * matAdd;

    // 1. allocate the memory buffers for the 3 matrices

    // 2. use the buffers
    cout << "Enter elements in 1st matrix of size " << row << 'x' << col << ":\n";
    MatInput(mat1, row, col);
    cout << "Enter elements in 2nd matrix of size " << row << 'x' << col << ":\n";
    MatInput(mat2, row, col);

    MatMul(mat1, mat2, matAdd, row, col);

    cout << "Multiplication of two matrices: \n";
    MatDisplay(matAdd, row, col);

    // 3. free the allocated buffers

    return 0;
}
```