
National University
of computer and emerging sciences

# CL-1004
# Object Oriented Programming- Lab
# Spring' 2023

# BS-SE

# Lab Manual 03

## Problem 1:

Write a function that receive arguments: two 2D pointer p, number of rows and number of columns of both matrix and return the product of both matrix.

**Prototype:** int** **Product** (int ** matrix1, int ** matrix2, int rowM1, int colM1, int rowM2, int colM2)

## Problem 2:

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

e.g.,

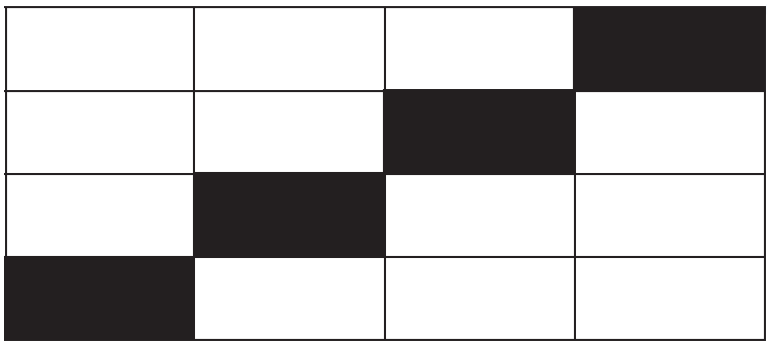
| | | |
|---|---|---|
| 2 3 1 5 0 | | 2 3 1 5 0 |
| 7 1 5 3 1 | | 1 5 3 1 |
| 2 5 7 8 1 | Output will be: | 7 8 1 |
| 0 1 5 0 1 | | 0 1 |
| 3 4 9 1 5 | | 5 |

**Prototype:** int** **Upper_half** (int** Array, int rows, int columns)

## Problem 3:

Write a function that receives three arguments: (i) a 2D pointer p; (ii) number of rows sizeA; (iii)number of columns sizeB and return the sum of the diagonal (boundary rows and columns which are shown as black area in the following figure) of the 2D array. Remember you have to calculate sum of values of the array shown as black area in the given figure only and return that sum.

**Prototype:** int **calDiagonal**(int **p, int sizeA, int sizeB)



## Problem 4:

Write a program that can be used to assign seats for a commercial airplane. The airplane has 13 rows, with six seats in each row. Rows 1 and 2 are first class, rows 3 through 7 are business class,

and rows 8 through 13 are economy class. Your program must prompt the user to enter the following information:

- Ticket type (first class, business class, or economy class)
- Desired seat

## Problem 5:

Write a recursive function that takes a number as input and check whether the number is prime or not.

**Prototype:** bool **PrimeNumber** (int n)

## Problem 6:

Write a recursive function that takes a character array of size **n** and then return character array with first **k** words in reverse order and remaining words in the same order.

For Example:

Input: “I AM GOOD, How are you” (if k= 3)

Output: “GOOD AM I, How are you”

**Prototype**: char* **kWOrdsReverser** (char* ptr, int k, int n)

## Problem 7:

Write a recursive procedure called findWordLocation that takes two arguments, a word and a sentence. It should return a number indicating where in the sentence that word can be found. If the word isn't in the sentence, return 0. If the word appears more than once, return the location of the first appearance.

**Prototype**: int **findWordLocation**(char* sentence, char* word)

## Problem 8:

Write a recursive function that takes array as input and return sorted array.

**Prototype:** int* **ArraySorting**(int* array, int n)