Lab 04 - Queues & Breadth First Search

Instructions:

- **The lab requires writing a complete cpp file and cpp file within an hour. It requires completing 2 tasks.**
- **Accompanying this file is a template cpp file that you must modify. You cannot include additional libraries to or remove any libraries from the template file. All other modifications are allowed.**
- **Your submission must be submitted to the Labs directory of your github repository and/or as an attachment on Google classroom under the Lab04 assessment. The file must remain a cpp file.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating and/or failing to follow any of the rules will result in an automatic zero (0) for the lab.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, AT THE BEGINNING OF YOUR SUBMISSION(S), ADD A COMMENT THAT CONSISTS OF YOUR NAME AND THE DATE

Along with the **Queue.h** file, you are provided the **Path.h** file that contains the class *Path* and two functions `To2D()` and `To1D()`.

The class *Path* represents an 8×8 char grid with as corresponding 8×8 bool grid. Its methods are

- `GetValue(r,c)` - if r and c, it returns the element of the char grid which can be modified; otherwise, it throws an error message.
- `GetState(r,c)` - if r and c, it returns the element of the bool grid which can be modified; otherwise, it throws an error message.
- `Toggle()` - it switches between returning a string of the char grid or the bool grid for the `ToString()` method. If it returns true, a string of the char grid is returned; otherwise, a string of the bool grid is returned.
- `GeneratePath()` - it populates a random char grid of pluses (`+`) and spaces ( ), and it makes the bool grid all false.
- `ToString()` - it returns a string of either the char grid or the bool grid.
- `operator<<` - it displays is in the same format as the return of `ToString()`.

Furthermore, the function `To2D()` that accepts an int parameter and two int reference parameters respectively, assigns the reference parameters to corresponding row and column indices respectively of an 8×8 grid based on the int parameter and returns true if the int parameter represents a valid index of one-dimensional array that corresponds to a 8×8 grid; otherwise, it returns false.

And the function `To1D()` that accepts two int parameters and an int reference parameter respectively, assigns the reference parameter the corresponding index of an one-dimensional array that corresponds to a 8×8 grid based on the int parameters and returns true if the int parameters represent valid indices of a 8×8 grid; otherwise, it returns false.

Your objective is to write a complete program that uses a breadth first search to determine if a path exists from a given start point to a given end point. A breadth first search allows you search through a graph or tree by using a queue. Its algorithm is as follows:

A1. if the current node has not been visited, then

  B1. if the current node is the target, then

    a. terminate algorithm as a success.

  B2. else,

    a. add adjacent nodes of the current node to the queue.

    b. mark current node as visited.

A2. if the queue is empty, then

  B1. terminate algorithm as a failure.

A3. else,

  B1. remove the front from the queue and make it the current node.

  B2. goto to step A1.

To accomplish your objective, complete the following tasks

- ☐ define a void function named `GetAdjacencies()` that takes a constant *Path* reference, an int *Queue* reference and an int parameter respectively. If the int parameter is a valid index of an one-dimensional array corresponding to an 8×8 grid, the function will inserts all the corresponding one-dimensional array indices of the elements of the *Path* object that are one horizontal or vertical space away from the element whose corresponding one dimensional array index is equal to the int parameter. Otherwise, the function does nothing.

- ☐ define a bool function named `HasPath()` that takes a constant *Path* reference parameter and two int parameters. If both int parameters represent one-dimensional array indices that correspond to an 8×8 grid, the function determines if a path exists in the *Path* object starting from the first int parameter and ending at the second int parameter. Otherwise, the function returns false. A path exists if there exists an ordered collection of elements from the *Path* object such that the char values of all the elements in the collection are the space character with the possible exceptions of the first and last elements, the one-dimensional index of the first element of the collection is equal to the value of the first int parameter, one-dimensional index of the last element of the collection is equal to the value of the second int parameter, and each sequential element in the collection is either one horizontal or vertical element away from each other.