Name: - Atish Kumar

Roll No: - 120CS0173

Lab Sheet:- 04

Q1. Write a program to implement the Doubly linked list. Perform the following operations on the doubly linked list:

- Creating an empty doubly linked list
- Adding the new element at the beginning of the linked list.
- Deletion of a node after a particular location
- . Counting the no of nodes.
- Displaying the linked list.

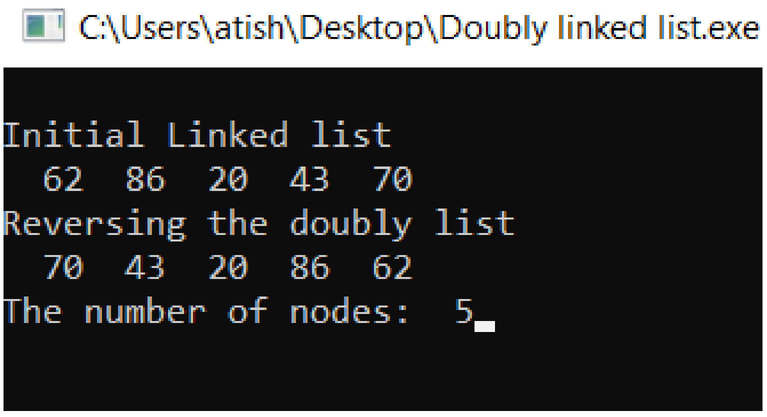
Program:-

```
#include <stdio.h>
#include <stdlib.h>
struct DoublyList{
   int obj;
   struct DoublyList* nextNode;
   struct DoublyList* prevNode;
};
void insertAtBeg(struct DoublyList** head, int info)
{
   struct DoublyList* newNode = (struct DoublyList*)malloc(sizeof(struct
DoublyList));

   newNode->obj = info;

   newNode->nextNode = (*head);
   newNode->prevNode = NULL;
   if ((*head) != NULL)
      (*head)->prevNode = newNode;

   (*head) = newNode;

}
void EmptyLinkedList(struct DoublyList* node){
        node->nextNode = NULL;
        node->prevNode = NULL;
}



int Size_Nodes(struct DoublyList* node){
        int x = 0;
        while(node!=NULL){
                x++;
                node = node->nextNode;
        }
        return x;
}



void deleteNodeAtLocation(struct DoublyList** head, struct DoublyList*
node)
{

   if (*head == NULL || node == NULL)
```

```
        return;



    if (*head == node)
        *head = node->nextNode;



    if (node->nextNode != NULL)
        node->nextNode->prevNode = node->prevNode;



    if (node->prevNode != NULL)
        node->prevNode->nextNode = node->nextNode;



    free(node);
    return;
}
void printList(struct DoublyList* head)
{
    struct DoublyList* lastNode;
    printf("\nInitial Linked list \n");
    while (head != NULL) {
        printf("  ");
        printf("%d",head->obj);
        lastNode = head;
        head = head->nextNode;

    }

    printf("\nReversing the doubly list \n");
    while (lastNode != NULL) {
        printf("  ");
        printf("%d",lastNode->obj);

        lastNode = lastNode->prevNode;

    }
}
int main()
{

    struct DoublyList* N = NULL;
    insertAtBeg(&N, 70);
    insertAtBeg(&N, 43);
    insertAtBeg(&N, 20);
    insertAtBeg(&N, 86);
    insertAtBeg(&N, 62);
    printList(N);
  printf("\nThe number of nodes:  ");
  printf("%d",Size_Nodes(N));
```

```
    getchar();

    return 0;

}
```

Output:-



C:\Users\atish\Desktop\Doubly linked list.exe

```

Initial Linked list
  62  86  20  43  70
Reversing the doubly list
  70  43  20  86  62
The number of nodes:  5_
```

Q2. Write a program to remove the duplicate elements from a sorted linked list?

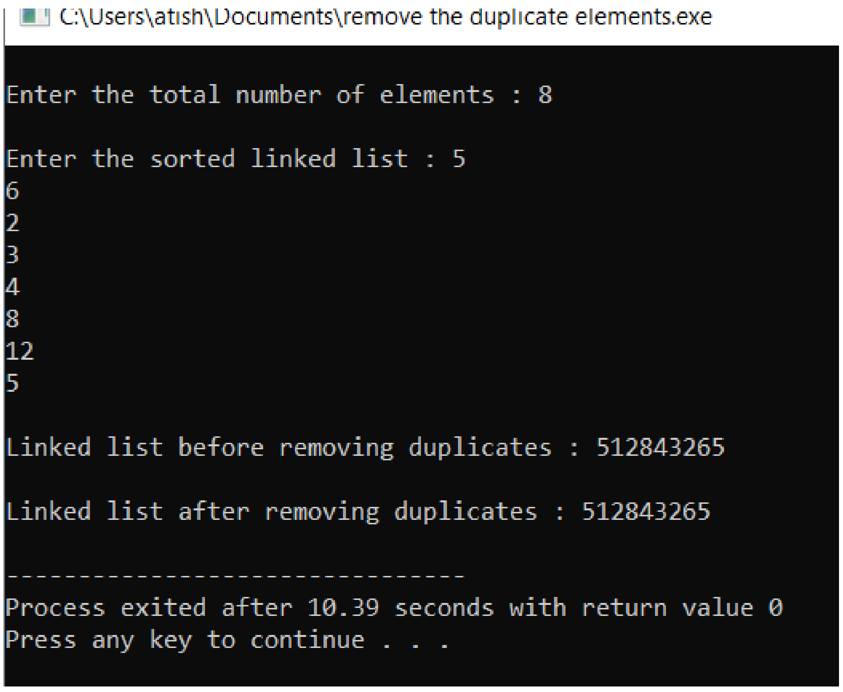
Program:-

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
int data;
struct node* next;
};

void insert_elements(struct node** head, int new_data)
{
struct node* new_node = (struct node*) malloc(sizeof(struct node));
new_node -> data = new_data;
new_node -> next = (*head);
(*head) = new_node;
}
void display_list(struct node *node)
{
while (node!=NULL)
{
printf("%d", node->data);
node = node -> next;
}
}
void remove_duplicate_elements(struct node* head)
{
struct node* current = head;
struct node* next_next;

if (current == NULL)
```

```
return;

while (current -> next != NULL)
{

if (current -> data == current -> next -> data)
{
next_next = current -> next -> next;
free(current -> next);
current -> next = next_next;
}
else
{
current = current -> next;
}
}
}
int main()
{
struct node* head = NULL;
int n;
printf("\nEnter the total number of elements : ");
scanf("%d", &n);
printf("\nEnter the sorted linked list : ");
int i;
for(i = 0; i < n; i++)
{
int data;
scanf("%d", &data);
insert_elements(&head, data);
}
```

```
printf("\nLinked list before removing duplicates : ");

display_list(head);

printf("\n");


remove_duplicate_elements(head);


printf("\nLinked list after removing duplicates : ");

display_list(head);

printf("\n");

return 0;

}
```

Output:-



```
C:\Users\atish\Documents\remove the duplicate elements.exe

Enter the total number of elements : 8

Enter the sorted linked list : 5
6
2
3
4
8
12
5

Linked list before removing duplicates : 512843265

Linked list after removing duplicates : 512843265

--------------------------------
Process exited after 10.39 seconds with return value 0
Press any key to continue . . .
```

Q3. Write a program to print all the elements of the single linked list in reverse order. The algorithm should have linear time complexity and constant space complexity.

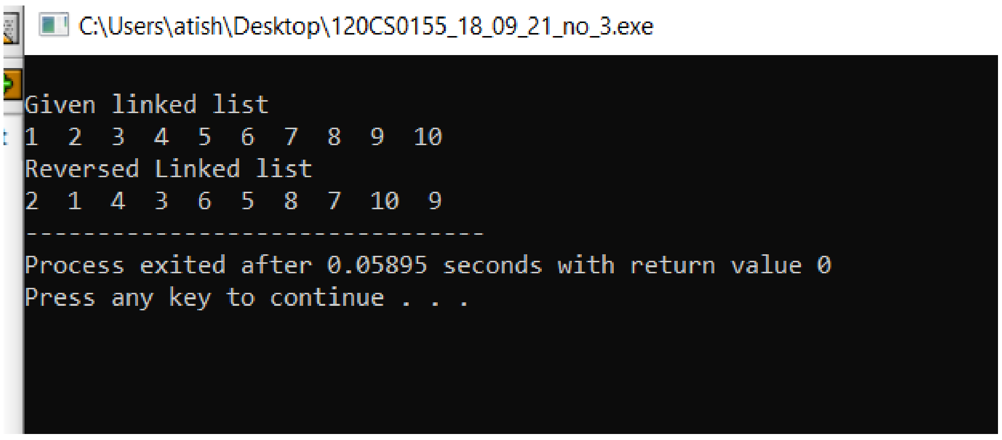
Program:-

```
#include<stdio.h>

#include<stdlib.h>

struct Node

{

 int data;

 struct Node* next;

};

struct Node *reverse (struct Node *head, int k)

{

  if (!head)

   return NULL;

    struct Node* current = head;

    struct Node* next = NULL;

    struct Node* prev = NULL;

    int count = 0;

    while (current != NULL && count < k)

    {

        next  = current->next;

        current->next = prev;

        prev = current;

        current = next;

        count++;

    }

    if (next !=  NULL)

   head->next = reverse(next, k);

    return prev;

}
```

```
void push(struct Node** head_ref, int new_data)
{

   struct Node* new_node =

         (struct Node*) malloc(sizeof(struct Node));
   new_node->data  = new_data;
   new_node->next = (*head_ref);
   (*head_ref)    = new_node;
}
void printList(struct Node *node)
{
   while (node != NULL)
   {
      printf("%d  ", node->data);
      node = node->next;
   }
}
int main(void)
{
   struct Node* head = NULL;
   push(&head, 10);
    push(&head, 9);
    push(&head, 8);
    push(&head, 7);
    push(&head, 6);
    push(&head, 5)
    push(&head, 4);
    push(&head, 3);
```

```
  push(&head, 2);


   push(&head, 1);

   printf("\nGiven linked list \n");

  printList(head);

  head = reverse(head, 2);

   printf("\nReversed Linked list \n");

   printList(head);

   return(0);

}
```

Output:-


C:\Users\atish\Desktop\120CS0155_18_09_21_no_3.exe

```

Given linked list
1  2  3  4  5  6  7  8  9  10
Reversed Linked list
2  1  4  3  6  5  8  7  10  9
--------------------------------
Process exited after 0.05895 seconds with return value 0
Press any key to continue . . .
```