Lab Manual – 2D Pointer && Dynamic 2D Array

Important Note:
- Have an improved understanding of pointers.
✔ Declaring and Initializing pointers
✔ Pointer Operations
There shouldn't be any memory leakage or dangling pointers in your program.
- Make separate functions for input and output of arrays. Your main should be a sequence of function calls only
- You are not allowed to use global variables and goto instruction
- Submit only one cpp file having main function testing all the following functions

Task 1

i) Write a function void populate_array(int **my_array,int m,int n) that initialize an array with random data.
ii) Write a function void disp_array(int **my_array,int m, int n) display an element of an array correctly (m*n format)
iii) write a function int **transpose_array(int **my_array,int m,int n) that creates a dynamic 2d array of size n*m that is the transpose of an array and returns a 2d pointer to the transpose array.

Sample input:
Enter the size of matrix m*n : 3 3

Original Matrix:

1 2 3
4 5 6
7 8 9

Transpose Matrix:

1 4 7
2 5 8
3 6 9

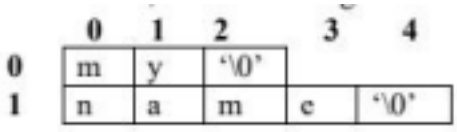
Task 2

i) Write a function char** AllocateMemory(int& rows, int& cols) that takes the size of matrix (rows and
columns) from the user, allocates memory for the matrix and returns its pointer.
ii) Write a function void InputMatrix(char** matrix, const int rows, const int cols) which takes input the
values in matrix from user(console).

iii) Write a function void DisplayMatrix(char** matrix, const int & rows, const int &cols) that displays the
matrix in proper format.
iv) Write a function that does the following:
Creates three dynamic char arrays namely alphabets, numbers, and specialchar.(Define the sizes yourself). Iterate the 2D array and separate alphabet elements and save them in the alphabets array, separate number elements and save them in numbers array and separate special character
elements and save them in the specialchar array. The function returns the three arrays alphabets, numbers, and specialchar. Note: The three arrays should not consume any extra space. Resize the arrays
accordingly. For example, if the Sample Matrix is
A 1 v @
+ 9 s 6
P # ^ 4
Your function will return the following arrays:
alphabets = A v s P
numbers = 1 9 6 4
specialchar = @ + # ^

Task 3

Implement a C++ Function void **myTokenizer**(char *data, char **list_tokens, char delimiter) Your function should store the tokens in the list_tokens and split the data array on the basis of delimiter. Delimiter is another name for 'separator'. Call the function in main and print the list_tokens. Start traversing the data array until you find delimiter. Once you find the delimiter store the first token in the first row of list_tokens. Now find second token and store in the second row of list_tokens and so on... First find the number of tokens that can be formed from data. This will be the number of rows for char**__list_tokens__. Each row will have a different number of columns. e.g. If string is my,name and delimiter is ',' then following shall be the result.



Don't allocate extra memory. Release memory before exiting the program.
Sample:
Input: my,name,is,Mr,Faheem
Delimiter: ,
Tokens are:
my
name
is
Mr
Faheem