Reverse Linked List using recursion

1 Problem description

Problem:	Given a singly linked list A. Reverse A
Input:	Linked list A
Output:	Link list A after reversed
Example:	Input: A = 1 → 2 → 3 → 4 → None
	Output: A = 4 → 3 → 2 → 1 → None

2 Pseudo code

Algorithm Using recursive method to reverse a singly linked list

```
procedure REVERSELIST($head \leftarrow pointer\ point\ to\ head\ of\ linked\ list$)
    if $head$ is $None$ then
        return $head$
    end if
    Split list into 2 part:
            1. $first \leftarrow head$ (Part 1 is the first element of list)
            2. $rest \leftarrow head.next$ (Part 2 is the remaining part → rest is a pointer points to the second element)
    if $rest$ is $None$ then
        return $head$
    end if
    $rest = reverseList(rest)$ (recursive: reverse the 2nd part)
    Merge part 1 at the end of part 2:
            $first.next.next = first$
            $first.next = None$
    Head pointer points to 1st element of new list (1st element of the rest part)
            $head = rest$
    return $head$
end procedure
```