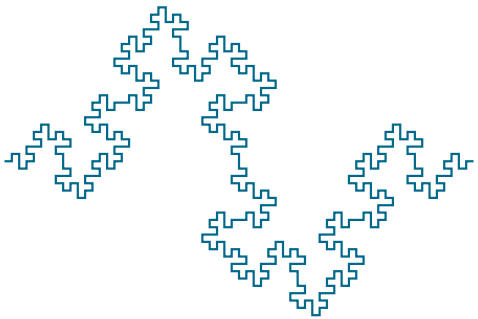
Fundamental Data Structures

Part1: Lists

Phung Hua Nguyen
HCMC University of Technology



April 6, 2020

Outline

IS IT A LIST?

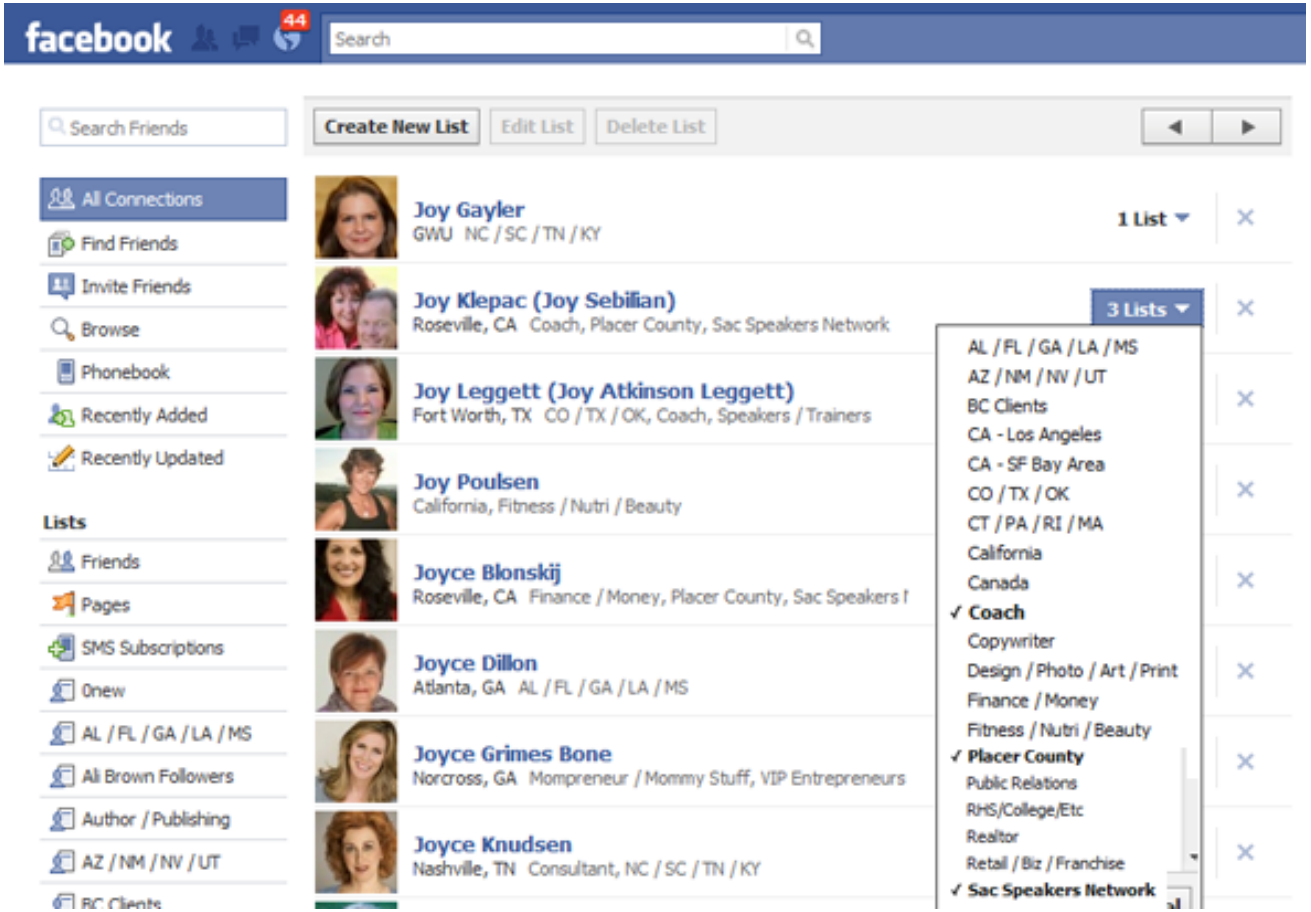
Check In List

Tuesday, May 17, 2011 from 1:00 PM - 4:00 PM (CT)

	Last Name	First Name	Qty	Ticket Type	Payment Status
☐	Amaral	Megan	1	Sample Ticket	Paid with Check Order 1519072587-33401865
☐	Arnold	Megan	1	Sample Ticket	Paid with Check Order 1519072587-33401803
☐	Atefi	Yassmin	1	Sample Ticket	Paid with Check Order 1519072587-33401943
☐	Bach	Sarah	1	Sample Ticket	Paid with cash Order 1519072587-33414775
☐	Chandler	Laura	1	Sample Ticket	Paid with cash Order 1519072587-33414593
☐	Harrigan	Miles	1	Sample Ticket	Paid with cash Order 1519072587-33401741
☐	Hogan	Kevin	1	Sample Ticket	Paid with Check Order 1519072587-33414711
☐	Kramer	Seth	1	Sample Ticket	Paid Online With PayPal Order 1519072587-33401563
☐	Turner	Nicole	1	Sample Ticket	Paid Online With PayPal Order 1519072587-33414485

☐

IS IT A LIST?


facebook
44
Search
Search Friends
Create New List
Edit List
Delete List
All Connections
Find Friends
Invite Friends
Browse
Phonebook
Recently Added
Recently Updated
Lists
Friends
Pages
SMS Subscriptions
0new
AL / FL / GA / LA / MS
Ali Brown Followers
Author / Publishing
AZ / NM / NV / UT
BC Clients
Joy Gayler
GWU NC / SC / TN / KY
1 List
Joy Klepac (Joy Sebilian)
Roseville, CA Coach, Placer County, Sac Speakers Network
3 Lists
Joy Leggett (Joy Atkinson Leggett)
Fort Worth, TX CO / TX / OK, Coach, Speakers / Trainers
Joy Poulsen
California, Fitness / Nutri / Beauty
Joyce Blonskij
Roseville, CA Finance / Money, Placer County, Sac Speakers
Joyce Dillon
Atlanta, GA AL / FL / GA / LA / MS
Joyce Grimes Bone
Norcross, GA Mompreneur / Mommy Stuff, VIP Entrepreneurs
Joyce Knudsen
Nashville, TN Consultant, NC / SC / TN / KY
AL / FL / GA / LA / MS
AZ / NM / NV / UT
BC Clients
CA - Los Angeles
CA - SF Bay Area
CO / TX / OK
CT / PA / RI / MA
California
Canada
✓ Coach
Copywriter
Design / Photo / Art / Print
Finance / Money
Fitness / Nutri / Beauty
✓ Placer County
Public Relations
RHS/College/Etc
Realtor
Retail / Biz / Franchise
✓ Sac Speakers Network

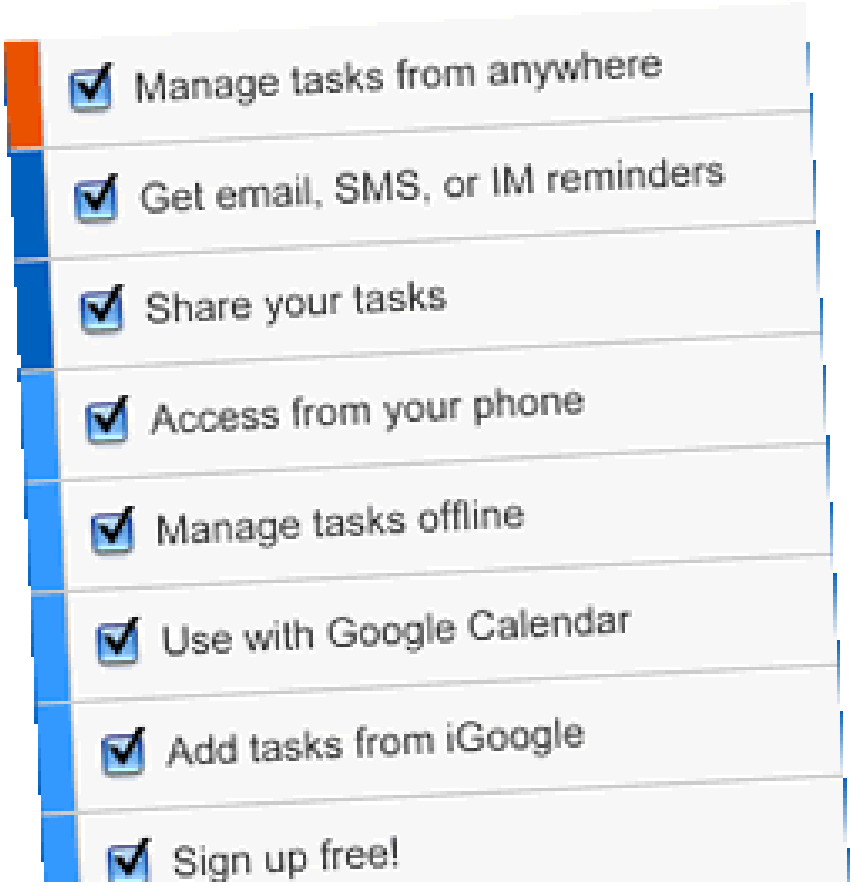
IS IT A LIST?

□

Is It A List?

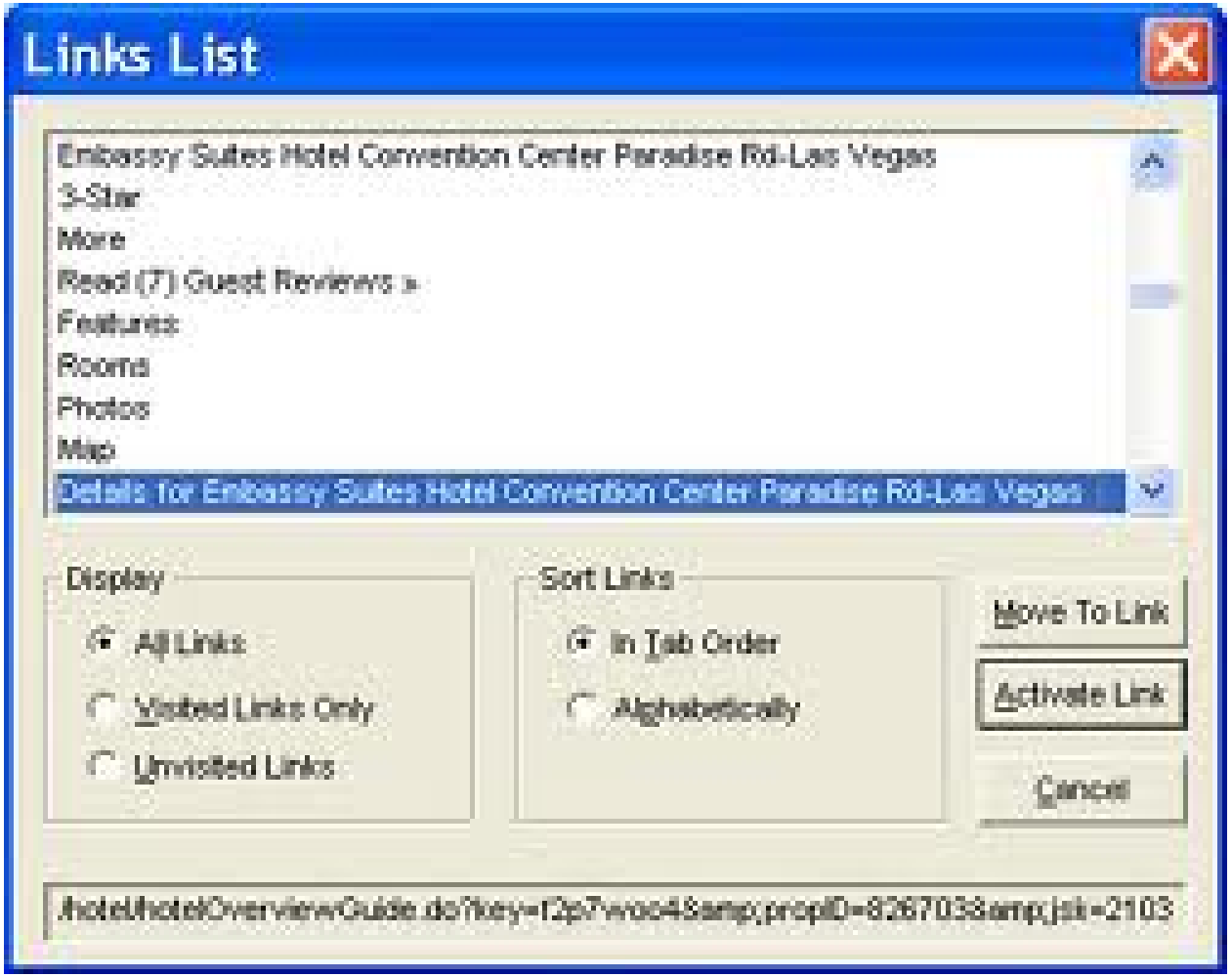
□

IS IT A LIST?


Manage tasks from anywhere
Get email, SMS, or IM reminders
Share your tasks
Access from your phone
Manage tasks offline
Use with Google Calendar
Add tasks from iGoogle
Sign up free!

□

IS IT A LIST?


Links List
Embassy Suites Hotel Convention Center Paradise Rd-Las Vegas
3-Star
More
Read (7) Guest Reviews »
Features
Rooms
Photos
Map
Details for Embassy Suites Hotel Convention Center Paradise Rd-Las Vegas
Display
All Links
Visited Links Only
Unvisited Links
Sort Links
In Tab Order
Alphabetically
Move To Link
Activate Link
Cancel
/hotel/hotelOverviewGuide.do?key=f2p7vxoc4&propID=8267038&jsk=2103

□

IS IT A LIST?

□

IS IT A LIST?

□

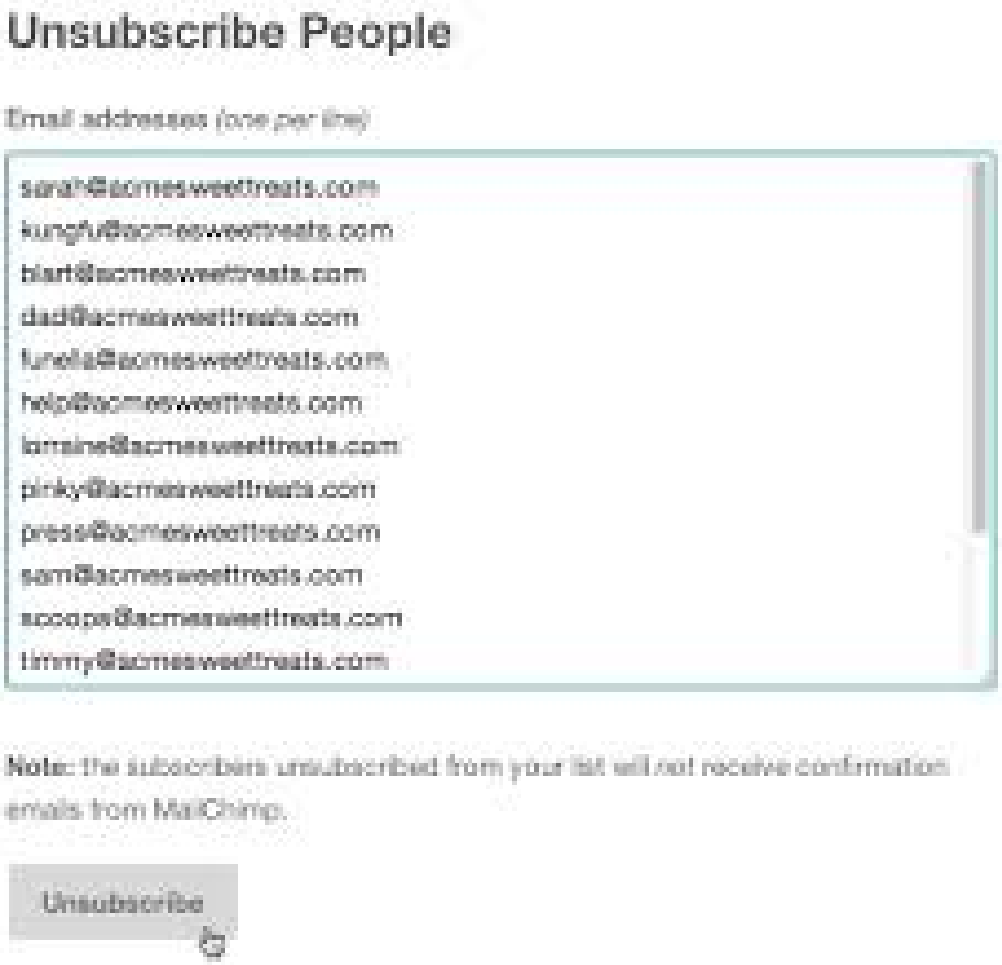
IS IT A LIST?

□

IS IT A LIST?

□

IS IT A LIST?


Unsubscribe People
Email addresses (one per line)
sarah@acmesweettreats.com
kungfu@acmesweettreats.com
blart@acmesweettreats.com
dad@acmesweettreats.com
funella@acmesweettreats.com
help@acmesweettreats.com
lorraine@acmesweettreats.com
pinky@acmesweettreats.com
press@acmesweettreats.com
sam@acmesweettreats.com
scoops@acmesweettreats.com
timmy@acmesweettreats.com
Note: the subscribers unsubscribed from your list will not receive confirmation emails from MailChimp.
Unsubscribe

□

WHAT IS A LIST?

A list	Not a list
	

List is a finite, ordered sequence of data items, known as elements. So, each element, except the last, has a unique successor.

$$<3, 4, 2, 6, 1>$$

WHAT CAN YOU DO ON A LIST?

- on the whole list
  - *clear* the list,
    clear($<3, 4, 2, 6, 1>$) $\Rightarrow <>$
  - *concatenate* two lists,
    $<3, 4, 2, 6, 1> + <5, 9> \Rightarrow <3, 4, 2, 6, 1, 5, 9>$
  - *sort* the list,
    sort($<3, 4, 2, 6, 1>$) $\Rightarrow <1, 2, 3, 4, 6>$
  - *length* of the list,
    length($<3, 4, 2, 6, 1>$) $\Rightarrow 5$
  - *reverse* the list,
    reverse($<3, 4, 2, 6, 1>$) $\Rightarrow <1, 6, 2, 4, 3>$
- on an element of the list

WHAT CAN YOU DO ON A LIST?

- on an element of the list
  - *cursor* at a position
    - move cursor to the beginning/first
      moveToFirst(<3, 4, 2, 6,| 1 >) ⇒<| 3, 4, 2, 6, 1 >
    - move cursor to the last
      moveToLast(<3, 4,| 2, 6, 1 >) ⇒ <3, 4, 2, 6, 1 |>
    - move cursor to position i
      moveToPos(<| 3, 4, 2, 6, 1 >,2) ⇒ <3, 4,| 2, 6, 1 >
    - move cursor forward
      next(<3, 4,| 2, 6, 1 >) ⇒ <3, 4, 2,| 6, 1 >
    - move cursor backward
      prev(<3, 4,| 2, 6, 1 >) ⇒ <3,| 4, 2, 6, 1 >
    - position of the cursor
      currPos(<3, 4,| 2, 6, 1 >) ⇒ 2

WHAT CAN YOU DO ON A LIST?

- on an element of the list
  - *get* value of the element at the cursor
    getValue(<3, 4,| 2, 6, 1 >) ⇒ 2
  - *insert* an element at the cursor/a position
    insert(<3, 4,| 2, 6, 1 >,9) ⇒ <3, 4,| 9, 2, 6, 1 >
  - *delete* an element at a position/the cursor
    remove(<3, 4,| 2, 6, 1 >) ⇒ <3, 4,| 6, 1 >

List ADT

```
template <typename T>
class List { // List ADT

public:
    List() {}
    virtual ~List() {}

    virtual void clear() = 0;
    virtual int length() const = 0;
    virtual void print() const = 0;
```

List ADT

```
    virtual void moveToStart() = 0;
    virtual void moveToEnd() = 0;
    virtual void moveToPos(int pos) = 0;
    virtual void prev() = 0;
    virtual void next() = 0;
    virtual int currPos() const = 0;

    virtual const T& getValue() const = 0;

    virtual void insert(const T& item) = 0;
    virtual void append(const T& item) = 0;

    virtual T remove() = 0;
};
```

EXAMPLE

- To iterate on the list to do something

```
for(L.moveToStart();
    L.currPos() < L.length();
    L.next()) {
    it = L.getValue();
    dosomething(it);
}
```

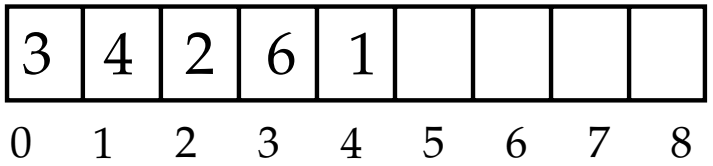
EXAMPLE

- To print all elements in the list

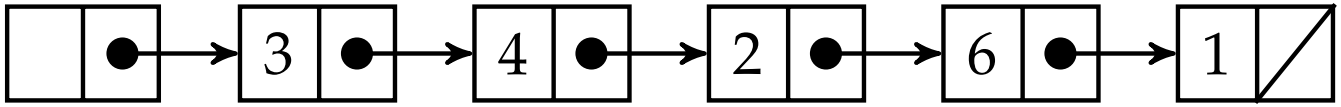
```
for(L.moveToStart();
    L.currPos() < L.length();
    L.next()) {
    it = L.getValue();
    cout << it << " ";
}
```

IMPLEMENTATION

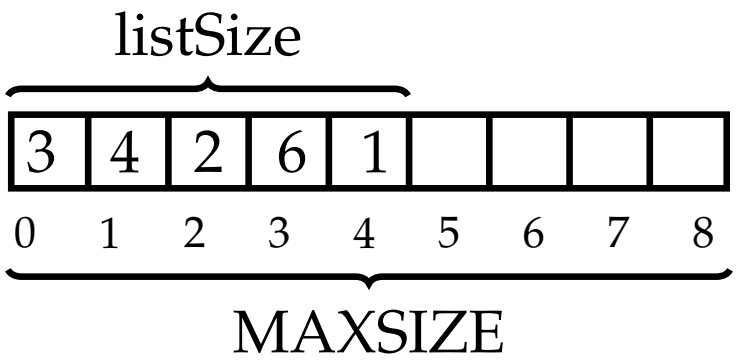
- Array-based List



- Linked List



ARRAY-BASED LIST



- How to implement a cursor?
- How to move the cursor?
- How to get the value of a list at the cursor position?
- What is the complexity of these actions?

Array-based List Implementation

```
template <typename T>
class AList : public List<T> {
    private:
        T * arr;
        int listSize = 0;
        int cursor = 0;
        const int MAXSIZE = 50;
    public:
        AList() {
            arr = new T[MAXSIZE];
        }
        ...
};
...
AList<int> x;
AList<int>* p = new AList<int>();
```

MOVE THE CURSOR TO POSITION POS

```
void moveToPos(int pos) {
    Assert((pos>=0)&&(pos<=listSize),"Pos out of range")
    cursor = pos;
}
```

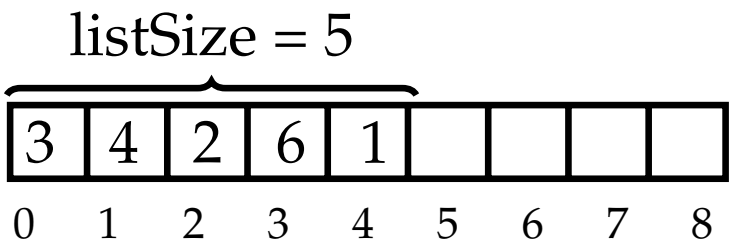
What is the time complexity of this method?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST

3	4	2	6	1				
0	1	2	3	4	5	6	7	8

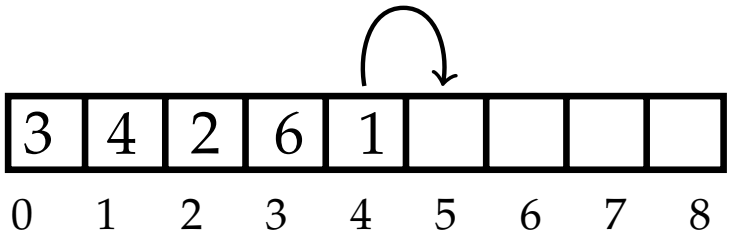
- insert($<3, 4, | 2, 6, 1>, 9$)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



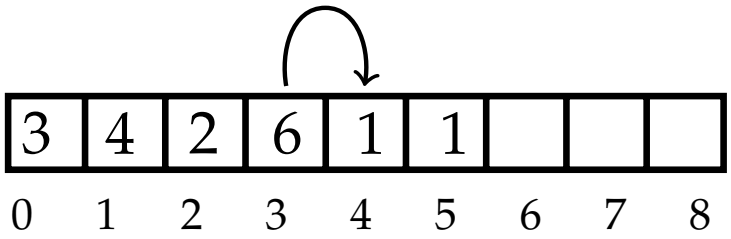
- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST

3	4	2	6	1	1			
0	1	2	3	4	5	6	7	8

- insert($<3, 4, | 2, 6, 1>, 9$)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



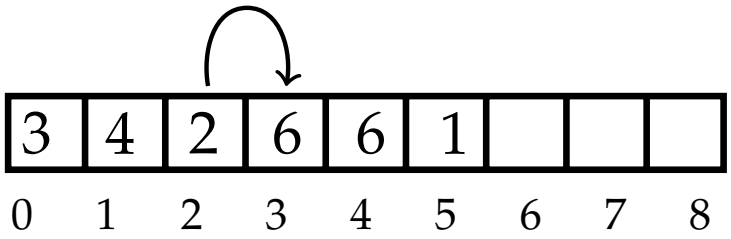
- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST

3	4	2	6	6	1			
0	1	2	3	4	5	6	7	8

- insert($<3, 4, | 2, 6, 1>, 9$)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



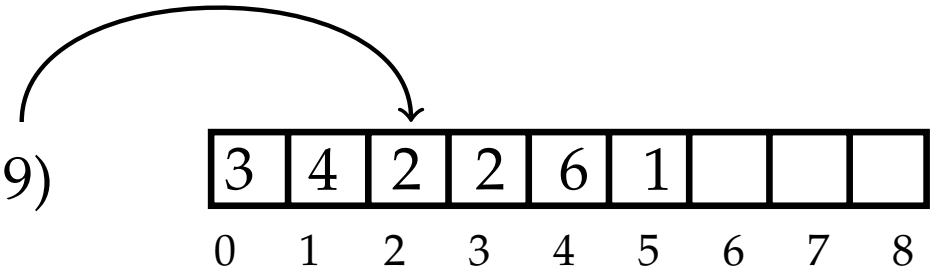
- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST

3	4	2	2	6	1			
0	1	2	3	4	5	6	7	8

- insert($<3, 4, | 2, 6, 1>, 9$)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



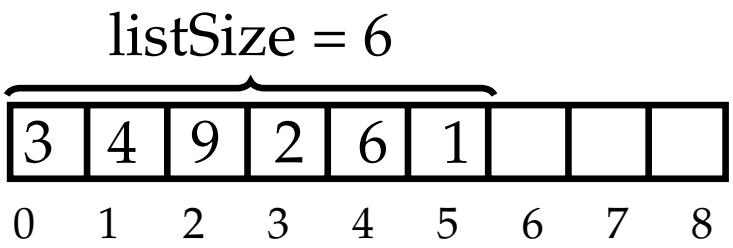
- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST

3	4	9	2	6	1			
0	1	2	3	4	5	6	7	8

- insert($<3, 4, | 2, 6, 1>, 9$)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

INSERT A NEW ELEMENT TO AN ARRAY-BASED LIST



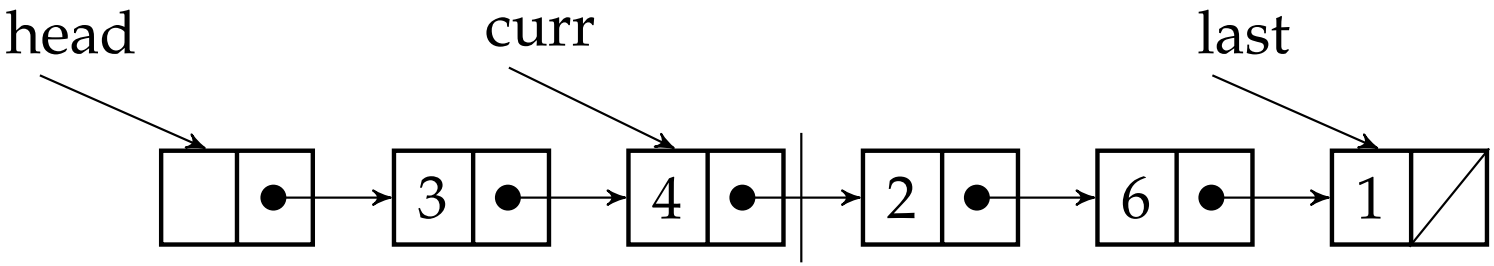
- insert(<3, 4, | 2, 6, 1 >, 9)
- cursor = 2
- What is the worst case complexity of the insertion algorithm?
- Write the insertion algorithm?

Insert Method

```
void insert(const T& it) {
    Assert(listSize < MAXSIZE, "List capacity exceeded");
    for(int i=listSize; i>cursor; i--)
      arr[i] = arr[i-1];  //Shift elements up
    arr[cursor] = it;       // to make room
    listSize++;           // Increment list size
}
```

Linked List

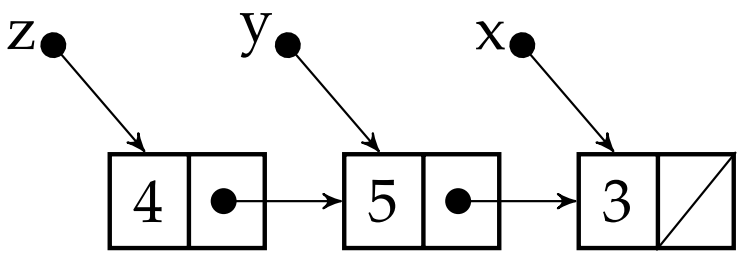
<3, 4, | 2, 6, 1 >



LINKED LIST ELEMENT

```
template <typename T>
class Link {
   public:
      T data;
      Link<T>* next;
      //Constructors
      Link(const T& dataval, Link<T>* nextval=NULL)
      {
         data = dataval;
         next = nextval;
      }
      Link(Link<T>* nextval = NULL)
      {
         next = nextval;
      }
}
```

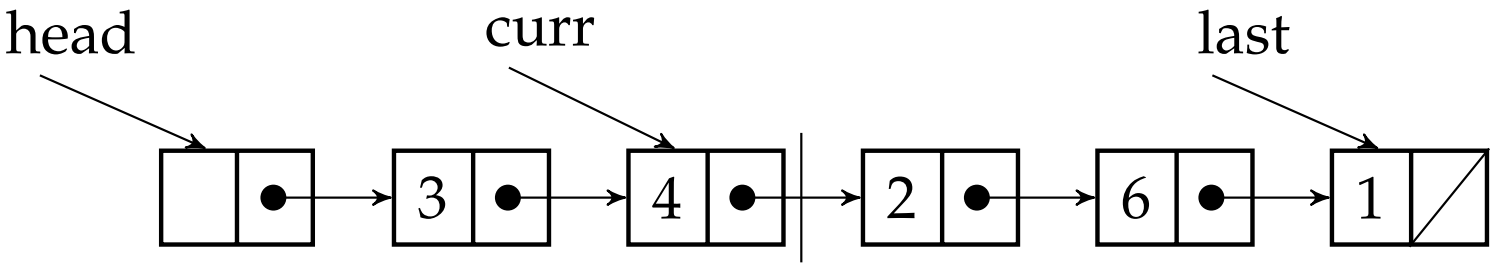
Example




```
Link<int> * x = new Link<int>(3);
Link<int> * y = new Link<int>(5,x);
Link<int> * z = new Link<int>(y);
z -> data = 4;
cout << z -> data << '\n';
cout << z -> next -> data << '\n';
cout << z -> next -> next -> data << '\n';
```

Linked List

<3, 4, | 2, 6, 1 >



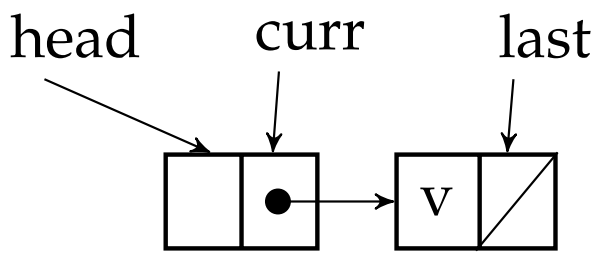
LINKED LIST TYPE

```
template <typename T>
class LList: public List<T>{
   private:
      Link<T>* head;
      Link<T>* last;
      Link<T>* curr;
      int   listSize;
   public:
      LList() {
         head = last = curr = new Link<T>(NULL);
         listSize = 0;
      }
      ...
};
```

Example

Write a constructor that creates a linked list with one element whose value is passed through the parameter of the constructor.

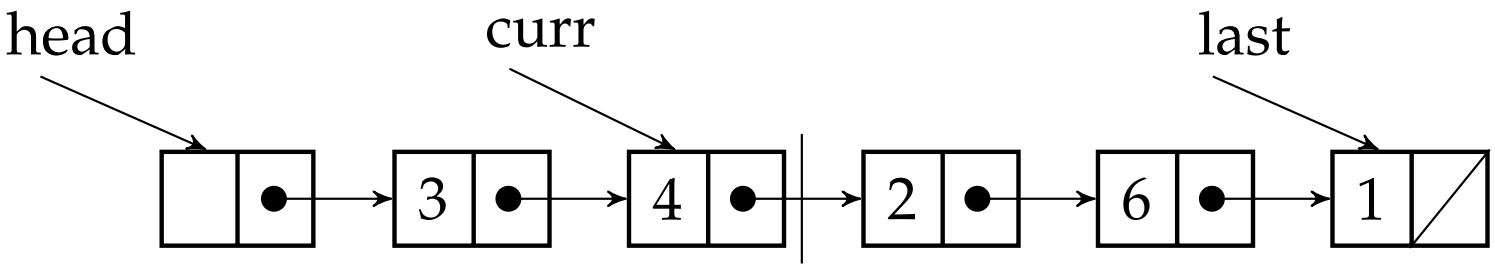
```
LList(const T& val)
```




```
LList(const T& val) {
    last = new Link<T>(val);
    head = curr = new Link<T>(last);
    listSize = 1;
}
```

INSERT AN ELEMENT INTO A LINKED LIST

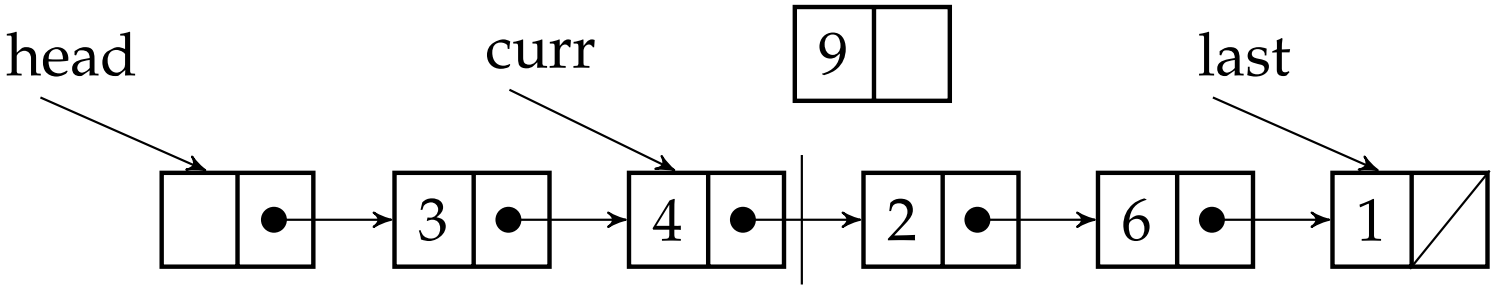
insert(<3, 4, | 2, 6, 1 >,9) ⇒ <3, 4, | 9, 2, 6, 1 >



What is the complexity of the insertion algorithm?

INSERT AN ELEMENT INTO A LINKED LIST

insert(<3, 4, | 2, 6, 1 >,9) ⇒ <3, 4, | 9, 2, 6, 1 >

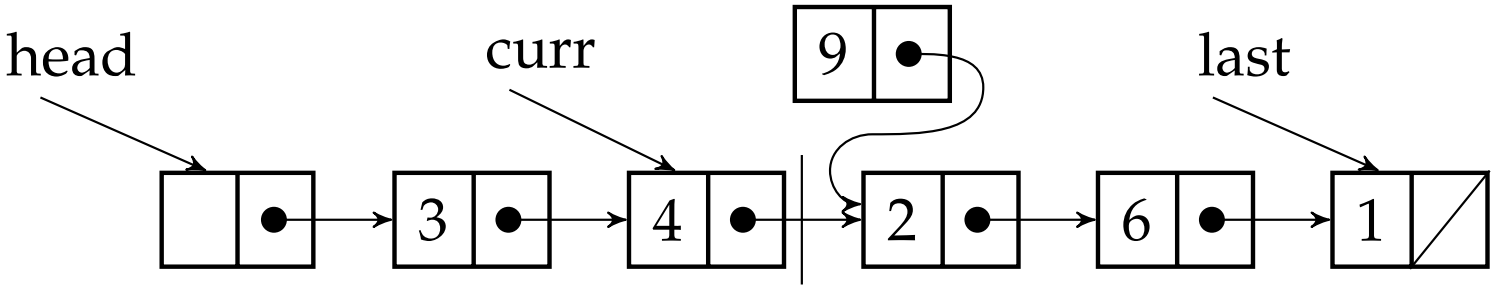


1.create a new node containing the new data

What is the complexity of the insertion algorithm?

INSERT AN ELEMENT INTO A LINKED LIST

insert(<3, 4, | 2, 6, 1 >,9) ⇒ <3, 4, | 9, 2, 6, 1 >

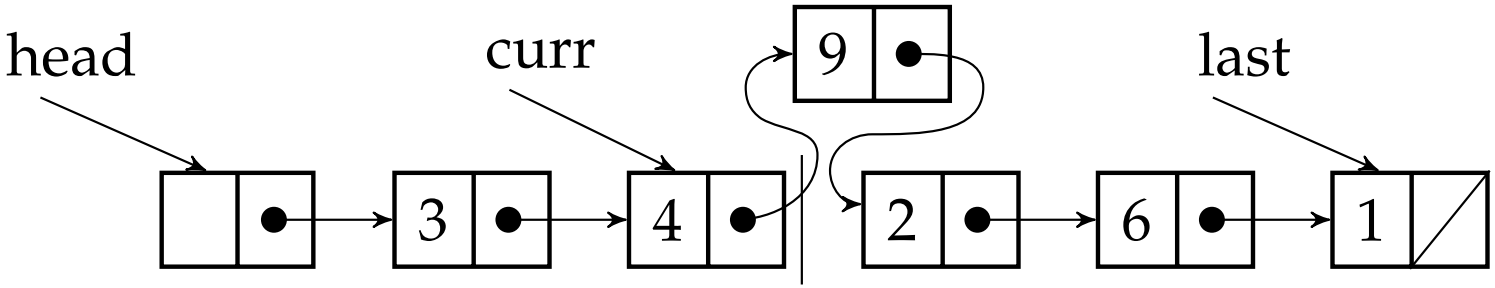


2. its *next* points to *current* node

What is the complexity of the insertion algorithm?

INSERT AN ELEMENT INTO A LINKED LIST

insert(<3, 4, | 2, 6, 1 >,9) ⇒ <3, 4, | 9, 2, 6, 1 >



3. change *curr* → *next* to the new node.

What is the complexity of the insertion algorithm?

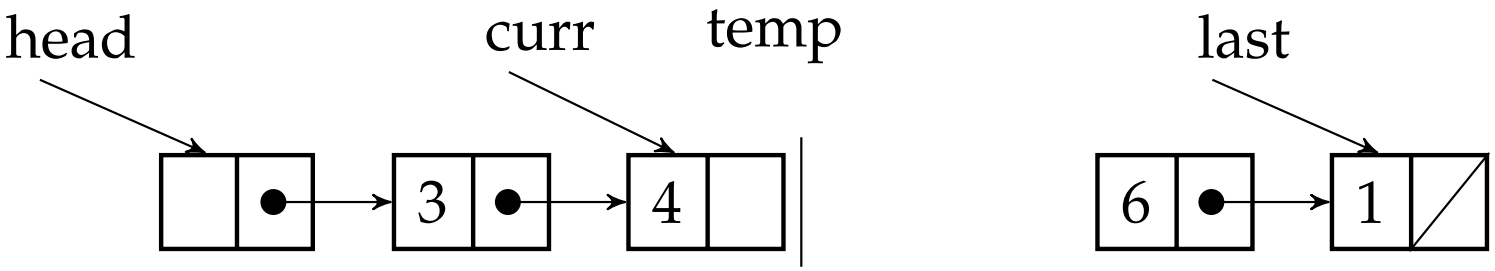
INSERTION ALGORITHM

1. create a new node containing the new data
2. its *next* points to *current* node
3. change *curr* → *next* to the new node.

```
void insert(const T& it) {
    curr->next = new Link<T>(it, curr->next);
    if (last == curr)
        last = curr->next;
    listSize++;
}
```

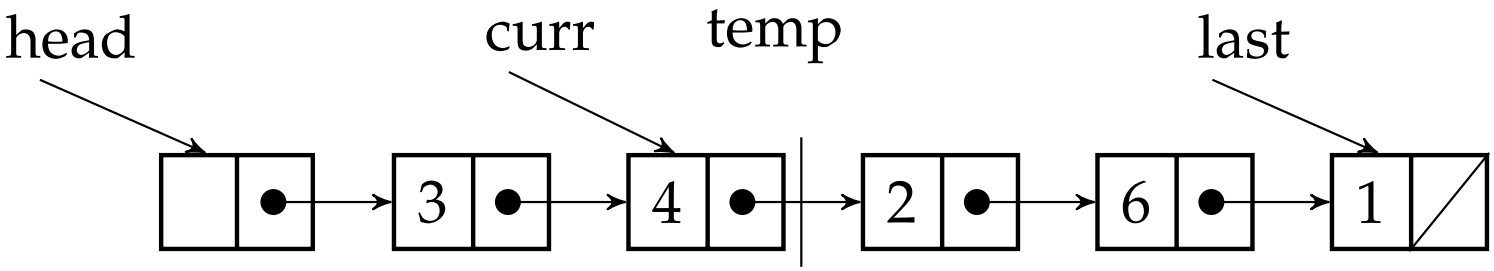
DELETE THE ELEMENT AT THE CURSOR

remove(<3, 4, | 2, 6, 1 >) ⇒ <3, 4, | 6, 1 >



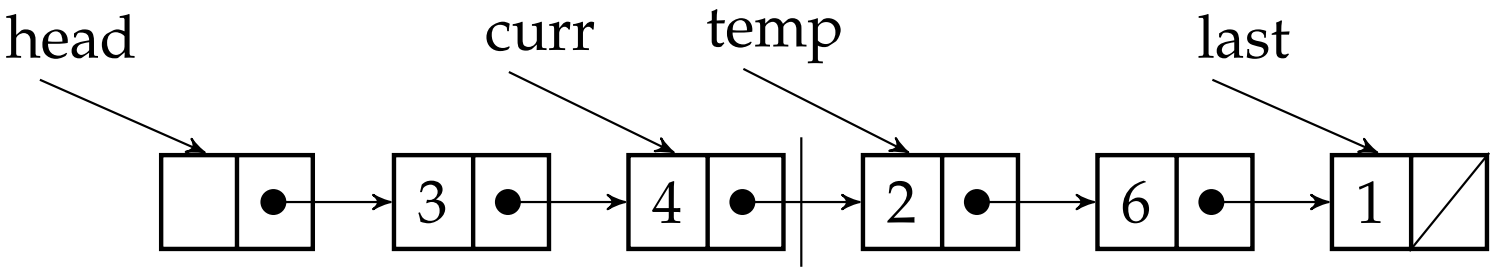
DELETE THE ELEMENT AT THE CURSOR

remove(<3, 4, | 2, 6, 1 >) ⇒ <3, 4, | 6, 1 >



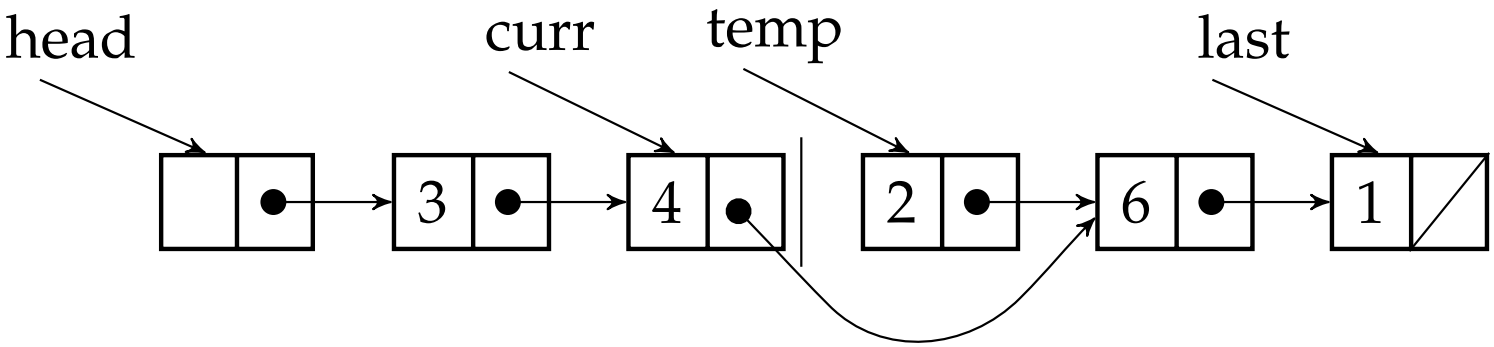
DELETE THE ELEMENT AT THE CURSOR

remove(<3, 4, | 2, 6, 1 >) ⇒ <3, 4, | 6, 1 >



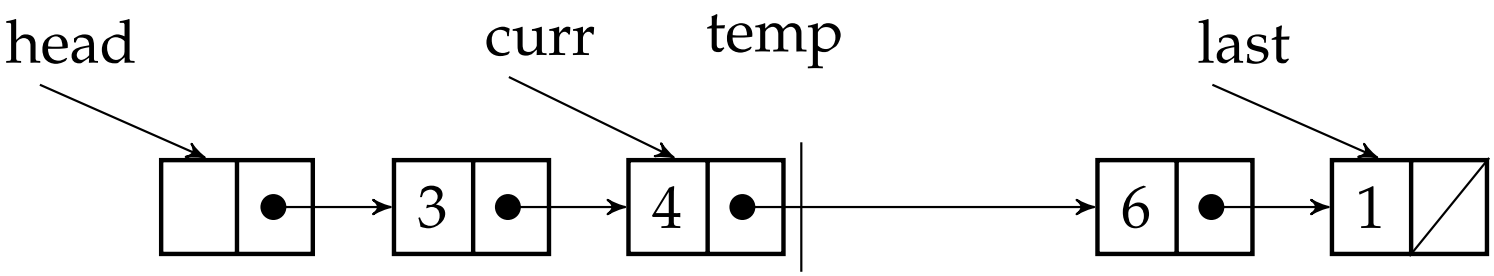
DELETE THE ELEMENT AT THE CURSOR

remove(<3, 4, | 2, 6, 1 >) ⇒ <3, 4, | 6, 1 >



DELETE THE ELEMENT AT THE CURSOR

remove(<3, 4, | 2, 6, 1 >) ⇒ <3, 4, | 6, 1 >



What is the complexity of the removal algorithm?

What should do when killing a linked list?

Remove all elements of the list before killing the list

```
~LList(){
    while (head != NULL) {
        curr = head;
        head = head -> next;
        delete curr;
    }
}
```