#1

```
#include <iostream>
using namespace std;
struct node {
    int data;
    node* next;
};
class SLinkedList {
    private:
        node* head;
        node* tail;
    public:
        SLinkedList() {
            head = nullptr; // or NULL
            tail = nullptr; // or NULL
            cout << "head and tail nodes are initiated with nullpoint" << endl;
        }
};
int main() {
    SLinkedList numList1;
    return 0;
}
```

```
head and tail nodes are initiated with nullpoint
```

#2

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

struct node {
    int data;
    node* next;
};
class SLinkedList {
    private:
        node* head;
        node* tail;
    public:
        SLinkedList() {
            head = nullptr;
            tail = nullptr;
            cout << "head and tail nodes are initiated with nullpoint" << endl;
```

```
    }


void ListAppend(int elem) {
   node *newNode = new node;
   newNode->data = elem;
   newNode->next = nullptr;
   if (head == nullptr ) {
      head = newNode;
      tail = newNode;
   }
   else {
      tail->next = newNode;
      tail = newNode;
   }
}

void ListDisplay() {
   node *tmp;
   tmp = head;
   while (tmp != nullptr) {
      cout << tmp->data << " ";
      tmp = tmp->next;
   }
   cout << endl;
}
};

int main() {
   SLinkedList numList1;
   numList1.ListAppend(30);
   numList1.ListAppend(40);
   numList1.ListDisplay();
   return 0;
}
```

```
head and tail nodes are initiated with nullpoint
30 40
```

#3

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

```
struct node {
    int data;
    node* next;
};
class SLinkedList {
    private:
        node* head;
        node* tail;
    public:
        SLinkedList() {
            head = nullptr;
            tail = nullptr;
            cout << "head and tail nodes are initiated with nullpoint" << endl;
        }


void ListAppend(int elem) {
    node *newNode = new node;
    newNode->data = elem;
    newNode->next = nullptr;
    if (head == nullptr ) {
        head = newNode;
        tail = newNode;
    }
    else {
        tail->next = newNode;
        tail = newNode;
    }
}
void ListPrepend(int elem) {
    node* newNode = new node;
    newNode->data = elem;
    newNode->next = nullptr;
    if (head == nullptr) {
        head = newNode;
        tail = newNode;
    }
    else {
        newNode->next = head;
        head = newNode;
    }
}
```

```
void ListDisplay() {
    node *tmp;
    tmp = head;
    while (tmp != nullptr) {
        cout << tmp->data << " ";
        tmp = tmp->next;
    }
    cout << endl;
}
};

int main() {
    SLinkedList numList1;
    numList1.ListAppend(30);
    numList1.ListAppend(40);
    numList1.ListPrepend(20);
    numList1.ListPrepend(10);
    numList1.ListDisplay();
    return 0;
}
```