COMP202 FALL 2020 HOMEWORK 2

I have completed this assignment individually, without support from anyone else. I hereby accept that only the below listed sources are approved to be used during this assignment:

(i) Course textbook,

(ii) All material that is made available to me by the professor (e.g., via Blackboard for this course, course website, email from professor / TA),

(iii) Notes taken by me during lectures.

I have not used, accessed or taken any unpermitted information from any other source. Hence, all effort belongs to me.

I signed as Sinan Cem Erdoğan (68912).

```
ListNode reverseList(head)
      while(current != null)
         next = current.next
         current.setNext(previous)
         previous = current

      return previous;
```

We shift our 3 pointers n times and do 1 comparison.

Time Complexity for this function = T(n) = 3n+1
So it is O(n).

We have some local variables(pointers).

Space Complexity for this function = O(1)

```
ListNode middleElement (ListNode head) {
      slower = head;
      faster = head;

    if(head != null)

      while(faster != null && faster.next != null)
        faster = faster.next.next
        slower = slower.next


     return slower
```

We shift our 2 pointers n-(n%2) times and do one comparison.

Time Complexity for this function =
T(n) = 2(n-(n%2)) + 1
So it is O(n).

We have 2 pointer variable locally.

Space Complexity for this function = O(1)

```
boolean isPalindromeRestricted(head)
      resersed = reverseList(middleElement(head))
      while (revesed.next != null)
         if(head.data != reversed.data)
               return false
         head = head.next
         reversed = reversed.next
      return true
```

Time Complexity

Time Compleixty for isPalindromeRestricted
= T(n) for reverseList() + T(n) for middleElement() + 2 (two comparison).
= (3n+1) + (2n+1) + 2

We have T(n) for isPalindromeRestricted T(n) = 5n+4

Time Complexity for this algorithm is O(5n+4) = O(n).

Space Complexity

For this algorithm we use nothing but a few pointers. When we call the function the space for them allocated and when the function returns they are deallocated.

Space Complexity for this algorithm is O(1).

```
ListNode listToNewList(head)
    tempHead = head
    head2 = null
    current = null
    while(head != null)
       if (head2 == null)
              head2 = new ListNode(head.data)
              current = head2
              head = head.next
       else
          current.setNext(new ListNode(head.data))
          current = current.next
          head = head.next

    return head2
```

We copy the values from list to alist one by one. We go until the end of the list and assign them to the list.

Time Complexity for this function = T(n) = 2n.
It is O(n).

We create an list of n integer. n is the number of integer in the old list.

Space Complexity for this function = O(n).

```
boolean isPalindromeUnrestricted(ListNode head)
    reversedHead = reverseList(listToNewList(head))


    if(head !=null && reversedHead != null)
      if(head.data != reversedHead.data)
        return false

    while(head.next !=null && reversedHead.next != null)
      reversedHead = reversedHead.next
      head = head.next
      if(head.data() != reversedHead.data)
        return false

    return true
```

Time Complexity

Time Compleixty for isPalindromeRestricted
= T(n) listToNewList() + T(n) for reverseList() + 4n+4= 2n + 3n+1 + 4n + 4 = 9n+5

We have T(n) for isPalindromeRestricted T(n) = 9n+5

Time Complexity for this algorithm is O(9n+5) = O(n).

Space Complexity

For this algorithm we create an list of size n to compare the values. Allocated space for this algorithm basically n + c(constant).

Space Complexity for this algorithm is O(n).