My algorithms time complexity is quadratic and roughly 0(n^2) at worst case.

The first thing we do is read through the .txt file and pull out the integer values to be placed in our linked list method using the only for loop in the program utilizing the insert\_node method. I then created the recursive method that checks for matches in the linked list by checking the next node the list points to. If the node isn’t point to a null value, it continues to check down the list. After the recursive method returns a null value and all the list has already been checked the method returns false to exit the recursion method.

The algorithm compares each and every node in the list minus one. So, the first time through the list if we pass in 7 integers it only makes 6 comparisons. The next time through the recursive method the place holder node moves up the list and make yet another comparison between this node and the list. So, the more times it goes through the list the less comparisons it must make. The first time through the list its n-1 the second time through its n-2 and so on. In this algorithm we check the adjacent node to make sure that it’s not a null value before we call the recursion method again that way, we don’t have any unnecessary processes.

Pseudo-code

Method list\_Checker(currentNode, nextNode){ // two node objects to compare as arguments

IF the value of the nodes match THEN return TRUE // print yes

IF the value of node currentNode is NULL return FALSE// print no

IF the node after nextNode is NULL THEN set the nextNode to the next node in the list and return a recursive method call list\_Checker passing back in the new values // recursive call

ELSE set the currentNode to the next node in the list and set the nextNode to the

currentNode and return list\_checker passing back the newly set values. // recursive call

}