```
#include "Node.cpp"
#include<iostream>
using namespace std;

template<class T>
class LinkedList
{
        private:
                Node<T> *head;
                Node<T> *tail;

        public:
                LinkedList():head(nullptr),tail(nullptr){}

                Node<T>* getHead(){

                return head;
                        }

                bool isFull()
                {
                        return false;
                }

                bool isEmpty()
                {
                        return(nullptr==head && nullptr==tail);
                }

                bool AddAtEnd(T ele)
                {
                        Node<T> *temp=new Node<T>;
                        bool bSuccess=false;

                        temp->setData(ele);
                        temp->setNext(nullptr);


                        if(isEmpty()){

                                head=tail=temp;
                                bSuccess=true;
                        }
                        else{
                                tail->setNext(temp);
                                tail=temp;


                                bSuccess=true;
                        }

                        return bSuccess;

                }
//Add At Beginning
                bool AddAtBeg(T ele)
                {
                        Node<T> *t= new Node<T>;
                        t->setData(ele);
                        if(!isEmpty())
                        {
                                t->setNext(head);
                                head=t;
                        }
                        else
                        {
                                t->setNext(nullptr);
                                head=tail=t;
                        }
                }

                void DelAtEnd()
                {
                        T ele;
                        if(!isEmpty())
                        {

                                Node<T> *t=head;

                                if(head->getNext()==nullptr && tail->getNext()==nullptr){
                                        head=tail=nullptr;
                                        delete t;
                                }
                                else{
                                        while(t->getNext()!=tail)
                                                {
                                                t=t->getNext();
                                                }
                                         ele=tail->getData();
                                         t->setNext(nullptr);
                                         delete tail;
                                         tail=t;
                                         }
                                         cout<<"DeletedAtEnd"<<endl;

                        }
                        else
                        {
                                cout<<"\nLinkedList is Empty\n"<<endl;
                        }
```

```cpp
                }

                void DelAtBeg(){
                        T ele;
                        if(!isEmpty())
                        {

                                Node<T> *t=head;

                                if(head->getNext()==nullptr && tail->getNext()==nullptr){
                                        head=tail=nullptr;
                                        delete t;
                                }
                                else{
                                        head=head->getNext();
                                        delete t;
                                        }

                                cout<<"DeletedAtBeg"<<endl;
                        }
                        else
                        {
                                cout<<"\nLinkedList is Empty\n"<<endl;
                        }

                }
                int Size()
                {
                        int n=0;
                        Node<T> *t=head;
                        while(t!=nullptr){
                                t=t->getNext();
                                n++;
                        }
                        return n;
                }

                void InsertAtN(int n, T ele){
                int s=Size();
                if(n<=s && n>=0){
                        if(n<=1)
                                AddAtBeg(ele);
                        else{
                                int count=1;
                                Node<T> *temp=new Node<T>;
                                temp->setData(ele);
                                Node<T> *p=head;
                                while(count<n-1){
                                        p=p->getNext();
                                        count++;
                                }
                                temp->setNext(p->getNext());
                                p->setNext(temp);
                        }
                        Display();
                }
                else
                        cout<<"Enter Value less than "<<s<<endl;


}

                void DelAtN(int n){
                        int s=Size();
                        T ele;
                if(n<=s && n>=1){
                        if(n==1)
                                DelAtBeg();
                        else{
                                int count=1;
                                Node<T> *p=head;
                                Node<T> *q=head->getNext();
                                while(count<n-1){
                                        p=p->getNext();
                                        count++;
                                }
                                q=p->getNext();
                                ele=q->getData();
                                p->setNext(q->getNext());
                                delete q;
                }
                        Display();
                }
                else
                        cout<<"Enter Value less than "<<s<<endl;


}

                void Reverse()
                {
                if(!isEmpty()){
                        if(!(head->getNext()==nullptr && tail->getNext()==nullptr)){

                        Node<T> *t1=head;
                        Node<T> *t2=nullptr;

                        head=head->getNext();
                        t1->setNext(nullptr);
                        tail=t1;
                        // tail=head;
                        //cout<<head->getData();
                        while(head!=nullptr)
                        {
                                t2=head;
                                head=head->getNext();
                                t2->setNext(t1);
                                t1=t2;
                        }
                        head=t2;
                 }
                 else return;
        }

                }

        /*      void Display(Node<T> *t)
                {
                        if(!isEmpty()){
                        //Node<T> *t=head;
                        while(nullptr!=t){
                                cout<<t->getData()<<"-->";
                                t=t->getNext();
                        }
                        cout<<"nullptr\n"<<endl;
                        }
                        else
                                cout<<"LinkedList isEmpty"<<endl;
                }
        */
                void Display()
                {
                        if(!isEmpty()){

                        Node<T> *t=head;

                        while(nullptr!=t){
                                cout<<"cLL->";

                                Node<int>* ct=t->getData();//Getting Child LL's Head Address

                                //Traversing and Printing Child
                                while(ct!=nullptr){
                                cout<<ct->getData()<<"->";
                                ct=ct->getNext();
                                }

                                t=t->getNext();
                                cout<<"nullptr"<<endl;
                        }
                        cout<<"End main LL"<<endl;
                        }
                        else
                                cout<<"LinkedList isEmpty"<<endl;
                }
};
int main(){

        LinkedList<Node<int>*> mLL;//Main Linked List

        LinkedList<int> c1LL;//Children LL's
        LinkedList<int> c2LL;
        LinkedList<int> c3LL;

        c1LL.AddAtEnd(11);//Inserting Data's in child LL's
        c1LL.AddAtEnd(22);
        c1LL.AddAtEnd(33);

        c2LL.AddAtEnd(44);
        c2LL.AddAtEnd(55);
        c2LL.AddAtEnd(66);
        c2LL.AddAtEnd(77);
        c2LL.AddAtEnd(88);

        c3LL.AddAtEnd(99);

        mLL.AddAtEnd(c1LL.getHead());//Adding Child LL to Main LL
        mLL.AddAtEnd(c2LL.getHead());
        mLL.AddAtEnd(c3LL.getHead());

        cout<<"\n////Main Linked List//////\n"<<endl;
        mLL.Display();  //Printing Main LL

        mLL.Reverse();//Reversing MainLL

        mLL.Display();

        return 0;
}
```