Lab Assignment 3
Longest 'A' Path

Due Date (a two-week LA)	
Wednesday Labs	**2/26/20 @ 11:59pm**
Thursday Lab	**2/27/20 @ 11:59pm**

Objectives

- Manipulating two-dimensional lists
- Using recursion to solve a problem

Problem Specification

Write a Python application to find the longest 'A' path on a map of capital (uppercase) letters. The map is represented as a matrix (2-dimensional list) of capital letters. Starting from any point, you can go left, right, up and down (**but not diagonally**). A path is defined as the unbroken sequence of letters that only covers the spots with the letter A. The length of the A path is defined as the number of 'A's on the path. Your program needs to find the longest A path on the map. In the following example, the characters in the longest A path are highlighted and the length of the longest path is 7.

Note that you cannot move backwards over a character 'A' that has already been counted. In the example below, start from the 'A' in the 4th row, 4th column, move upwards and end at the 'A' on the second row, second column.

A U A A

B A A A

T K T A

A A X A <--- Start from here and move upwards.

D A S X

Your program should read the data for the map from a text file (multiple .txt input files will be provided), find the longest A path, and output the length of that path. In the input file, the first line contains 2 integers **m** and **n**. These integers represent the **number of columns *m*** and the **number of rows *n*** of the

map. Starting from the second line of the text file, the map is represented as **n** strings with **m** characters each.

Example Input:

```
17  7
VAAAAAPAAEAFRLDAR
AAAYPBAKAWPKAJEAA
AFAAAUTFENWAQALPC
YYXTHALXSCCMIAAGA
AAXAAZRANZAJALQAA
ZAAAXGJIOAAPAALAX
JAAAAIAAAAATAAADA
```

Example output:

Map dimensions:
Number of columns: 17 Number of rows: 7

Length of the longest "A" path: 10

Example input:

```
15  15
PAYAAHARACAAAAA
ASHAAOAEBFYALAS
YAAOVBAAAEJBAKQ
AAEAXAAAQAQEAAT
AEJZLMUAXHIAAYA
AFAEAAAFKAATPKA
AAKAAAUAATAAAPU
AAAAAAAXAKAAAAQ
VBAJHPAANWAAAAT
AAAWAAZOAIAOHOE
AAAJADMAIZANAEA
MAFWUAAKBPAALGB
AHKAITAAXFAGAAA
DIAGWZARRAANAUA
SAAXHAHAFJAYZAA
```

Example output:

Map dimensions:
Number of columns: 15 Number of rows: 15

Length of the longest "A" path: 20

Design Requirements

You MUST use recursion for this assignment. Recall that recursion is when a method calls itself (one or more times), and each call might again call itself, over and over until some base condition is met and the methods return.

Your design must use the following 3 functions:

```
'''
Use open(), readline() and/or readlines() functions to read the following from
the input file:
- length and width of the map;
- the characters to be stored in the map.
Create the map using the data read from the input file.
Note: you may need to use strip and split functions.

The next part is OPTIONAL but might be helpful to use.
Declare and initialize a Boolean list with similar dimensions to the map; this
list can be used to keep track of the A's in the input file that have already
been counted in the path of A's being 'discovered' in the program.

Parameter to function: input file name.
'''
def readDataFromFile(filename)


'''
Iterate through all the positions in the map (2-dimensional list);
at each position, call the recursive method findPathLengthRecursive(), and at
each position, update the maximum number of A's in the path so far.

Function return: The maximum number of A's in the path.
'''
findLongestPathLength()


'''
This method uses recursion to check the cells to the left, right, above and
below the current cell to determine if any of these is an 'A' that hasn't yet
been counted as part of the longest path of A's.
NOTE: Each 'A' in the path should be counted only once.
Function parameters:
      i: The current row.
      j: The current column.
Function return: Return either zero or the current length signifying the num-
ber of connected A's so far.
'''
findPathLengthRecursive(i, j)


'''
This method determines which of the four values passed to it is the maximum.
The four values passed represent the path lengths for the four paths of recur-
sive calls from a specific character in the 2D list.
Function parameters:
      a: The length returned from position [i-1, j].
      b: The length returned from position [i+1, j].
```

```
      c: The length returned from position [i, j-1].
      d: The length returned from position [i, j+1].
Function return: Returns the Maximum of all lengths passed to it.
'''
find_max(a, b, c, d)
```

Hint:

- OPTIONAL: you might want to use a boolean list with dimensions similar to the map, which keeps track of whether or not a particular 'A' has been counted as part of the A's in the longest path.
- The following piece of code shows how the recursive calls can be made while keeping track of the current length of 'A's and finding the maximum length out of all four recursive calls from a specific position in the 2D list.

```
path_length = 1 + find_max(findPathLengthRecursive(i + 1, j),
                findPathLengthRecursive(i - 1, j),
                findPathLengthRecursive(i, j - 1),
                findPathLengthRecursive(i, j + 1))
```

***Note**: Your output should match what has been displayed in the sample output, and instructions on what each function should do should be followed accurately.*

Additional Requirements

A proper *design* (with detailed pseudocode) and proper *testing* are essential.

> *Note: **Correct pseudocode development** will be worth **40%** of the total LA grade.*

You will also need to install **SPHINX** and use it to **generate HTML documentation** for your projects in Py-Charm. This has been reviewed in the Labs. Accurately following the steps shown in class (a copy of which will be made available in Elearning) will result in the creation of a file "**index.html**" for your project; this file can be opened in a browser and will contain your project's documentation.

Coding Standards

You must adhere to all conventions applicable to writing programs. This includes the use of white spaces and indentations for readability, the use of comments to explain the meaning of various methods and attributes, and the conventions for naming classes, variables, method parameters and methods.

Assignment Submission

- Generate a .zip file that contains all your files including:
  - Program Files
  - Any input or output files
  - The document containing your pseudocode

- Submit the .zip file to the appropriate folder on ELearning.

NOTE: The Elearning folder for LA submission will remain open beyond the due date but will indicate how many days late an assignment was submitted where applicable. The dropbox will be inaccessible seven days after the due date by which time no more credit can be received for the assignment.

The penalty for late submissions as stated in the course syllabus will be applied in grading any assignment submitted late.