**INSTRUCTIONS**

In this assignment, you will be writing the board interface for the Game of Life program that will be assigned as programing project #03. The Game of Life is a simulation routine that takes place on a two-dimensional rectangular grid. Each of the squares on the grid represents a cell that can contain either life or death. Before the game can begin, the board must be loaded with a series of states for each cell that indicates whether or not life exists in that particular location. We will be writing a number of utilities in this assignment that will make the development of the game logic itself a smoother process. The following functions will be written and tested by the provided driver and data file:

Signature: Create2Darray(firstDimensionLength, secondDimensionLength)

Description: This program will take two numeric arguments. It will construct a Python two-dimensional list with dimensions described by the arguments. After the 2D list has been created, it will return a reference to that list for other portions of the program can access it. By default, all of the elements in the list should contain a value of 0.

Signature: InitializeBoard(board)

Description: This function will take a two-dimensioanl list as its only argument. The function should create a two-dimensional loop that goes through the range of coordinates along the length and width of the board and populate each of the cells with the symbol designated for death (MARK\_DEAD). The function should behave in a robust manner and use the len() function to determine how far along each of the coordinates the loops must go when populating the cells. This function does not return any values.

Signature: PopulateBoardFromFile(board, fileName)

Description: This function will take the two-dimensional board and the name of a file as its arguments. The file should be opened and all coordinates that are read from the file should be analyzed and the appropriate spot on the board should be marked with life (MARK\_LIVE). If there is an error in the format of the data file, a message should be displayed that indicates what was found in the file that your routine could not parse as a valid coordinate. Each coordinate should also be checked to determine if it actually exists on the board. In the event that a coordinate is taken from that file that cannot be translated to a specific spot on the board, a message should be displayed indicating which coordinate was bad and no action should be taken. In the event that either error is encountered, the offending entry in the file should be disregarded and the next entry should be processed. Please refer to the provided data file and make sure you

Signature: DisplayBoard(board, generationLabel)

Description: This function will take the two-dimensional game board and a generation label as its arguments. The function should process the entries in the two-dimensional list and display the game board in the fashion demonstrated by the provided output. Above the game board, there should be a numeric label for the generation that this board represents. When the Game of Life actually runs, an initial configuration for board life is given some cells live and some cells die based on a series of rules that will be described with the project description. Each time a cycle of life and death occurs, it is referrered to as a single ‘ Generation’. These generation labels will increment when the final project is written and the simulation runs. For now, just provide a space at the top of the board where these labels can be displayed.

After you have written your functions, test them with the following driver. Take note of the global constants that appear at the top of the code. These will be used to determine the size of the two-dimensional array that makes up the game board and the values of the single characters that will be used to represent life or death for a given cell.

SAMPLE DRIVER

# IMPORTS

import sys

# GLOBAL CONSTANTS

INPUT\_FILE = "A08S-GOLBoardInterfaceData.dat"

NUM\_ROWS = 10

NUM\_COLS = 10

MARK\_LIVE = "L"

MARK\_DEAD = "D"

# FUNCTIONS

def main():

 # create a 2d array that represents the board

 gameBoard = Create2DArray(NUM\_ROWS, NUM\_COLS)

 # populate the board with dead squares to start

 InitializeBoard(gameBoard)

 # populate the board with live coordinates from the data file

 PopulateBoardFromFile(gameBoard, INPUT\_FILE)

 # display the board to see if everything worked

 DisplayBoard(gameBoard, 0)

SAMPLE OUTPUT

EXCEPTION: The token '0' or '0a' could not be parsed from the data file.

EXCEPTION: The coordinate (10, 10) cannot be accessed on the board.

+------------------------------+

| GENERATION # 0 |

+------------------------------+

 D | D | D | D | D | D | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | L | D | D | D | D | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | L | D | D | D | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | L | D | D | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | L | D | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | D | L | D | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | D | D | L | D | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | D | D | D | L | D | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | D | D | D | D | L | D

---+---+---+---+---+---+---+---+---+---

 D | D | D | D | D | D | D | D | D | L