National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Programming Fundamentals**	**Course Code:**	**CS1002**
Program:	**BS(Computer Science)**	**Semester:**	**Fall 2023**
Duration:	**N/A**	**Total Marks:**	**150**
Due Date:		**CLO:**	
Section:	**1K**	**Page(s):**	**4**
Exam:	**Assignment 7**	**Roll No.**	

Instructions:

- *Late submissions will lead to negative marking and submissions after 20 hours past the due time will not be accepted.*
- *This is an individual assignment and the solution submitted must be your own.*
- *Any sort of plagiarism will be dealt with seriously and may lead to severe consequences including negative marking.*
- *Submit .cpp files named as XXL-XXXX_Q#X.cpp*
  i.e [your roll number]_[question number].cpp

QUESTION#1: (10)

Given a 2D array, find a substring in the given array, return true in case of existence and false otherwise. The substring can be in all three directions, horizontal, vertical, and diagonal.

Sample Output:

- **Input:** words = [[a, s, e, t, w],
  [f, e, r, t, y],
  [f, w, i, n, k],
  [a, r, q, o, p],
  [n, j, t, z, l]]
  name = "wink"
  Output: true

- **Input:** words = [[u, s, e, t, w],
  [e, m, r, q, y],
  [q, w, t, u, k],
  [p, r, q, i, p],
  [q, j, t, z, r]]
  name = "quiz"
  Output: true

QUESTION#2: (10)

You are given two 2D integer arrays nums1 and nums2.

- nums1[i] = [id_i, val_i] indicate that the number with the id id_i has a value equal to val_i.
- nums2[i] = [id_i, val_i] indicate that the number with the id id_i has a value equal to val_i.

Each array contains unique ids and is sorted in ascending order by id.
Merge the two arrays into one array that is sorted in ascending order by id, respecting the following conditions:
Only ids that appear in at least one of the two arrays should be included in the resulting array.
Each id should be included only once and its value should be the sum of the values of this id in the two arrays. If the id does not exist in one of the two arrays then its value in that array is considered to be 0.
Return the resulting array. The returned array must be sorted in ascending order by id.

(Hint: You only need to input number of rows for both the arrays as number of columns is constant i.e. 2)

Sample Output:

- **Input:** `nums1 = [[1,2],[2,3],[4,5]], nums2 = [[1,4],[3,2],[4,1]]`
  Output: `[[1,6],[2,3],[3,2],[4,6]]`
  Explanation: The resulting array contains the following:
  - id = 1, the value of this id is 2 + 4 = 6.
  - id = 2, the value of this id is 3.
  - id = 3, the value of this id is 2.
  - id = 4, the value of this id is 5 + 1 = 6.

QUESTION#3: (10)

Given a 2D grid of size m x n and an integer k. You need to shift the grid k times.
In one shift operation:

- Element at grid[i][j] moves to grid[i][j + 1].
- Element at grid[i][n - 1] moves to grid[i + 1][0].
- Element at grid[m - 1][n - 1] moves to grid[0][0].

Return the 2D grid after applying shift operation k times.

Sample Output:

- **Input:** `grid = [[1,2,3],[4,5,6],[7,8,9]], k = 1`
  Output: `[[9,1,2],[3,4,5],[6,7,8]]`

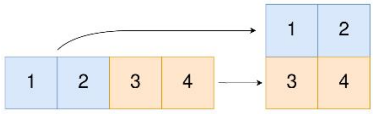
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 9 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$$

QUESTION#4: (10)

You are given a 0-indexed 1-dimensional (1D) integer array original, and two integers, m and n. You are tasked with creating a 2-dimensional (2D) array with m rows and n columns using all the elements from original.
The elements from indices 0 to n - 1 (inclusive) of original should form the first row of the constructed 2D array, the elements from indices n to 2 * n - 1 (inclusive) should form the second row of the constructed 2D array, and so on.
Return an m x n 2D array constructed according to the above procedure, or an empty 2D array if it is impossible.

Sample Output:

- **Input:** `[1,2,3,4], m = 2, n = 2`
  Output: `[[1,2],[3,4]]`



- **Input:** `original = [1,2], m = 1, n = 1`
  Output: `[]`
  Explanation: There are 2 elements in original.
  It is impossible to fit 2 elements in a 1x1 2D array, so return an empty 2D array.

QUESTION#5: (10)

Given a 2D integer array matrix, return *the* **transpose** *of* matrix. The **transpose** of a matrix is the matrix flipped over its main diagonal, switching the matrix's row and column indices.

Sample Output:

- **Input:** `[[1,2,3],[4,5,6],[7,8,9]]`
  Output: `[[1,4,7],[2,5,8],[3,6,9]]`

QUESTION#6: **(10)**

Given an m x n matrix of **distinct** numbers, return *all* **lucky numbers** in the matrix in the form of an array. A **lucky number** is an element of the matrix such that it is the minimum element in its row and maximum in its column. (The output array can be of size m*n)

Sample Output:

- **Input:** `[[3,7,8],[9,11,13],[15,16,17]]`
  Output: `[15]`

QUESTION#7: **(10)**

Given a square matrix mat, return the sum of the matrix diagonals.

Sample Output:

- **Input:** `mat = [[1,2,3],`
  `[4,5,6],`
  `[7,8,9]]`
  Output: `25`
  Explanation: `Diagonals sum: 1 + 5 + 9 + 3 + 7 = 25`
  `Notice that element mat[1][1] = 5 is counted only once.`

QUESTION#8: **(10)**

Write a user defined function named Upper-half() which takes a two dimensional array A, with size N rows and N columns as argument and prints the upper half of the array.

Sample Output:

- **Input:**
  `[[2,3,1,5,0],[7,1,5,3,1],[2,5,7,8,1],[0,1,5,0,1],[3,4,9,1,5]]`
  Output:

```
2 3 1 5 0
  1 5 3 1
    1 7 8
      0 1
        5
```

QUESTION#9: **(10)**

Write a program to add two matrices A and B of size m x n.

Sample Output:

- **Input:** `mat1 = [[1,2],[3,4]], mat2 = [[5,6],[7,8]]`
  Output: `[[6,8],[10,12]]`

QUESTION#10: **(10)**

Write a program to multiply matrices A and B of order n x l and l x m.

Sample Output:

- **Input:** `mat1 = [[1,2],[3,4]], mat2 = [[3,4],[5,1]]`
  Output: `[[13,6],[29,20]]`

QUESTION#11: **(10)**

Write a function in C++ which accepts a 2D array of integers and its size as arguments and displays the elements of middle row and the elements of middle column.
The 2D Array must be a square matrix with odd dimension i.e. 3x3, 5x5, 7x7 etc.

Sample Output:

- **Input:** `[[1,2,3],[4,5,6],[7,8,9]]`
  Output: `Middle Row: 4,5,6  Middle column: 2,5,8`

QUESTION#12: (10)

Sort the strings in a 2D character array in lexicographically ascending order.

Sample Output:

- **Input:** `{"banana", "apple", "grape", "guava", "cherry"}`
  Output: `{"apple", "banana", "cherry", "grape", "guava"}`

QUESTION#13: (5)

Write a function to check if each row in a 2D character array is a palindrome. A 2D array is a palindrome if all its rows are palindrome.

Sample Output:

- **Input:** `{"madam", "hello"}`
  Output: `No`
- **Input:** `{"level", "deed"}`
  Output: `Yes`

QUESTION#14: (10)

Write a function to concatenate the elements from each column of a 2D character array and return the result in a new 2D array.

Sample Output:

- **Input:** `{"abc", "def", "ghi"}`
  Output: `{"adg", "beh", "cfi"}`

QUESTION#15: (15)

Write a function that takes an m x n matrix and return all elements of the matrix in spiral order.

Sample Output:

- **Input:** `[[1,2,3,4],[5,6,7,8],[9,10,11,12]]`
- **Output:** `[1,2,3,4,8,12,11,10,9,5,6,7]`