# COMP 1603 – Tutorial/Lab 6

1. Write a function `reverse` which returns a pointer to the reversed list. The function accepts a pointer to the top of a linked list. Note that it is not allowed to create any new nodes.

2. Write a function `numPairs` which accepts two linked lists of integers and an integer sum as parameters which counts all the pairs of nodes, one from the first and the other from the second list that add up to the sum given and return this result.

3. Write a function `unionL` which accepts two linked lists of integers with each individual linked list having a distinct set of keys and returns a pointer to a new linked list (new nodes are created) containing the union of both linked lists **without** duplicates in the new list.
   For example, if we have top1->3->1->5->NULL and top2->5->4->2->NULL, then the result could be top->3->1->5->4->2->NULL (note that 5 appears once).

4. Write a function `intersection` which accepts two linked lists of integers with each individual linked list having a distinct set of keys and returns a pointer to a new linked list (new nodes are created) containing the intersection which is all the elements common to both linked lists without duplicates in the new list.
   For example, if we have top1->3->1->5->NULL and top2->5->4->2->NULL, then the result could be top->5->NULL.