|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
| C:\Users\saif\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\final design.jpg | **Course:** | **Programming Fundamentals** | **Course Code:** | **CS-118** |
| **Program:** | **BS (Computer Science)** | **Semester:** | **Spring 2020** |
| **Duration:** | **(180+15) Minutes** | **Total Marks:** | **70** |
| **Paper Date:** | **09-July-2020** | **Weight** | **50%** |
| **Section:** | **All** | **Page(s):** | **8** |
| **Exam:** | **Final Exam** |  |  |
|  |  | | | |
| **Instruction/Notes:** | 1. Solve this exam on an A4 size paper or assignment sheets or any presentable paper. 2. You must provide solution of the exam in this document, past clear screen shots of your solution only, any ambiguous/unclear answer will be considered wrong. You may provide your solution in a single pdf document but make sure solution of each question must be under proper heading of question number and respective part. 3. Submit rough work at the end of every question with proper heading. 4. The exam is open book and notes, but you are not allowed to take help from anyone.Use of any other unfair means is strictly prohibited. This may lead to an **F**grade in course**.** 5. Name of the solution document must be as **Final\_Exam\_Section\_X [L19-1001],** otherwise no marks will be awarded. 6. Late submissions will not be allowed. So, make sure you have a back-up ready in case of power failure. 7. **Submit your solution on Google classroom. If you are facing any problem in uploading your work on Google class room then send your work via email.** 8. Questions during exam are not allowed. Take reasonable assumption if there's any ambiguity. 9. **180 Minutes** are allocated to solve the exam and **extra 15 mints** will be used to take pictures, make solution file and submission of the solution. 10. **Kindly read the problem completely before starting to solve it, otherwise there may a possibility to come up with wrong solution.** | | | |

**Question 1: [5 points]**

Write a user defined function named “**findTwosComplement**” in C++, which will find 2’s complement of a signed number. Input must be passed by reference to the function and it will return the answer. There is no need to call this function anywhere.

You can use following operators only:

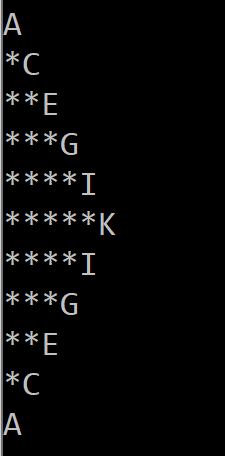
<<**>>** ~ & ^ |

**Question 2:[12 marks]**:

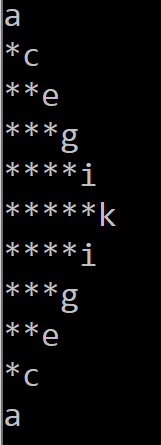
Write C++ program to take input in variables "**n"** and "**L"** from user and print following character pattern .For reference, ASCII chart is also given below.

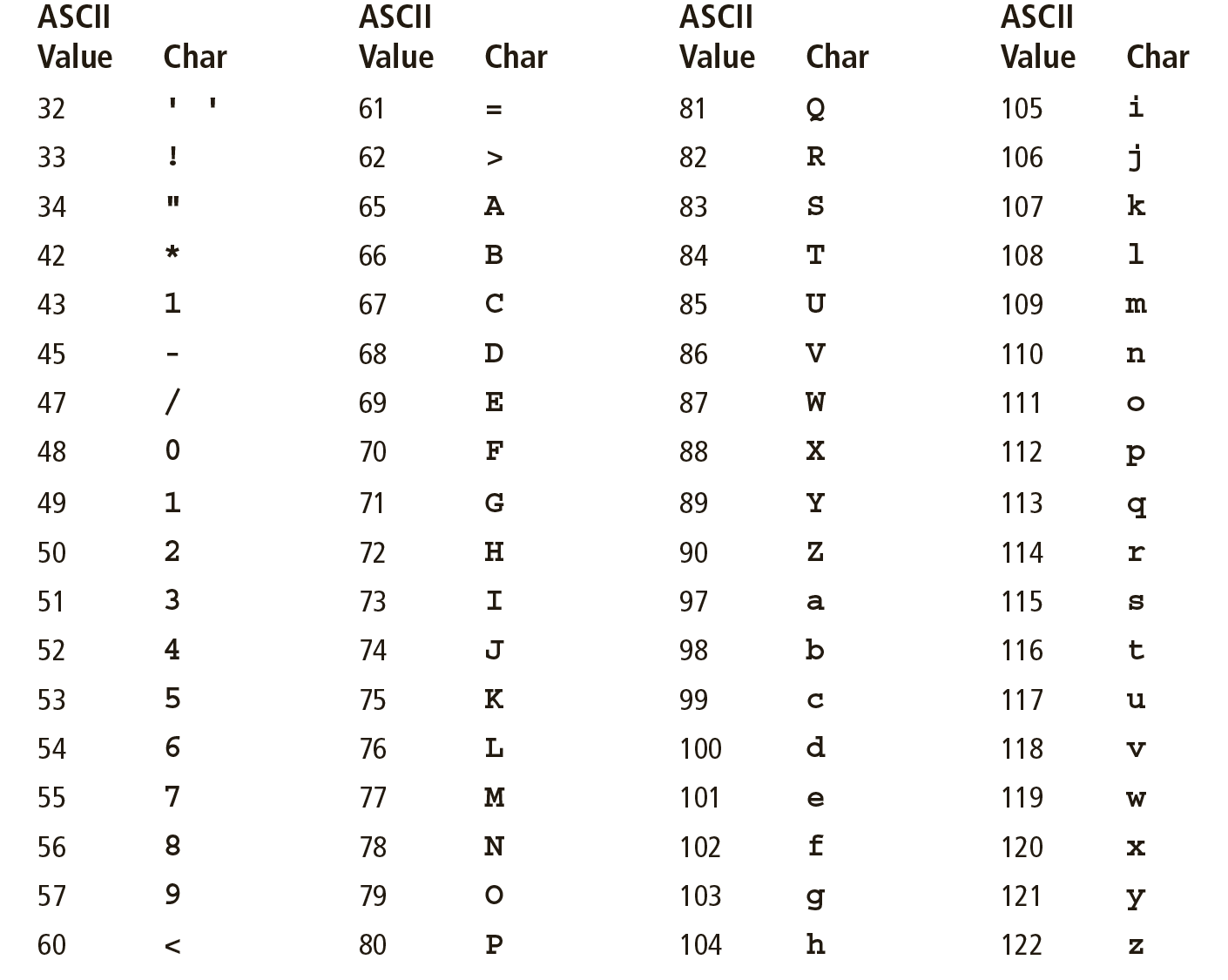
Sample inputs and output are shown below

n=5, L=’A’



For sample inputs: n=5, L=’a’





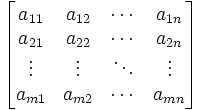
**Question 3: [30 points]**

Read followingdefinitionscarefully so that problem statement of this question can be understood clearly.

**Please note that the scope of these definitions is limited to this question only**.

1. **IMatrix:**

An arrangement of elements i.e. numbers, alphabets, or functions in the form of rows and columns as a rectangular array is called an**IMatrix**. A general representation of an **IMatrix** is given below



1. **Order of anIMatrix:** Order of anIMatrix is represented as **number of rows X number of columns.** For example, if an **IMatrix**contains 5 rows and 7 columns then its order will be 5 X 7, similarly for an N row and M column matrix, order will be N X M.
2. **Main/Primary Diagonal of an IMatrix:** A diagonal in which elements run from lower left entry to upper right entry is called Main/Primary diagonal of an IMatrix. Highlighted elements in following IMatrix belong to Primary diagonal.

A picture containing object, clock

Description automatically generated

1. **Secondary Diagonal of an IMatrix:**The Secondary diagonal of an **IMatrix** consists of those elements that lie on the diagonal that runs from top left position to bottom right position.Highlighted elements of following **IMatrix** form the secondary diagonal of it.

A picture containing clock, drawing

Description automatically generated

1. **Triangular Joint:** A triangular joint is a submatrix with following properties

* It is an **IMatrix** of order 3X3.
* All rows of such matrix are completely filled. Except second row.
* Second row consists of one element only which will lie at the central position of the row. This element is also called center of the joint.

An example of triangular joint is given below:

A B C

D

E F G

1. **General Triangular Joint (GTJ):** A Triangularjoint withfollowing properties is a **GeneralTriangularJoint**:
2. All elements of the elements in this Joint are alphabets.
3. There is always small alphabet at the center of the joint.
4. Elements of main diagonal are in increasing order from top to bottom.
5. Elements of secondary diagonal are in decreasing order from top to bottom.
6. Elements that lie in the column where center of the jointlies, have a relation of following kind with centralelement:
   1. Elements which lie in the rows whose row number is less than the row of the center, contains those alphabets which occur before (in alphabetical order) the element at center.
   2. Elements which lie in the rows whose row numbers are greater than the row number of the center, contains those alphabets which occur after (in alphabetical order) the element at center.
7. Small and Large alphabets will be considered two different sets i.e. we cannot call ‘**A’** to be smaller or larger than**‘a’**, both have same signification in the context of this problem, whereas**‘B’** is smaller than **‘E’** and **‘m’** is greater than **‘N’.**

An example of General Triangular Joint is given below:

|  |  |  |
| --- | --- | --- |
| O | C | D |
| - | m | - |
| P | r | f |

In this scenario:

1. All elements of the joint are alphabets.
2. Element at the center of the triangular joint is small alphabet i.e. **‘m’**.
3. Elements of primary diagonal are in increasing order i.e. **D <m <P.**
4. Elements of secondary diagonal are in decreasing order i.e. **O>m>f**
5. **‘C’** and **‘r’** are the only elements which lie in the same column where center of joint i.e. **‘m’** lies.
   1. **‘C’**is the only element which lies in row number less than row number of **‘m’** and it comes earlier than **‘m’** in alphabetical order**.**
   2. **‘r’**is the only element which lies in row number greater than row number of **‘m’** and it comes after**‘m’** in alphabetical order**.**

Hence it is meeting all the conditions of GeneralTriangular Joint so it is aGTJ.

1. **Special Triangular Joint (STJ):** A GTJ in which at least4 upper case letters exist regardless of their positions is called aSTJ.For example,GTJ considered above contains 4 upper letters (D, P, O, C), so it is a Special Triangular Joint.

Write a C++ function named ‘**findTriangularJoints’**which will accept an N X M (Where N is number of rows and M is number of columns)**IMatrix** as an input.

1. **[10 points]**Count all **GeneralTriangular**Joints and return it to the calling function.
2. **[5 points]**Display all GTJs on standard output stream.
3. **[10 points]**Find % STJs and return it to the calling function. Formula to calculate %STJ is given below

% STJs =

1. **[5 points]**Call this function in main and show count of GTJs and percentage of STJs for sample input given below.

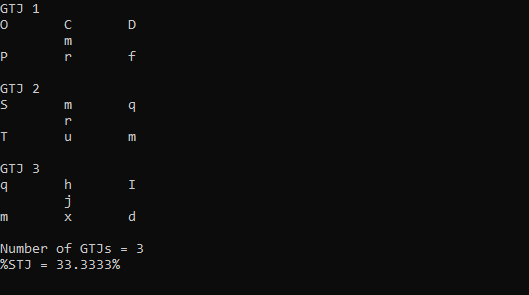
**Coding Requirements:** You code must fulfill, following coding requirements**otherwise no credit will be given:**

1. Only 1 loop is allowed inside the other to form nested one, and only one nested loop of this type can be used.
2. Only **iostream** library is available to use in this program, you can use any user defined function of this library to minimize your coding time.
3. **String**data type is not allowed to use in this program.
4. There is no space in RAM to declare an extra array, so you can use **only one array in the whole program.**
5. Your function must be generic enough that will accept matrix of any order as an input.
6. Validations and conditions of all types must be properly handled in the code written.
7. You have no need to take input from user inside **findTriangularJoints’**function, however you may initialize the array with fixed values in **main** function.

Sample Input:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| O | C | D | B | A |
| S | M | Q | H | I |
| P | R | F | J | Z |
| T | U | M | X | d |
| S | M | E | H | x |

Sample Output:



**Question 4:[23 marks]**

Ahmed and Iffrah are playing a game "Search Word". In this game there are sequence of characters stored in 8 X 25 table and this table is stored in "characters.txt" file . Both Ahmed and Iffrah have to search particular words available in "WordstoSearch.txt". There are 4 words of length 6 in "WordstoSearch.txt" . Words can be searched horizontally, vertically and diagonally.

The one who search more words from characters.txt won the game, but Iffrah is much younger than Ahmed, so we have decided to help Iffrah by developing a program that will tell about Iffrah total number of words found in "characters.txt".

Your task is to develop a program to search words in "characters.txt" and count them . The list of words to be searched is available in "WordstoSearch.txt". Your program should follow divide and conquer rule.

Following functions should be there in the program.

1. Load "characters.txt" in 2d array named "characters" using function named "LoadCharacters".**[2.5 marks]**

2. Load "WordstoSearch.txt" in 2d array named "WordstoSearch" using function named "LoadWordstoSearch".**[2.5 marks]**

3. Search word horizontally using function named "SearchHorizontally".**[5 marks]**

4. Search word vertically using function named "SearchVertically ".**[5 marks]**

5. Search word diagonally using function named "SearchDiagonally ".**[5 marks]**

6. Return the total numbers of words found in "characters.txt" using function "ReturnTotalCount".**[3 marks]**

Following are the input files:

**Input**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **h** | **e** | **l** | **l** | **o** | **w** | **u** | **c** |
| **h** | **s** | **d** | **s** | **d** | **h** | **j** | **r** |
| **k** | **a** | **j** | **d** | **k** | **s** | **j** | **e** |
| **j** | **j** | **d** | **s** | **j** | **d** | **d** | **e** |
| **n** | **d** | **s** | **j** | **s** | **d** | **b** | **p** |
| **o** | **k** | **j** | **d** | **k** | **j** | **s** | **y** |
| **j** | **w** | **d** | **h** | **j** | **s** | **d** | **h** |
| **k** | **k** | **l** | **k** | **j** | **d** | **k** | **s** |
| **h** | **e** | **l** | **l** | **o** | **y** | **u** | **k** |
| **h** | **s** | **d** | **s** | **d** | **h** | **j** | **e** |
| **k** | **a** | **j** | **d** | **k** | **s** | **j** | **e** |
| **j** | **j** | **d** | **s** | **j** | **d** | **d** | **P** |
| **n** | **d** | **s** | **j** | **s** | **d** | **b** | **i** |
| **o** | **k** | **j** | **d** | **k** | **j** | **s** | **j** |
| **j** | **w** | **d** | **h** | **j** | **s** | **d** | **h** |
| **k** | **k** | **l** | **k** | **j** | **d** | **k** | **s** |
| **h** | **e** | **l** | **l** | **o** | **y** | **u** | **k** |

**characters.txt [there won’t be any space between the letters in the file.Table is just for more clarity]**

|  |
| --- |
| Hellow  creepy  sleepy  slowly |

**Wordstosearch.txt**

And **Output** of the program should be:

2 words