Write Algorithm/Pseudocode for following operations on a Singly Linked Lists

1. **Inserting a new node in a linked list at the beginning.**

Step 1) Start

Step 2) Create a new node

Step 3) Add the data in new node.

Step 4)Add the pointer of the linked list at the end of the new node.

Step 5)End

Time Complexity: O(n)

Space Complexity’s(1)

1. **Inserting a new node in a Linked List at the end**.

Step 1) Start

Step 2)Create a new node

Step 3) Add the data in new node.

Step 4)Add the pointer of the new node at the end of the linked list null pointer .

Step 5)End

Time Complexity: O(n)

Space Complexity:O(1)

**Inserting a new node in a Linked List after a given node.**

Step1)

Step 2)Create a new node

Step 3) Add the data in new node.

Step 4)Add the pointer of the new node at the end of the end of the node

Step 5)End

Time Complexity: O(n)

Space Complexity:O(1)

**Inserting a new node in a Linked List before a given node**

Step1)Start

Step 2)Create a new node

Step 3) Add the data in new node.

Step 4)Add the pointer of the new node at the end of the end of the node

Step 5)End.

Time Complexity: O(n)

Space Complexity:O(1)

**Deleting a node from a Linked List from beginning** .

Step1)Start

Step 2)Delete the first node from program

Step 3) Remove the pointer of the first nodes from the second node

Step 4)Add the pointer of the new node at the end of the end of the node

Step 5)End.

Time Complexity: O(n)

Space Complexity:O(1)

**Deleting a node from a Linked List from beginning**

Step1)Start

Step 2)Add the data in new node

Step 3) Add the pointer of the node which should be after it at the end of new node.

Step 4)Replace the pointer of the before node from the pointer of the new node

Step 5)End.

Time Complexity: O(n)

Space Complexity:O(1)

**Deleting a node from a Linked List from end**

Step1)Start

Step 2)Delete the end node from program

Step 3) Remove the pointer of the end node from the last second node.

Step 4)Replace the pointer of the before node from the pointer of the new node

Step 5)End.

Time Complexity: O(n)

Space Complexity:O(1)

**Deleting a node after a given node**

Step1)Start

Step 2)Delete the first node from program

Step 3) Replace the pointer of the before nodes from the next node

Step 4)End.

Time Complexity: O(n)

Space Complexity:O(1)

**Display the entire linked list (Traversal)**

Step 1)Start

Step2) Initialise a while loop with condition till the null pointer

Step3)iterate the linked list in the loop and print the elements .

Step 4)End

Time Complexity: O(n)

Space Complexity:O(1)

**Search for an element in linked list.**

Step1)Start

Step2) Initiate a while loop with condition till the null pointer

Step 3)Iterate at each element in loop and give if the element matches with the search data

Step4)Print the element if the data matches

Step5)End

Time Complexity: O(n)

Space Complexity:O(1)

**Count the number of nodes in a Singly Linked List**

Step1)Start

Step2) ) Initiate a while loop with condition till the null pointer and create a variable

Step 3)Iterate at each loop and give if condition if the item is pointer increment variable by 1

Step4)Print the variable as number of node in linked list

Step5)End

Time Complexity: O(n)

Space Complexity:O(1)

**Reverse the singly linked list**

in singly linked list reversing is not possible