2018 Spring 자료구조 실습 과제
#2: Sparse Matrix using Linked Lists

제출 관련 사항

· 제출 기한: 5월 16일 오후 5시
· 제출 방법: 코드 파일을 압축하여 지정된 E-mail에 첨부하여 발송
  E-mail 제목: ds_학번_02
  압축 파일명: ds_학번_02.zip
  코드 파일명: ds_학번_02.c
  제출 E-mail 주소: sgds2018s@gmail.com
· 유의 사항
  Late는 받지 않음. 반드시 제출 기한 전에 제출 완료 할 것.
  E-mail 제목, 파일명 오류 시, 건 당 10% 감점
  지정된 입출력 형식 오류 시, 건 당 10% 감점
  Copy의 경우 적발 시 F학점을 부여

1. 과제 목적

Linked Lists를 이용하여 Sparse Matrix를 생성하는 방법을 익히고 이를 바탕으로 row/column switching, transpose matrix 생성을 수행하는 함수를 구현한다.

2. 개발환경

Visual Studio 2015 (2017버전을 사용할 경우, 제출 시 메일에 반드시 명시)

3. 과제 수행 방법

교재 4장의 Linked Lists Sparse Matrix 부분을 참고하여 mread, mwrite, merase 함수를 익히고, 이를 바탕으로 matrix row/column switching function과 transpose function을 구현한다. main 함수 호출은 아래와 같이 진행한다.

```
#define MAX_SIZE 50
typedef enum {head, entry} tagfield;
typedef struct matrix_node *matrix_pointer;
typedef struct entry_node {
        int row;
        int col;
        int value;
};
typedef struct matrix_node {
        matrix_pointer down;
        matrix_pointer right;
        tagfield tag;
        union {
           matrix_pointer next;
          entry_node entry;
        } u;
matrix_pointer hdnode[MAX_SIZE];


matrix_pointer mread(FILE*);                 // Read matrix from File

void mwrite(File*, matrix_pointer);          // Write matrix to File

void merase(matrix_pointer*);                // Free matrix from main memory

matrix_pointer mswitch(matrix_pointer, int , int , int );
                                            //matrix switching(3.1 참조)

matrix_pointer mtranspose(matrix_pointer); // matrix transpose



void main()
{
int x                // row/column 선택 변수(0 or 1)
int y                // switching 할 행,열 선택 변수
int z                // switching 할 행,열 선택 변수
scanf("%d %d %d", &x, &y, &z)

matrix_pointer a, d;
```

```
a = mread(FILE);
mwrite(NULL , a); // a가 read 되었는지 확인하기 위해 출력

d = mswitch(a, x, y, z);
mwrite(FILE , d); //switch된 행렬을 .txt에 저장
merase(&d);

d = mtranspose(a);
mwrite(FILE , d); //transpose된 행렬을 .txt에 저장
merase(&d);
merase(&a);
mwrite(NULL , d); //d가 erase 되었는지 확인하기 위해 출력
mwrite(NULL , a); //a가 erase 되었는지 확인하기 위해 출력
}
```

3.1 Row/Column switching

linked list로 구현한 matrix에 대하여, 임의의 두 행 또는 두 열을 서로 switching 하는 mswitch 함수를 구현한다. 우선 위의 코드와 같이 변수 x, y, z를 설정한다. 변수 y와 변수 z는 0이상의 값으로, switching할 두 행 또는 열의 값을 의미한다. x는 행을 switching할 것인지 열을 switching할 것인지를 정하는 변수로, 0또는 1의 값을 갖는다. x=0이면 y행과 z행을 switching하며, x=1이면 y열과 z열을 switching하도록 함수를 구성한다.

아래 예시는 7-by-7 행렬에 대해 x=0, y=0, z=6을 입력했을 때의 결과이다.

$$\begin{bmatrix}4&1&0&0&0&0&2\\7&5&6&8&0&3&4\\3&0&0&0&0&5&0\\0&0&4&0&0&0&0\\1&0&0&0&7&0&2\\0&4&0&1&0&0&1\\5&0&0&0&6&0&0\end{bmatrix} \xrightarrow{row(0)\leftrightarrow row(6)} \begin{bmatrix}5&0&0&0&6&0&0\\7&5&6&8&0&3&4\\3&0&0&0&0&5&0\\0&0&4&0&0&0&0\\1&0&0&0&7&0&2\\0&4&0&1&0&0&1\\4&1&0&0&0&0&2\end{bmatrix}$$

만약 같은 행렬에 대해 x=1, y= 2, z=3을 입력하면 그 결과는 아래와 같다.

$$\begin{bmatrix}4&1&0&0&0&0&2\\7&5&6&8&0&3&4\\3&0&0&0&0&5&0\\0&0&4&0&0&0&0\\1&0&0&0&7&0&2\\0&4&0&1&0&0&1\\5&0&0&0&6&0&0\end{bmatrix} \xrightarrow{col(2)\leftrightarrow col(3)} \begin{bmatrix}4&1&0&0&0&0&2\\7&5&8&6&0&3&4\\3&0&0&0&0&5&0\\0&0&0&4&0&0&0\\1&0&0&0&7&0&2\\0&4&1&0&0&0&1\\5&0&0&0&6&0&0\end{bmatrix}$$

3.2 Transpose

linked list로 구현한 matrix에 대하여, 아래와 같이 transpose 행렬을 구하는 mtranspose 함수를 구현한다.

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

4. 입출력 양식

입력

입력은 행렬의 정보가 담긴 "A.txt"에서 받는다. 입력 파일의 예시는 다음과 같다.

```
6 7
0 4 0 7 0 0 9
2 0 0 0 6 5 0
0 0 3 8 0 4 7
0 0 0 0 0 1 0
1 2 3 0 0 0 8
0 5 0 4 3 0 2
```

파일의 맨 첫 줄에는 행렬의 행의 수와 열의 수가 입력되어 있다. 행과 열의 첫 번째 항을 0으로 하고, 마지막 항을 n-1이라고 지정한다. 즉 예시에서 (6 7)의 경우 0행부터 5행, 0열부터 6열로 구성된 행렬이다.

그 다음 줄부터는 행렬의 각 행 별 원소의 정보가 저장되어 있다. 즉, 입력 파일에서 3번째 줄의 5번째 항목인 값 "6"은 행렬의 1 행 4열의 원소임을 나타낸다.

출력

출력은 Sparse Matrix로 출력한다. 출력 파일의 예시는 아래와 같다.

0 0 1
0 2 5
1 2 9
2 0 2
2 1 1

출력 파일의 각 줄에는 행 열 값 순으로 출력되어 있다. 즉, 출력 파일의 예시에서 3번째 줄의 값 "1 2 9"는 행렬의 1행 2열의 값이 9임을 나타낸다.

본 과제에서 출력해야 하는 파일은 총 **2개**이다. 먼저 "A.txt"에 담긴 행렬을 입력 받아서 (x,y,z)값에 따라 행 또는 열을 switching하는 mswitch 함수를 거쳐 구한 행렬의 정보를 "switch_학번.txt"에 저장하여 출력한다. 그리고 "A.txt"에 담긴 행렬의 transpose 행렬을 구하기 위해 mtranspose 함수를 거쳐 구한 행렬의 정보를 "transpose_학번.txt"에 저장하여 출력한다.

모든 과정이 끝난 이후엔 사용한 메모리를 모두 해제하고, 프로그램을 정상적으로 종료한다.

5. 점수 부여 방식

점수는 **과제 채점 100%**로 점수를 부여한다. 과제 채점은 각각 3개의 테스트 케이스(행렬의 크기를 변화하며)에 대하여 정답/오답을 확인하고 채점한다. 각각의 테스트 케이스에서 발생하는 Memory 관련 오류에 대해선 해당 테스트 케이스를 0점 처리 한다. 이후, 감점 사항에 대해 감점을 진행하고 최종 과제 점수를 부여한다.