Homework 8

Problem 9.2 Linked Lists & Rooted Trees

a) **Pseudocode:**

```
REVERSE(ListNode head)
    // check if list is empty
    if(head == NULL)
    return NULL
     // take three pointers named current, prev and next
     // initialize them
    ListNode current = head
    ListNode prev = NULL
    ListNode next = NULL
    // iterate through the list until current is not equal to NULL
    while (current != NULL)
      next = current->next // go to the next node; otherwise in the next step the connection
// between the nodes is lost
      current->next = prev // now it is pointing to the previous node of the list
      prev = current // Move one step forward
      current = next // Move one step forward
    head = prev
end
```

- Time Complexity = O(n), where n is the length of the Linked List; we only have a single loop.
- Space Complexity = O(1)

The above algorithm is an in-situ algorithm as it transforms input using no auxiliary data structure. We are not creating any new copies of the list, we are only creating 3 new pointers,

but they all are pointing to our current list, and therefore the algorithm does not exceed a constant no matter how large the input.