Assignment One

Anthony Rodriguez

Anthony.rodriguez2@Marist.edu

September 17, 2020

1 Linked List

[Line 3] Creation of List class;
[Line 4 - 6] Creation of Node (Pointers), item refers to the embedded information within the container. The Node is the actual pointer that directs to the next.
[Line 9 - 11] Creation of a Node method with parameters of a String referring to an item. The "THIS" keyword refers to the properties of this method.
[Line 17-18] Designating the Node head and tail and setting them to a null value for future use.
[Line 20 - 28] Creation of addPoint method: Creates a new Node to be added to the Linked List under the condition the head is null. In this scenario both receive new points or Nodes, otherwise only the tail receives a new Node.
[Line 32 - 46] A display method is created in order to check to see if the head is null. This will ensure if the list is currently empty. A while loop is used to check if the list is occupied and will generate a print statement with the current items.
[Line 49 +] Unused in the final build, maintained to debug and check code.

2 SelfStack

[Line 5 - 8] Creation of the class and initialization of members.
[Line 10] Self Stack method declared.
[Line 14] Adds parameters to method with a generic type and value variable. Using a String cast to value in reference to THIS method.
[Line 24 - 26] Creation of a push method; requires the parameters of a List with the variable name newList from the main class. Assigns THIS with a new iteration of the SelfStack class with parameters of the previous and current values. NewList from main file is being used as a parameter and converted to a string to make the compiler happy.
[Line 31 - 39] Creation of the pop() method. Checks to see if the stack is empty and will throw an exception with an error saying so. Assigns top with the THIS value and moves elements. Returns the top element to be used for comparison.
[Line 43 - 44] Creation of the peek() method. Used to check the top value of the stack but does not remove it.

[Line 48 - 49] isEmpty() method to check to see if the stack is empty.
[Line 53 -54] Size method created to check the size of the stack, implements the isEmpty method.
[Line 58 - 66] Find a certain element. Refers to itself to check if the stack is empty and if not to refer to the element.
[Line 70 +] Not used for the final build. Used to debug and test the methods.

3 Queue

[Line 6 - 11] Creation of the class and initialization of its members to be used for the methods. The members include variables to store the string data from main class with identifiers of the front and back elements of the queue. Also added a means to see the size of the queue and a counter to help with enqueue and dequeue.
[Line 13 - 18] Creation of the queue method requiring an int for a parameter to dictate size. Adds values to the members of the class.
[Line 21 - 24] Enqueue method with parameters of an array that refers to the list in main class. Adds a element to the rear of the queue. Adds the elements of the file to the rear and increases a counter.
[Line 28 - 31] Dequeue method to remove from the queue. Adds a new front element, decreases the counter and returns the front for comparison.
[Line 38 +] Not used in the final build. Created to test methods without interference of other classes.

4 Read/Main

[Line 1 -11] Imports of various utilities including path and file.
[Line 14] Main method
[Line 15] Initialize a variable to hold the string that is to be received from a scanner.
[Line 18] Scanner used in conjunction with a File object. First find the file and then scan the contents.
[Line 21] Creation of a list with a new iteration of an array.
[Line 24 - 26] While loop to scan the contents of the file and added it to the generated list from above.
[Line 31 - 34] Creation of new objects from the Stack class and Queue class. Running the generated list into the stack and queue in preparation for comparison.
[Line 36 +] Work in progress. Ultimately, this is where the comparison and occurs along with a print statement to produce the requested palindromes.