# 2D Array Lab

## Problem 1:

In this lab you will be given a table of letters and a set of words to find. Your job is to write a program that reads the table and finds the words in the table. The output should be the location (row, column) and direction of each of the words given to look for, or it should indicate that the word could not be found. Words can come in the table horizontal (left to right), vertical (top, down), or diagonal (top-left to top-right). See example file:

Size 8, 8

| A | B | N | E | K | Z | U | X |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N | V | C | I | P | W | K | L |
| C | I | E | Z | T | E | F | R |
| B | A | F | I | T | P | T | G |
| U | I | T | S | Y | M | L | H |
| T | R | A | I | N | J | H | O |
| J | B | R | K | E | W | D | S |
| A | K | Z | I | Q | P | F | T |

Train  
Pet  
Dog  
Cat  
Ghost

Output:

Row Colum Direction  
Train: 5 0 Horizontal   
Pet: 1 4 Diagonal  
Dog: Not Found  
Cat: 2 0 Diagonal  
Ghost: 3 7 Vertical

Things to note: The width and height may be different, so the letter table may be 7 by 9 or 4 by 18. The max size of the letter table is 24 (row or column) and this should be coded in as a const variable (in case it changes). There is however no limit to the amount of words that you may have to search for.

Bonus: In addition to the current output also output the letter table, but this time edit the color (of style) of the font to highlight the words in the table:

| A | B | N | E | K | Z | U | X |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N | V | C | I | P | W | K | L |
| C | I | E | Z | T | E | F | R |
| B | A | F | I | T | P | T | G |
| U | I | T | S | Y | M | L | H |
| T | R | A | I | N | J | H | O |
| J | B | R | K | E | W | D | S |
| A | K | Z | I | Q | P | F | T |

## Problem 2:

Read in a file of numbers that form a sudoku puzzle. No we are not filling in the puzzle, this is already done for you. Your job is to output whether or not the puzzle is correctly done or not. A sudoku puzzle is 9 rows by 9 columns. Each row and column has to have the digits 1 through 9 in them. Additionally, Each block of 9 (3 by 3) also has to have all 9 digits in it. Your job again is to read a file like the one given here, and output if it is a correctly solved puzzle:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 3 | 5 | 2 | 6 | 9 | 7 | 8 | 1 |
| 6 | 8 | 2 | 5 | 7 | 1 | 4 | 9 | 3 |
| 1 | 9 | 7 | 8 | 3 | 4 | 5 | 6 | 2 |
| 8 | 2 | 6 | 1 | 9 | 5 | 3 | 4 | 7 |
| 3 | 7 | 4 | 6 | 8 | 2 | 9 | 1 | 5 |
| 9 | 5 | 1 | 7 | 4 | 3 | 6 | 2 | 8 |
| 5 | 1 | 9 | 3 | 2 | 6 | 8 | 7 | 4 |
| 2 | 4 | 8 | 9 | 5 | 7 | 1 | 3 | 6 |
| 7 | 6 | 3 | 4 | 1 | 8 | 2 | 5 | 9 |

Again each column and row is 1 through 9. But each block of 3 is as well. Lets look at the top left corner for example:

|  |  |  |
| --- | --- | --- |
| 4 | 3 | 5 |
| 6 | 8 | 2 |
| 1 | 9 | 7 |

1 – 9 is correctly assigned here as it is in each block of 3. You will have to break the arrays down into 9 individual blocks of 3 by 3 in order to check that each number is there. Again, the only output for this problem is a yes or no as to whether the puzzle is valid or not.

Bonus: Like before, in addition to the current output also output the table again, but this time edit the color (of style) of the font to highlight any errors in the table. If two numbers are the same on a row or column highlight them to show them as erroneous.