**DS Lab 4 Exercise**

**Task#1:**

You are given the array (3 1 2, 5, 8) of size 5 print that array on screen. Now add the 9, 11, 4 at the last **(do not create new array also do not change the size of array manually defined at the time of declaration)** and print the updated array. Now delete the 1, 2, and 5 then print the updated array.

**Note all the deletion and insertion must be performed in original array do not create the new array and change the size manually.**

**Task#2**:

Write a program that asks the user to enter the number to move the number of elements at last and then it moves exact number of elements to last element of given link list.

**Example:** given list: 5, 3 1 8 6 4 2

**Enter the number:** 2

**After rotation:** 1 8 6 4 2 5 3

**Task#3**

**Delete a** value from any position (not the first one only) from the **singly linked list** if it’s present in the list otherwise show a message to the user that the value is not present in the list.

**Task#4**

Solve the following problem using a Singly Linked List.

Given a Linked List of integers, write a function to modify the linked list such that all even numbers appear before all the odd numbers in the modified linked list. Also, keep the order of even and odd numbers same.

**Examples:**

**Input:** 17->15->8->12->10->5->4->1->7->6->NULL

**Output:** 8->12->10->4->6->17->15->5->1->7->NULL

Input: 8->12->10->5->4->1->6->NULL

Output: 8->12->10->4->6->5->1->NULL

// If all numbers are even then do not change the list

**Input:** 8->12->10->NULL

**Output:** 8->12->10->NULL

// If all numbers are odd then do not change the list

**Input:** 1->3->5->7->NULL

**Output:** 1->3->5->7->NULL

**Task#5**

Solve the following problem using a Singly Linked List.

Given a Linked List of integers or string, write a function to check if the entirety of the linked list is a palindrome or not

Examples:

**Input:** 1->0->2->0->1->NULL

**Output:** Linked List is a Palindrome

**Input:** B->O->R->R->O->W->O->R->R->O->B->NULL

**Output:** Linked List is a Palindrome

**Task#6:**

Create a circular link list and perform the mentioned tasks.

1. Insert a new node at the end of the list.
2. Insert a new node at the beginning of list.
3. Insert a new node at given position.
4. Delete any node.
5. Print the complete circular link list.

**Task#7:**

Give an efficient algorithm for concatenating two doubly linked lists **L** and **M**, with head and tail preserved nodes, into a single list that contains all the nodes of **L** followed by all the nodes of **M**.

**Task#8:**

Given a linked list, you have to perform the following task:

1. Extract the alternative nodes starting from second node.
2. Reverse the extracted list.
3. Append the extracted list at the end of the original list.

**Note**: Try to solve the problem without using any extra memory.

**Example 1:**

**Input:**

LinkedList = 10->4->9->1->3->5->9->4

**Output:**10 9 3 9 4 5 1 4

**Explanation:**Alternative nodes in the given linked list are 4,1,5,4. Reversing the alternative nodes from the given list, and then appending them to the end of the list results in a list 10->9->3->9->4->5->1->4.