Ex.no:03 Implementation of Singly,Doubly & Circular Linked Lists

Date: 16.07.24

Aim:

Program:

1.) You are given the head of a linked list.Remove every node which has a node with a greater value anywhere on the right.Return the head of the modified Linked List.

2.)C++ Program to remove duplicates from a sorted Linked List.Write a function that takes a list in a non decreasing order and deletes any duplicate node from the list.The list should be traversed once.

3.) C++ program to reverse every element in the doubly linked list and also reverse and print the reversed elements in the doubly linked list.

Algorithm:

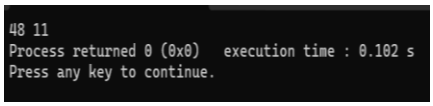
Code:

1.)

```
#include<iostream>
using namespace std;
class Node{
   public:
   int data;
   Node *next=NULL;};
Node *rearrange(Node *head){
   Node *prev=NULL;
   Node *curr=head,*Next=NULL;
   while(curr!=NULL){
      Next=curr->next;
curr->next=prev;
  prev=curr;
Curr=Next;
   }
   head=prev;
   Node *result=new Node();
   result->next=prev;
   Node *current=head;
   while(current!=NULL){
      current=current->next;
      if(current!=NULL){
      if(prev->data < current->data){
         prev->next=current;
         prev=prev->next;}
      else{
         prev->next=current->next;}}}
   return result->next;}
int main(){
```

```
    Node *head=new Node();
    Node *second=new Node();
    Node *third=new Node();
    head->data=48;
    head->next=second;
    second->next=third;
    second->data=7;
    third->data=11;
    Node *result=rearrange(head);
    Node *prev=NULL;
    Node *curr=result,*Next=NULL;
    while(curr!=NULL){
        Next=curr->next;
curr->next=prev;
prev=curr;
curr=Next;}
    result=prev;
while(result!=NULL){
        cout<<result->data<<" ";
        result=result->next;}}
```

Output:



2.)

```
#include<iostream>
using namespace std;
class Node{
   public:
   int data=0;
   Node *next=NULL;};
Node *removedup(Node *head){
  Node *prev = head;
Node *curr = head;
while (curr != NULL && curr->next != NULL) {
   if (prev->data != curr->next->data) {
      prev = prev->next;
      curr = curr->next;
   } else {
      curr->next = curr->next->next;}}
   return head;}
int main(){
   Node *head=new Node();
   Node *second=new Node();
   Node *third=new Node();
head->data=1;
   head->next=second;
Second->next=third;
   second->data=3;
 third->data=3;
Node *result=removedup(head);
 while(result!=NULL){
      cout<<result->data<<" ";
```

```
    result=result->next;

  }}
```

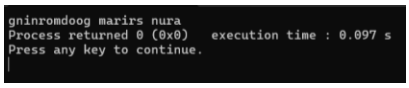
Output:

3.)

```
#include<algorithm>
#include<string>
#include<iostream>
using namespace std;
class Node{
   public:
   string data;
   Node *next=NULL;
};

Node *rearrange(Node *head){
   Node *prev=NULL;
   Node *curr=head,*Next=NULL;
   while(curr!=NULL){
      Next=curr->next;

      curr->next=prev;

      prev=curr;

      curr=Next;
   }
   head=prev;

   while(prev!=NULL){
      string s1=prev->data;
      reverse(s1.begin(),s1.end());
      prev->data=s1;
      prev=prev->next;
   }
return head;
}

int main(){
   Node *head=new Node();
   Node *second=new Node();
```

```
    Node *third=new Node();

    head->data="arun";
    head->next=second;

    second->next=third;
    second->data="sriram";

    third->data="goodmorning";

    Node *result=rearrange(head);

    while(result!=NULL){
        cout<<result->data<<" ";
        result=result->next;
    }


}
```

Output:



Result:

The above programs are executed successfully.