Assignment-1

Data Structures and Algorithms
Pointers, Abstract Data Types, and Linked Lists

Due Date: 28 March, 2023 11:59 PM

Important Notes

- For questions 1 and 2, all functions mentioned must be defined with **exact** function name and function parameters and return type (we would run scripts to check this, so match the case of alphabets also)
- OPERATION names are case sensitive
- Doubt Document
- Plagiarism Policy. Please read this before you start the assignment

TOTAL: [350 PTS]

1 Pointer Warmup [40 PTS]

Pointers is a very important concept in C programming. Let us begin with a few warmup questions.

1.1 File Structure

- `functions.h` - Which has the function prototypes.
- `functions.c` - Which implements the functions defined in 1.2.
- `main.c` - Which must use the functions and perform various operations on input and produce output as specified later.

1.2 Functions

- `void reverseString(char* str, int length)`: takes in a pointer to a string and its length and reverses the string in place.
- `char* compressString(char* str, int length)`: takes in a pointer to a string and its length, and returns a pointer to a string that is a compressed version of the original, where each character is followed by the number of times it appears consecutively in the original string. For example, "**aaabbc**" would be compressed to "**a3b2c1**".
  NOTE: No of times a character occurs continuously can be any number. (need not be single digit)
- `int* uniqueElements(int* arr, int length)`: takes in a pointer to an array of integers and its length, and returns a pointer to an array of integers that contains only the unique elements of the original array, without using any library functions except `malloc`.
  NOTE: The size of the array you are returning must be exactly equal to the number of unique elements.
- `int** transpose(int** matrix, int NumRow, int NumCol)`: takes in a pointer to a matrix of integers and its dimensions and returns a pointer to a matrix that represents the transpose of the original matrix.
  NOTE: The dimensions of the matrix you are returning must be exactly equal to the dimensions of the mathematical transpose.

1.3 Input and Output

The first line contains T, the number of OPERATION's that need to be done. $1 \leq T \leq 100$

The first line of an OPERATION consists of a string indicating the OPERATION name.

The next few lines contain the required input data according to the table i.e.

Operation name	**Corresponding function**
OPER1	*reverseString*
OPER2	*compressString*
OPER3	*uniqueElements*
OPER4	*transpose*

OPERATION	**INPUT FORMAT**	**CONSTRAINTS**	**OUTPUT FORMAT**
OPER1	First line contains an integer n indicating the length of the string. Second line contains the string	$1 \leq n \leq 10^4$ [**10 PTS**]	String (Reversed)
OPER2	First line contains one integer n indicating the length of the string. Second line contains the string	$1 \leq n \leq 10^4$ [**10 PTS**] $str[i]$ contains only lowercase alphabet	String (Compressed)
OPER3	First line contains an integer n indicating the length of the array. Second line contains n space separated integers.	$1 \leq n \leq 10^4$ [**10 PTS**] $1 \leq arr[i] \leq 10^4$	All unique elements in a line, space separated and should be in the order they appear first
OPER4	First line contains 2 integers R and C indicating the number of rows and the number of columns, respectively. Following R lines contains C space-separated elements	$1 \leq R, C \leq 10^2$ [**10 PTS**] $1 \leq Mat[i][j] \leq 10^6$	The transposed matrix. Each row should be printed on a new line

1.4 Example Test Cases

Input	**Output**
4	
OPER1 10 abcdefghij	 jihgfedcba
OPER2 12 aacceeggiiaa	 a2c2e2g2i2a2
OPER3 10 1 1 2 4 5 9 9 1 6 8	 1 2 4 5 9 6 8
OPER4 3 4 1 2 3 4 2 3 4 5 3 4 5 6	 1 2 3 2 3 4 3 4 5 4 5 6