Linked Lists

---

1. Write a function `removeNode()` that accepts a reference to the head of a linked list (`ptrHead`) and an integer `index`. The function should remove the node at the specified index and return `1` if the deletion is successful, and `0` otherwise. You should ensure the correct handling of the head pointer when deleting the first node.

   The function definition is as follows:

   ```
   def removeNode(ptrHead, index):
   ```

2. Re-write the function `removeNode()` using the `LinkedList` structure defined in the lecture materials. The function should remove a node at the specified index from the linked list.

   The function definition is as follows:

   ```
   def removeNode2(ll, index):
   ```

3. Write a function `split()` that copies the contents of a linked list into two other linked lists. The function prototype is given below:

   ```
   def split(head,
             ptrEvenList,
             ptrOddList):
   ```

   The function should copy nodes with even indices (0, 2, 4, etc.) to `evenList` and nodes with odd indices (1, 3, 5, etc.) to `oddList`. The original linked list should remain unmodified.

   Sample output:

   ```
   Current list: 1 3 5 2 4 6 19 16 7
   Even list: 1 5 4 19 7
   Odd list: 3 2 6 16
   ```

4. Write a function `duplicateReverse()` that creates a duplicate of a linked list with the nodes stored in reverse. The function prototype is given below:

   ```
   def duplicateReverse(head, ptrNewHead):
   ```

   The function should return 0 if the operation was successful and -1 otherwise. newHeadPtr should point to the first node of the reversed duplicate list.

   Sample output:

   ```
   Current list: 1 3 5 2 4 6
   Reversed list: 6 4 2 5 3 1
   ```