Memory required to traverse a linked list in reverse tail to head

* traversing from head to tail we only need 1 node pointer and call node->next to get the next node in the list.
* traversing from tail to head
* we need a length index eg. int length
* node pointer to traverse to the tail -> head to tail.
* Increment length every time we call node->next
* If node->next = null we return that
* Point our nodepointer to the head -> if head isn’t null then we traverse to length-1 node.
* Function is recursive -> uses a lot of memory

How could the structure of a linked list be changed to make this less memory intensive?

* Usage of doubly linked list.
* Node has a pointer to next & previous node.
* Allows traversal head to tail & tail to head without recursion in functions.