

# CSCI 174

## Term Project

Mukhasir Shah Syed  
109137268

For completing my personal term project, I have chosen the below mentioned hard problems and coded them in C# language using ASP.NET 4 Framework.

Challenges:

1. Telephone Words
2. Spiral Printing

## Telephone Words:

Challenge Description:

### Challenge

[VIEW ALL CHALLENGES](#)

#### TELEPHONE WORDS

##### CHALLENGE DESCRIPTION:

Given a 7 digit telephone number, print out all the possible sequences of letters that can represent the given telephone number. Note that in a standard 12 key pad, 0 and 1 do not have any letters associated with them. They are to be treated as such, i.e. a 0/1 in the telephone number will be retained in the final word as well. You may use the following mapping between numbers and characters:

```
0 => 0
1 => 1
2 => abc
3 => def
4 => ghi
5 => jkl
6 => mno
7 => pqrs
8 => tuv
9 => wxyz
```

Given a 7 digit telephone number, print out all the possible sequences of letters that can represent the given telephone number. Note that in a standard 12 key pad, 0 and 1 do not have any letters associated with them. They are to be treated as such, i.e. a 0/1 in the telephone number will be retained in the final word as well. You may use the following mapping between numbers and characters:

0 => 0

1 => 1

2 => abc

3 => def

4 => ghi

5 => jkl

6 => mno

7 => pqrs

8 => tuv

9 => wxyz

### Program Input and Output Values:

Input: The input file contains 7-digit telephone numbers, one per line. (Example: 4155230)

Output:

g1jjad0,g1jjae0,g1jjaf0,g1jjbd0,g1jjbe0,g1jjbf0,g1jjcd0,g1jjce0,g1jjcf0,g1jkad0,g1jkae0,g1jkaf0,g1jkbd0,g1jkbe0,g1jkb0,g1jkcd0,g1jkce0,g1jkcf0,g1jlad0,g1jlae0,g1jlaf0,g1jlbd0,g1jlbe0,g1jlbf0,g1jlcd0,g1jlce0,g1jlc0,g1kjad0,g1kjae0,g1kjaf0,g1kjbd0,g1kjbe0,g1kjbf0,g1kjcd0,g1kjce0,g1kjcf0,g1kkad0,g1kkae0,g1kkaf0,g1kkbd0,g1kkbe0,g1kkbf0,g1kkcd0,g1kkce0,g1kkcf0,g1klad0,g1klae0,g1klaf0,g1klbd0,g1klbe0,g1klbf0,g1klcd0,g1klce0,g1klcf0,g1ljad0,g1ljae0,g1ljaf0,g1ljbd0,g1ljbe0,g1ljbf0,g1ljcd0,g1ljce0,g1ljcf0,g1lkad0,g1lkae0,g1lkaf0,g1lkbd0,g1lkbe0,g1lkbf0,g1lkcd0,g1lkce0,g1lkcf0,g1llad0,g1llae0,g1llaf0,g1llbd0,g1llbe0,g1llbf0,g1llcd0,g1llce0,g1llcf0,h1jjad0,h1jjae0,h1jjaf0,h1jjbd0,h1jjbe0,h1jjbf0,h1jjcd0,h1jjce0,h1jjcf0,h1jkad0,h1jkae0,h1jkaf0,h1jkbd0,h1jkbe0,h1jkb0,h1jkcd0,h1jkce0,h1jkcf0,h1jlad0,h1jlae0,h1jlaf0,h1jlbd0,h1jlbe0,h1jlbf0,h1jlcd0,h1jlce0,h1jlc0,h1kjad0,h1kjae0,h1kjaf0,h1kjbd0,h1kjbe0,h1kjbf0,h1kjcd0,h1kjce0,h1kjcf0,h1kkad0,h1kkae0,h1kkaf0,h1kkbd0,h1kkbe0,h1kkbf0,h1kkcd0,h1kkce0,h1kkcf0,h1klad0,h1klae0,h1klaf0,h1klbd0,h1klbe0,h1klbf0,h1klcd0,h1klce0,h1klcf0,h1ljad0,h1ljae0,h1ljaf0,h1ljbd0,h1ljbe0,h1ljbf0,h1ljcd0,h1ljce0,h1ljcf0,h1lkad0,h1lkae0,h1lkaf0,h1lkbd0,h1lkbe0,h1lkbf0,h1lkcd0,h1lkce0,h1lkcf0,h1llad0,h1llae0,h1llaf0,h1llbd0,h1llbe0,h1llbf0,h1llcd0,h1llce0,h1llcf0,i1jjad0,i1jjae0,i1jjaf0,i1jjbd0,i1jjbe0,i1jjbf0,i1jjcd0,i1jjce0,i1jjcf0,i1jkad0,i1jkae0,i1jkaf0,i1jkbd0,i1jkbe0,i1jkb0,i1jkcd0,i1jkce0,i1jkcf0,i1jlad0,i1jlae0,i1jlaf0,i1jlbd0,i1jlbe0,i1jlbf0,i1jlcd0,i1jlce0,i1jlc0,i1kjad0,i1kjae0,i1kjaf0,i1kjbd0,i1kjbe0,i1kjbf0,i1kjcd0,i1kjce0,i1kjcf0,i1kkad0,i1kkae0,i1kkaf0,i1kkbd0,i1kkbe0,i1kkbf0,i1kkcd0,i1kkce0,i1kkcf0,i1klad0,i1klae0,i1klaf0,i1klbd0,i1klbe0,i1klbf0,i1klcd0,i1klce0,i1klcf0,i1ljad0,i1ljae0,i1ljaf0,i1ljbd0,i1ljbe0,i1ljbf0,i1ljcd0,i1ljce0,i1ljcf0,i1lkad0,i1lkae0,i1lkaf0,i1lkbd0,i1lkbe0,i1lkbf0,i1lkcd0,i1lkce0,i1lkcf0,i1llad0,i1llae0,i1llaf0,i1llbd0,i1llbe0,i1llbf0,i1llcd0,i1llce0,i1llcf0

### Procedure to Solve the challenge:

When we type a telephone number, we press a key only once so only one value will be considered when a key is pressed. For example, when “4” key is pressed it has “ghi” on it, when

forming a word we have to consider 3 cases of having 'g','h','i' so that all case of the key are considered.

In this challenge,

1. Read input from file or arguments.
2. Convert the string of numbers into characters.
3. Declare a List to store the permuted or combinations obtained from the input telephone number.
4. Use switch case to get into the specific functionality for each number.
5. To add the word values into list we call "MakeLoop(`ref List<string> Telephone_Values, string value`)" method.
6. MakeLoop method takes List which stores all combinations and specific value for the key
  - a. In MakeLoop method, firstly a check is done on the List (Telephone\_Values) if it has values or not.
  - b. If it doesn't have values in the List then value for respective number is added to list.
  - c. While adding respective value to list we check if the value for the respective number is single character[0 or 1] or has multiple characters(string)[ abc or ghi ].
  - d. If single character then that will be added directly to List.
  - e. If multiple characters are present then that would be iterated through the number of characters and each character will be appended to existing value in the list or will be just added to list.
    - i. If we have "abc" and there exists '1' in the list then "abc" value will be looped and will be appended to '1' and values "1a", "1b", "1c" will be added to the list.
  - f. As the List is mentioned as reference, it will be passed back to main method.
7. The method explained in step 6 will be called for each specific number.
8. Once all the numbers are read, then the list containing all combinations will be looped and printed with ","(comma) between values.

This way we get the combination of words that can be created from a specific 7-digit telephone number.

## **Spiral Printing:**

### Challenge Description:

A program to print a 2D array (n x m) in spiral order (clockwise)

## Challenge

[VIEW ALL CHALLENGES](#)

### SPIRAL PRINTING

#### CHALLENGE DESCRIPTION:

Write a program to print a 2D array (n x m) in spiral order (clockwise)

#### INPUT SAMPLE:

Your program should accept as its first argument a path to a filename. The input file contains several lines. Each line is one test case. Each line contains three items (semicolon delimited). The first is 'n'(rows), the second is 'm'(columns) and the third is a single space separated list of characters/numbers in row major order. E.g.

```
3;3;1 2 3 4 5 6 7 8 9
```

#### OUTPUT SAMPLE:

Print out the matrix in clockwise fashion, one per line, space delimited. E.g.

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

#### Program Input and Output Values:

Input:

The input file contains several lines. Each line is one test case. Each line contains three items (semicolon delimited). The first is 'n'(rows), the second is 'm'(columns) and the third is a single space separated list of characters/numbers in row major order.

Example: 3;3;1 2 3 4 5 6 7 8 9

Output:

Matrix is printed in clockwise fashion.

Example: 1 2 3 6 9 8 7 4 5

#### Procedure to Solve the challenge:

If we have a matrix then the challenge is to print the value in clockwise format. As in given example matrix will be for below format:

```
1    2    3
```

|   |   |   |
|---|---|---|
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Now the first row has to be printed and then the last column and then last row and then values remaining in first column from bottom and values remaining in second row from starting.

So if we represent values that would be printed in matrix form then that will be as below:




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

But according to the challenge the values will be printed in same single line as 1 2 3 6 9 8 7 4 5.

Code written in this challenge is as follows,



1. Read each line from input file containing details of row, column size and numbers in array.
2. Split the data on ';' (semicolon) and get row and column sizes (n x m) matrix.
3. Now get the numbers by splitting data on ' ' (white space).
4. Declare a 2 dimensional array "Matrix" to store the values.
5. Declare few variables such as "Start\_RowIndex = 0, End\_RowIndex = rows, Start\_ColumnIndex = 0, End\_ColumnIndex = columns" for processing the matrix and get the result.
6. Now a check condition if start index is less than end index for rows and columns.
7. If the condition satisfies then the code will go in.
8. Initially the first row will be stored in string variable(FinalResult) and the start row index value will be incremented.
9. Secondly, the last column will be looped excluding the already stored value in previous step and store values in string variable(FinalResult) and End Column index value will be decremented.
10. Next, a condition is checked if start row index is still less than end row index, if yes then store value of the last row from remaining rows into the variable(FinalResult) and decrement the value of End row index.
11. Then finally check a condition if start column index is less than end column index, if yes then loop through the first column of the remaining columns and store values in the variable.
12. Now the while loop will iterate again until condition in step 6 is not satisfied and display the matrix value in clockwise format.




## RESULTS:

 Challenges Scores Jobs & Offers Messages Timeline Connections Groups   

### Telephone Words



[VIEW SCORES](#)[BACK TO SCORE PAGE](#)[Challenge Description](#)

| REV | LANGUAGE | DATE         | STATUS   | SCORE | TIME, MS | MEMORY, BYTES | IN RANKING | UNIQUE | RANKING POINTS |  |
|-----|----------|--------------|----------|-------|----------|---------------|------------|--------|----------------|--|
| 1   | C#       | Dec 01, 2015 | ✓ Solved | 100   | 1007     | 5730304       | yes        | ✓      | 81.303         |   <a href="#">DELETE</a> |

 Challenges Scores Jobs & Offers Messages Timeline Connections Groups   

### Spiral Printing

[VIEW SCORES](#)[BACK TO SCORE PAGE](#)[Challenge Description](#)

| REV | LANGUAGE | DATE         | STATUS   | SCORE | TIME, MS | MEMORY, BYTES | IN RANKING | UNIQUE | RANKING POINTS |  |
|-----|----------|--------------|----------|-------|----------|---------------|------------|--------|----------------|--|
| 1   | C#       | Dec 02, 2015 | ✓ Solved | 100   | 180      | 5021696       | yes        | ✓      | 87.124         |   <a href="#">DELETE</a> |