**과제2\_1번.cpp**

**1. functions and variables**

1) typedef struct element

: to store only non-zero elements of a matrix

-value: value of an element

-row, col: index of an element

2) typedef struct SparseMatrix

-data: value and location of non-zero elements

-rows: number of rows in a sparsematrix

-cols: number of columns in a sparse matrix

-terms: number of non-zero terms in a sparse matrix

3) bool compare(element a, element b)

: this function sorts 'data' array into ascending order

4) int main()

-BT: transpose matrix of B

-sort(BT.data, BT.data + 7, compare): this function sorts 'data' array of BT into ascending order. it is in algorithm header

-matrixB: store all elements of matrix B in 2D array form(including zero elements)

-matrixBT: store all elements of transpose of B in 2D array form(including zero elements)

**2. Theoretical explanation**

1) declare SpareMatrix B and BT, and define B freely

2) transpose operation

- swap the number of rows and columns of B and BT

- using for loop(for (int i = 0; i < B.terms; i++) ), swap row and col of each value

- using sort function from algorithm header, sort the data array in ascending order

3) declare matrixB and matrixBT, and store all elements of matrix B and BT

4) print B and transpose of B in a dense matrix form

**3. Result**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**과제2\_2번.c**

**1. functions and variables**

1) int floor = 3; //first axis of 3D array

int row = 3; //second axis of 3D array

int col = 3; //third axis of 3D array

2) double\*\*\* mem\_alloc\_3D\_double()

: allocate 3D array(m) using dynamic memory allocation, and return m

3) void printArray(double\*\*\* a)

: print a 3D array

parameter

-a: a 3D array to print

4) double\*\*\* addition\_3D(double\*\*\* a, double\*\*\* b)

: add 3D array A and B, store the result into double\*\*\* type variable n, and then return n

paramter

-a, b: two 3D arrays. we will add a and b in this function

5) int main()

-A, B: 3D array of double type

-C: 3D array of double type, we will store addition of array A and B in C

**2. Theoretical explanation**

1) declare double\*\*\* type A, B, and C

2) define two matrices A and B using dynamic memory allocation

3) perform addition of two matrices A and B

4) print A, B, and A+B

: three arrays are 3D array(3x3x3), so lets print 3 different 2D arrays and distinguish them by first axis(3D array)

5) deallocate A and B and C

**3. Result**

**텍스트이(가) 표시된 사진

자동 생성된 설명**