ADS HW 8

Problem 8.2

a) Pseudocode:

```
// reverses a linked list in Θ(n) time
        reverseList(mylist)
                // create pointers of type struct linkedlist
                struct linkedlist *current, *prev, *next;
                make current = mylist
                prev = NULL;
                while (current != NULL)
                        next = current->next;
                        current->next = prev;
                        prev = current;
                        current = next;

                end while

                mylist = prev;

                return mylist;

          end function
```

The time complexity is Θ(n) or linear, since we only have a single while loop repeating "n" number of times (for n being the size of the linked list).

An algorithm is said to be an in-situ algorithm if the extra amount of memory required to execute the algorithm is a constant which does not depend on the size of the input. In the above algorithm, three struct linkedlist pointers are used as extra storage or memory while executing the algorithm, but since they all point to a single node, no matter the size of the linked list, the extra memory used by the algorithm will be constant. Hence it is an in-situ algorithm.

b) Algorithm to convert BST to sorted linked list:

For the best case scenario, the BST has equal number of left and right nodes and we have $T(n) = 2T(n/2) + O(n)$, whose time complexity is $O(n*\log n)$. This is because of the two recursive calls during the conversion to a linked list which are both in $T(n/2)$. To clarify where $O(n)$ comes from, $O(n)$ is the time complexity of the pushback function.

Moving on to the worst-case scenario, one of the recursive calls would be $T(n-1)$ and the other wouldn't do anything at all considering the case when the BST has only the left or right side. In this case, the time complexity would be: $T(n) = T(n-1) + O(n)$, which is $O(n^2)$.