```cpp
#include<iostream>
using namespace std;

class Linked_list
{
private:
    struct jnode
    {
        int data;
        struct jnode *next;
    };
    typedef struct jnode node;
    node *head=NULL;
    node *tail=NULL;
    int cnt=0;

public:

    void insert_begi(int val)
    {
        node *new_node=new node;
        cnt++;
        new_node->data=val;
        if(head==NULL)
        {
            tail=new_node;
        }
        new_node->next=head;
        head=new_node;
    }
    void insert_pos(int pos, int val)
    {
        int i;
        node *tmp=head;
        if(pos>cnt)
        {

            return;
        }
        if(pos==cnt)
        {
            append(val);
            return;
        }
        if(pos==0)
        {
            insert_begi(val);
            return;
        }
        else
        {
            node *new_node=new node;
            cnt++;
            new_node->data=val;
            for(i=0; i<pos-1; i++)
                tmp=tmp->next;
            new_node->next=tmp->next;
            tmp->next=new_node;
        }


    }
    void append(int val)
    {
        node *new_node=new node;
        cnt++;
        new_node->data=val;
        new_node->next=NULL;
        if(head==NULL)
        {
            head=new_node;
        }
        else
        {
            tail->next=new_node;
```

```
    }
    tail=new_node;
}
void display()
{
    node *tmp=head;
    while(tmp!=NULL)
    {
        cout<< tmp->data << " ";
        tmp=tmp->next;
    }
    cout<<endl;
}
void sort()
{
    bool s=true;
    node *lptr=NULL;
    node *tmp;
    if(head==NULL)
        return;
    while(s)
    {
        s=false;
        tmp=head;
        while(tmp->next!=lptr)
        {
            if( (tmp->data)>(tmp->next->data))
            {
                int c=tmp->data;
                tmp->data=tmp->next->data;
                tmp->next->data=c;
                s=true;
            }
            tmp=tmp->next;
        }
        lptr=tmp;
    }

}
void del_pos(int pos)
{
    int i;
    if(pos>=cnt)
    {
        cout<<"invalid position"<<endl;
        return;
    }
    node *tmp=head;
    for(i=0; i<pos-1; i++)
        tmp=tmp->next;
    node *free=tmp->next;
    tmp->next=tmp->next->next;
    delete free;
    cnt--;
}
void del_begi()
{
    node *tmp=head;
    head=head->next;
    free(tmp);
    cnt--;
}
void del_end()
{
    node *tmp=head;
    while(tmp->next!=tail)
    {
        tmp=tmp->next;
    }
    tmp->next=NULL;
    node *cur=tail;
    tail=tmp;
    free(cur);
    cnt--;
```

```
    }
    void read_pos(int pos)
    {
        node *tmp=head;
        int i;
        for(i=0; i<pos; i++)
            tmp=tmp->next;
        cout<< "Data in position "<<pos<< " is "<<tmp->data<<endl;
    }
    void uptate_pos(int pos, int val)
    {
        node *tmp=head;
        int i;
        for(i=0; i<pos; i++)
            tmp=tmp->next;
        tmp->data=val;
        cout<< "Value uptated succesfulll !....."<<endl;
    }
    void length()
    {
        cout<< "Length of the list = "<<cnt<<endl;
    }


};
int main()
{
    Linked_list list1;
    list1.append(6);
    list1.append(32);
    list1.append(8);
    list1.append(9);
    list1.display();
    list1.insert_begi(32);
    list1.insert_pos(3,40);
    list1.display();
    list1.append(19);
    list1.append(67);
    list1.display();
    list1.sort();
    list1.display();
    list1.del_begi();
    list1.del_end();
    list1.del_pos(3);
    list1.display();
    list1.length();




    return 0;
}
```