```cpp
#include<bits/stdc++.h>
using namespace std;
class Node{
public:
   int info;
   Node *prev;
   Node *next;
   Node(int val){
      info = val;
      prev = NULL;
      next = NULL;
   }
};
void create(Node **head , int item){
   Node* newPtr = new Node(item);
   if(*head == NULL){
      *head = newPtr;
   }
   else{
      Node* temp = *head;
      while(temp->next != NULL)temp = temp->next;
      temp->next = newPtr;
      newPtr->prev = temp;
   }
}
void display(Node **head){
   cout<<"Displayed data ";
   Node *temp = *head;
   while(temp!=NULL){
      cout<<temp->info<<" ";
      temp = temp->next;
   }
   cout<<endl<<endl;
}
int isPalindrome(Node **head){
   Node *last = *head;
   Node *first = *head;
   while(last->next != NULL)last = last ->next;
   while(first != NULL){
      if((last->info) != (first->info))return 0;
      first = first->next;
      last = last->prev;
   }
   return 1;
}

void firstInsert(Node **head , int item){
   Node *newPtr = new Node(item);
   newPtr->next = *head;
   if(*head != NULL)(*head)->prev = newPtr;
   *head = newPtr;
}
void lastInsert(Node **head , int item){
   if(*head == NULL){
      firstInsert(head , item);
      return;
   }
   Node *newPtr = new Node(item);
   Node *temp = *head;
   while(temp -> next != NULL)temp = temp->next;
   temp->next = newPtr;
   newPtr->prev = temp;
}
void dataInsert(Node **head , int data , int item){
   Node *newPtr = new Node(item);
   Node *temp = *head;
   while(temp!= NULL && temp -> info != data)temp = temp->next;
   if(temp == NULL){
      cout<<"can't be inserted"<<endl;
      return;
   }

   newPtr->next = temp->next;
   newPtr->prev = temp;
   if(temp->next != NULL){
      temp->next->prev = newPtr;
   }
   temp->next = newPtr;
}

void firstDelete(Node **head){
   Node *nextNode = (*head)->next;
   *head = nextNode;
   (*head)->prev = NULL;
}
void dataDelete(Node **head , int data){
   Node *temp = *head;
   while(temp!= NULL && temp->info!=data)temp = temp->next;
   if(temp == NULL){
      cout<<"cant be deleted"<<endl;
      return;
   }
   Node *prevNode = temp->prev;
   Node *nextNode = temp->next;
   prevNode->next = nextNode;
   if(nextNode != NULL)nextNode->prev = prevNode;
}
int main(){
   int n;cin>>n;
   Node *head = NULL;

   for(int i = 0 ; i < n ; i++){
      int data;
      cin>>data;
      create(&head , data);
   }


   while(1){
   cout<<"Select operation "<<endl;
   cout<<"FirstInsert 1"<<endl;
   cout<<"LastInsert 2"<<endl;
   cout<<"Data Insert 3"<<endl;
   cout<<"FirstDelete 4"<<endl;
   cout<<"Data delete 5"<<endl;
   cout<<"is palindrome 6"<<endl;
      int op;cin>>op;
      if(op == 1){
            int item;cin>>item;
         firstInsert(&head , item);
      }
      if(op == 2){
         int item;cin>>item;
         lastInsert(&head , item);
      }
      if(op == 3){
         int remData , addData;
         cout<<"remData addData ";
         cin>>remData>>addData;
         dataInsert(&head , remData , addData);
      }
      if(op == 4){
            firstDelete(&head);
        }
        if(op == 5){
            cout<<"Remdata ";
            int remData;cin>>remData;
            dataDelete(&head , remData);
        }

        if(op == 6){
            if(isPalindrome(&head))cout<<"Palindrome"<<endl;
            else cout<<"NOt palindrome"<<endl;
        }
        else display(&head);
    }

    firstInsert(&head , 10);

    lastInsert(&head , 10);

    display(&head);
    cout<<isPalindrome(&head);

}
```