CSCI 3110 Extra Project 1

A maximum of 50 points may be added to the total grade for the projects in the class.

This program requires you to solve the maze problem using recursion plus backtracking.

Test your program against the 3 datafiles MyMaze1.dat, MyMaze2.dat, and MyMaze3.dat.

The data file is formatted as the following:

```
20 7      《--size of the maze  number of columns  number of rows
0 18      《-- the coordinates of the entrance point
6 12      《--the coordinates of the exit point
******************* *
*       *        ***** *
* ***** ***           *
* ***** *****     ** *
* *                 *  *
* *******   *         *
************ *******
```

Your program needs to :

(1) Use dynamic memory allocation for the 2D array Maze
(2) Define two classes: **CreatureClass** and **MazeClass** (for MazeClass, you have to define your own destructor to free memory space dynamically allocated for the Maze).
(3) Implement recursive GoNorth(), GoSouth(), GoEast(), GoWest() client program functions as described in book.
(4) If a path exists between entrance and exit, print out the entire maze, including
    a. the path from the entrance point to the exit point marked by characters 'p',
    b. spaces explored by the creature by character ‘v’, like the following:

```
******************p*
*      *pppppp*****p*
* *****p***vppppppp*
* *****p*****    ** *
* *vvvvppppppp    *  *
* *******  *p       *
************p*******
```

If no path exists for the problem, display the entire maze after the exploration process. P

Here is a recommended Steps in developing this program:

Part A:

For part A, implement the **CreatureClass.** A creature can more to the north, or the south, or the east, or the west one step at a time. It can be assigned to a location, and can report its current location. The only data it needs to keep up with is its location, or coordinates at all times. First, write the header file and the implementation file of CreatureClass. Then, write a simple client program to test this class. (The client program may create a creature, assign it to a

particular location, move it x steps to the east, y steps to the south, z steps to the north, q steps to the west, and then report its current location.)

Part B:

Implement the MazeClass. A Maze object consists of a maze (2D array of character), the total number of rows and columns of the maze, and the entrance and exit locations of the maze. In terms of the methods of the class:

- a maze may be read from a file,
- the maze may be displayed
- it may return the locations of the entrance and exit points
- mark that one of the maze location is visited by a created, or is part of a path being explored by a creature,
- it is able to determine whether a particular location in the maze is wall, is clear of objects, is the exit point, is in fact part of the maze

You are required to used dynamic allocation of memory to create the maze array. As a result, you need to explicitly define the destructor for the MazeClass to release memory space allocated when the maze object exits its scope.

Write the header file and implementation file for the MazeClass, and then write a simple client program to test your MazeClass. Create and initialize a maze object, report the entrance and exit locations, and display for 3 different locations of the maze whether it is wall or is clear.

Final Program

Implement the final client program that solve the maze. Implement recursive GoNorth(), GoSouth(), GoEast(), GoWest() client program functions as described in the text book and discussed in class. This program will include both the mazeClass and the creatureClass header files.