**Link List Searching Application**

**Author**: John Eric Simmons

**Date Written**: 12/17/2017

**Application stored in my personal GitHub repository at**:

<https://github.com/billionayer/CardlyticsEvalCode/tree/master/LinkedListTester>

**Development Environment**: Visual Studio 2015 (Community Edition)

Development Language: C#

Application Structure:

1 Projects

* **LinkListTester** – This project houses the link list objects and the user interface to test the linked list.
  + **MyLinkListItem Class** – encapsulates the necessary logic for a single item in the list list. The class is written as a .NET generic whereby any data type can be stored in the Value property of the MyLinkListItem Class.
  + **MyLinkList Class –** encapsulte the necessary logic to store and manage a collection of MyLinkListItem objects as a single linked list.
    - Provides methods to push new items into the link list.
    - Provides methods to clear the link list
    - Provides methods to find a specific item in the list based on item location number and can be searched either in ascending or descending order.
    - The key to searching the link list in descending order is to setup an array of reference pointers to linked list objects. The size of the array is the number of entries equal to the value the user for the item location from the back of the list. The code will iterate through the entire list (since we do not know how many entries are in the list) and at each iteration a reference to the link list item will be stored in a corresponding location in the array. If the current position in the link list exceed the size of the array then we start assigning back at the beginning of the array (a circular array).
    - Once we iterate through the entire link list, we simply take the count of items obtained during the iteration, and do a modulo operation on the count versus the item number. This operation should provide the index location in the array of the desired item from the link list.
    - Care was taken to consider edge cases such as empty link list or specifiying a zero location.
  + **MainForm** – The main user interface window to test the application.

Care was taken to consider situations such as invalid data in the file, passing of null reference objects, and file handling expections should an I/O exception occurred. The application also attempt to handle situations where the files does not exists.

