**PSG College of Technology**

**Department of AMCS**

**II Sem M.Sc. CS / TCS / SS / DS Data Structures Lab**

**Linked Lists**

**Implementation of Linked Lists**

1. Write a menu driven program to implement a singly list with the following options,

a. Insert in the beginning / at end / in the middle

b. delete in the front / at last / any element (given position, given item)

c. search for an element / display the linked list

1. Write a menu driven program to implement a circular list with the following options,

a. Insert in the beginning / at end / in the middle

b. delete in the front / at last / any element (given position, given item)

c. search for an element / display the linked list

1. Write a menu driven program to implement a doubly list with the following options,

a. Insert in the beginning / at end/ in the middle

b. delete in the front / at last / any element (given position, given item)

c. search for an element / display the linked list

**Singly Linked Lists**

1. Divide a linked list such that nodes with odd values from one sublist and nodes with even values form another sublist.
2. Divide a linked list such that nodes 1,3,5, … from one sublist and nodes 2,4,6,… form another sublist.
3. Given two linked lists A and B, concatenate list B after list A.
4. Write a removeDuplicates() function that takes a list and deletes any duplicate nodes from the list. The list is not sorted.

**Examples:**

**Input: i**f the linked list is 12->11->12->21->41->43->21 then

**Output:** 12->11->21->41->43.

1. Given a linked list and two keys in it, swap nodes for two given keys. Nodes should be swapped by changing links. Swapping data of nodes may be expensive in many situations when data contains many fields. It may be assumed that all keys in the linked list are distinct.

**Examples:**

**Input :** 10->15->12->13->20->14, x = 12, y = 20

**Output:** 10->15->20->13->12->14

**Input :** 10->15->12->13->20->14, x = 10, y = 20

**Output:** 20->15->12->13->10->14

1. Write a function that moves the last element to the front in a given Singly Linked List. For example, if the given Linked List is 1->2->3->4->5, then the function should change the list to 5->1->2->3->4.
2. Given two lists sorted in increasing order, create and return a new list representing the intersection of the two lists. The new list should be made with its own memory — the original lists should not be changed.

**Example:**

**Input:**

First linked list: 1->2->3->4->6

Second linked list be 2->4->6->8,

**Output:** 2->4->6.

1. Perform the following sorting on Singly Linked List.
   1. Insertion Sort,
   2. Selection Sort,
   3. Merge Sort and
   4. Quick Sort
2. Given pointer to the head node of a linked list, the task is to reverse the linked list. We need to reverse the list by changing the links between nodes.

**Examples**:

**Input**: Head of following linked list   
1->2->3->4->NULL   
**Output**: Linked list should be changed to,   
4->3->2->1->NULL

**Circular Linked List**

1. There are n people standing in a circle waiting to be executed. The counting out begins

at some point in the circle and proceeds around the circle in a fixed direction. In each

step, a certain number of people are skipped and the next person is executed. The

elimination proceeds around the circle (which is becoming smaller and smaller as the

executed people are removed), until only the last person remains, who is given

freedom. Given the total number of persons n and a number m which indicates that m-1

persons are skipped and m-th person is killed in circle. The task is to choose the place in

the initial circle so that you are the last one remaining and so survive.

Examples :

Input : Length of circle : n = 4, Count to choose next : m = 2

Output : 1

Input : n = 5; m = 3

Output : 4

1. Exchange the first and last node of a circular linked list.

Example:

Input list : 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> …

Output list : 6 -> 2 -> 3 -> 4 -> 5 -> 1 -> …

**Doubly Linked List**

1. Given a doubly linked list with even number of nodes, reverse every alternate node with its neighbour. For example,

Input list : 1 -> 2 -> 3 -> 4 -> 5 -> 6.

Output list : 2 -> 1 -> 4 -> 3 -> 6 -> 5.

1. Reverse a doubly linked list

Input list : 1 <-> 2 <-> 3 <-> 4 <-> 5 <-> 6.

Output list : 6 <-> 5 <-> 4 <-> 3 <-> 2 <-> 1.

**Implementation of other ADTs.**

1. Implement A STACK with necessary operations of PUSH, POP, OVERFLOW,

UNDERFLOW and DISPLAY using (i) Singly linked list (ii) Doubly linked list

1. Given Nodes with their priority, implement a priority queue using (i) Singly linked list (ii) Doubly linked list.
2. Implement a circular queue using a circular linked list.
3. Implement a double ended queue (DEQUE) using doubly linked list.