Lab 3 / Assignment (Level 1)

1. Write the following functions related to integer array:

a. `void inverse (int *trg, int *src, const int SIZE)`

Consider both source and target have the same memory size. Store **additive inverse** of each element in source array in target array at corresponding position. For example, if the source array has elements [24, 45, -34, 16, -28] then the target array will have elements [-24, -45, 34, -16, 28].

b. `int countElementsInOrder (int *src, const int SIZE)`

Consider general sorting that is sorting in ascending order. Elements are sorted if they are in order, that is every element except the last element is smaller or equal to its adjacent (right side) next element. **Write a function to count**, how many elements are in order. For example, if the source array has elements [24, 45, 34, 55, 66, 28, 39, 22] then **three** elements are in order. 24 is smaller than its next element, 45. Similarly, 34 is smaller than 55 and 28 is smaller than 39.

c. `int countOccurence (int *src, const int ELEMENT, const int SIZE)`

Write a function to find and count a given element in the source array. For example, if the source array has elements [24, 45, 34, 45, 24, 28, 45, 22] then there are **three** occurrences of 45 that are at index 1, index 3 and index 6.

d. `int countCommonElements (int *a1, int *a2, const int SIZE)`

Write a function to find the count of elements common in both arrays. For example, if array 1 has elements [24, 45, 34, 45, 84, 28, 85, 22] and array 2 has elements [34, 55, 24, 65, 54, 78, 22, 92] then there are **three** elements common in both arrays that is 24, 34 and 22.

e. `int countCommonElements (int *a1, int SIZE1, int *a2, int SIZE2)`

Put const with size 1 & size 2. Write a function to find count of element common in both arrays, where both arrays may have different size. For example, if array 1 has elements [24, 45, 34, 95, 84, 28, 85, 22] and array 2 has elements [34, 55, 24, 65, 54, 78, 92] then there are **two** elements common in both arrays that are 24, and 34.

f. `int countDifferentElements (int *a1, const int SIZE1, int *a2, const int SIZE2)`

Write a function to find different elements in both arrays, where both arrays may have different sizes. For example, if array 1 has elements [24, 45, 34, 95, 84, 28, 85, 22] and array 2 has elements [34, 55, 24, 65, 54, 78, 92] then there are eleven elements common in both arrays that is 45, 95, 84, 28, 85, 22, 55, 65, 54, 78 and 92.

<----------->

2. Write following functions related to character array/ string:

a. `int countVowels (char *a, const int SIZE)`

Array of characters will be passed to function, with number of characters greater than equal size and function will count & return vowel characters in the array, which includes both capital and small letters.

b. `int countAlphabets (char *a, const int SIZE)`

Array of characters may have alphabets, numbers or any other characters. Write a function to find and count capital & small alphabet characters exist in the array.

c. `int countVowels (char *a)`

It is a similar function as “a)”, somehow but the difference is this function has only one parameter that is string, which means the character array has a null character at the end.

d. `int countWordsStartingWithCapitalLetter (char *a)`

String may have multiple words, count all the words starting with capital letters. Note that there may be capital letters in between the words, ignore them.

e. `char* replaceCommaWithSpace (char *a)`

String may have multiple words, separated with comma, find and replace all comma characters with space characters. At the end return string, so that function may be called with a print statement.

f. `char* removeExtraSpacesInTarget (char *src, char *trg)`

String may have multiple words, separated with spaces and there may be extra spaces in between words like “This is interesting”. Source string & target string is of same size, copy all characters of source string into target string except extra spaces. See example:

Source String: This is heavy. You can’t lift this alone.
Target String: This is heavy. You can’t lift this alone.

g. `char* removeExtraSpaces (char *s)`

String may have multiple words, separated with spaces and there may be extra spaces in between words like “This is interesting”. Remove extra spaces from given string:

String before calling function: This is heavy. You can’t lift this.
String after calling function: This is heavy. You can’t lift this.

<----------->

3. Write following functions related to 2D integer array :

a. `void print2D (int a[][5], const int ROWS)`

2D array with 5 columns and variable number of rows. Write a nested loop to print 2D arrays in tabular form that is row/ column form.

b. `void printEvenOddCount (int a[][10], const int ROWS)`

2D array with 10 columns and variable number of rows. Write program to print and count number of even & odd values in each row, the output should be like:

Row 1 has 6 even and 4 odd values

Row 2 has 3 even and 7 odd values

...

c. `void print2D (int **a, const int ROWS, const int COLS)`

2D array with variable size columns and variable size rows. Write a nested loop to print 2D arrays in tabular form that is row/ column form.

d. `void printPositiveValuesOnly (int **a, const int ROWS, const int COLS)`

2D array with variable size columns and variable size rows having both negative & positive values. Write a nested loop to print only `positive` values in tabular form that is row/ column form.

e. `int findMaxElement (int **a, const int ROWS, const int COLS)`

Find maximum value `from the complete` 2D array and return.

f. `void printMaxElementsRowColumn (int **a, const int ROWS, const int COLS)`

Find maximum value from complete 2D array and print row and column of maximum value.