National University of Computer and Emerging Sciences, Lahore Campus

Course:	PF Lab	Course Code:	CL 1002
Program:	BS-DS, BS-SE	Semester:	Fall 2021
Duration:	2 Hours	Total Marks:	100
Date:	22-Jan-2022	Weight	40 %
Section:	BCS(1H,1J,1K,1L,1M,1N) BDS(1A)	Page(s):	
Exam:	Lab Final Term	Reg. No.	

Read below Instructions Carefully:

- Understanding the question statement is also part of the exam, so do not ask for any clarification. In case of any ambiguity, make suitable assumptions.
- You have to complete exam in 2 hrs. No extra time will be given for submission.
- For submission, submit all .cpp files (containing function and main) named as 21L-1122-Q1.cpp, 21L-1122-Q2.cpp, etc.
- Submission path: \\Cactus\Xeon\PF Final Exam Submissions\Your specific section
- Submit all questions .cpp file on Google Classroom under assignment titled as **PF- Lab Final term Submission.** (**Don't create zip file**)
- Your code should be intended and commented properly. Use meaningful variable names.
- It is your responsibility to save your code from being copied. All matching codes will be considered cheating cases. PLAGIARISM will result in forwarding of case to Disciplinary Committee and negative marks in Final term.

(Q-1) Encryption: [25 + 15 + 10 Points]

Encryption is a classical method for protecting messages. Typical encryption schemes divide a given message into blocks and then encrypt each block using a pre-defined **method**.

One such **method** of encrypting a message block of size at most 25 works as follows.

The message is placed in a grid of 5 x 5 size **row wise** and then is read column wise to create the confidential message.

For example if we need to encrypt the message "**Protect this message well**" it is placed in a 5x5 grid as shown below (the character . specifies space here)

P	r	o	t	e
c	t	.	t	h
i	s	.	m	e
s	s	a	g	e
.	w	e	l	l

next read it column wise to create the encrypted message "**Pcis.rtsswo..aettmgleheel**"

To decrypt the message we just place the encrypted message block in the 5x5 grid column wise and read it row wise to get the decrypted/original message

During encryption, if the message block has length smaller than 25 then the remaining gri filled with special symbol **$**. So to encrypt the message **"Protect this message"** the grid looks a follows

P	r	o	t	e
c	t	.	t	h
i	s	.	m	e
s	s	a	g	e
$	$	$	$	$

And hence the encrypted message becomes **"Pcis$rtss$o..a$ttmg$ehee$** while this special symbol is ignored during decryption.

Part a) Write a C++ function that takes a character array containing a null terminated string **MessageBlock** (text block to be encrypted), and a second character array **Encrypted_Message** as parameters. It should encrypt and return the encrypted message to the calling function by using the Encrypted_Message array.

void Encrypt(char MessageBlock[], char Encrypted_Message[]);

Example(s): The following Table shows inputs and the corresponding outputs for three different test cases.

Message	Encrypted_Message
"middle-Outz"	**"mez$$i-$$$dO$$$lt$$$"**
"Always-see-the-right-Side"	**"As-r-l-siSwsegiaeehdye-te"**
"He is a friend indeed"	**"H.iideaen$..nd$ifde$sr.e$"**

Note: All test input will be a valid ASCII string.

Part b) Write a C++ function that takes an encrypted message block of size exactly 25 and return the decrypted message to the calling function. The proposed prototype of the function is

void Decrypt(char Encrypted_Message[], char Message[]);

Part c) Write a main function to test your encryption and decryption function. The main must ask the user to enter a message/string of length at most 25 and then display the original message, the encrypted message and the decrypted message on console.

Points:

A position (i, j) in a m x n 2D array A[][] of integers is said to be a balance point of the array if **[35 + 15 Points]**

A[i][0] + A[i][1] + ... + A[i][j-1] is equal to A[i][j+1]+A[i][j+2]+ ... + A[i][n]

AND

A[0][j] + A[1][j]+ ... + A[i-1][j] is equal to A[i+1][j] + A[i+2][j]+ ... + A[m][j]

For example the following 5 x 5 array has two balance points at positions (2, 2) and (3, 3)

1	0	1	-1	2
1	5	3	1	2
1	9	3	3	7
12	5	2	7	19
10	20	2	3	15

Similarly the following 5 x 5 array has no balance point.

1	2	3	4	5
6	7	8	9	0
1	2	3	4	5
6	7	8	9	0
1	2	3	4	5

Part a) Write a C++ function that has a 5 x 5 array as an input parameter and return the count of balance points in that array. This function must also display each balance point on a new line.

If the first array given above is passed to the function then function must return a value 2 to the calling function and also display the following two lines

(2, 2)

(3, 3)

Part b) Write a main function that declares and inputs numbers in a 5 x 5 array of integers and then pass it to the function of part a to compute the total balance points in that array. The main function must display this information in a proper format on console