CS 180 - Basic Linked Lists

Note Title 9/19/2011

Announcements

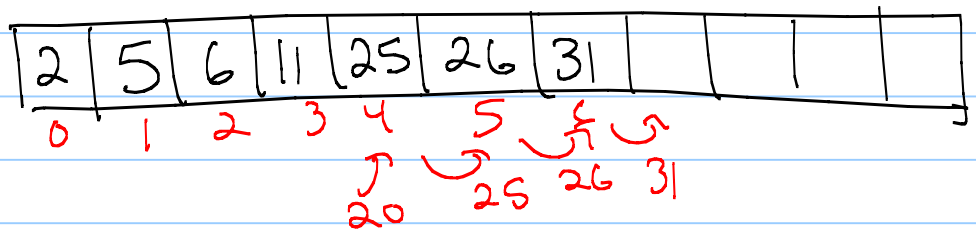
- Turing is dead
- HW2 is now on paper, due next Monday
- No lab tomorrow — lab on Friday

Recap of arrays (Ch 3.1 of text)

Limits

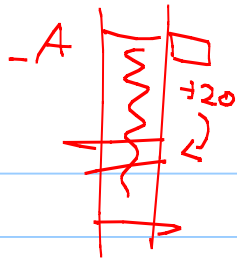
- not very flexible
  - size is fixed at creation
  - 1 kind of data
  - inserting + moving can be difficult

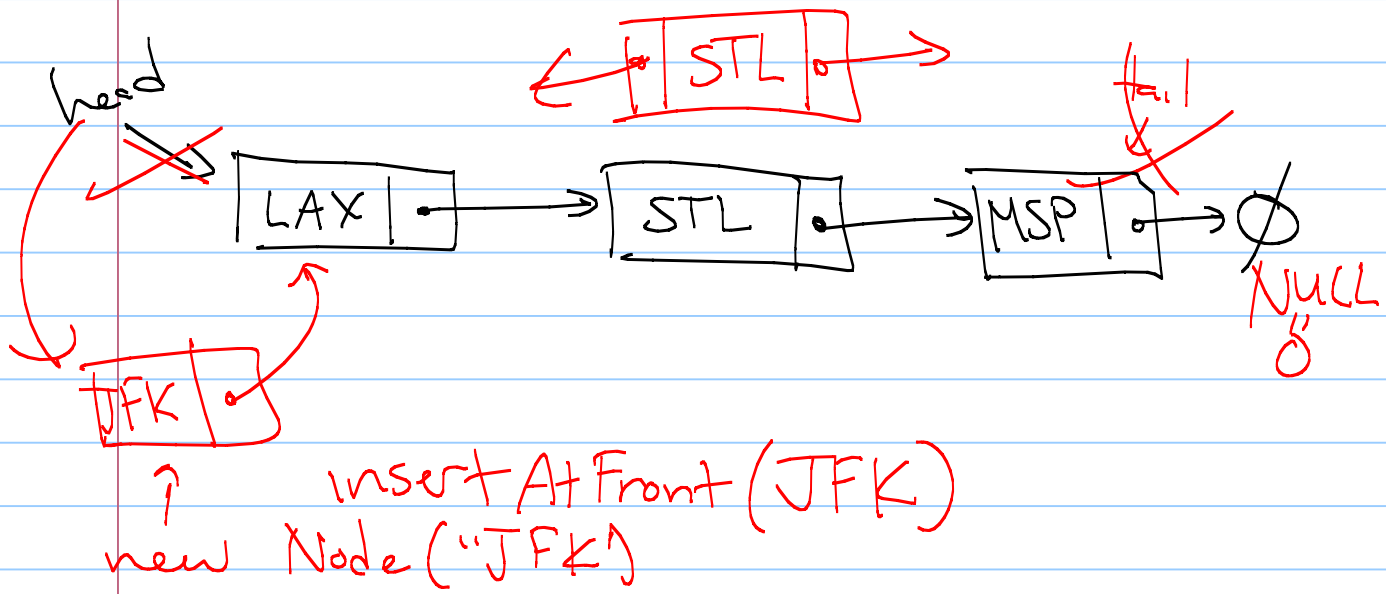
Q: How would we insert an element in the middle of an array?
ex: insert(20) in sorted order

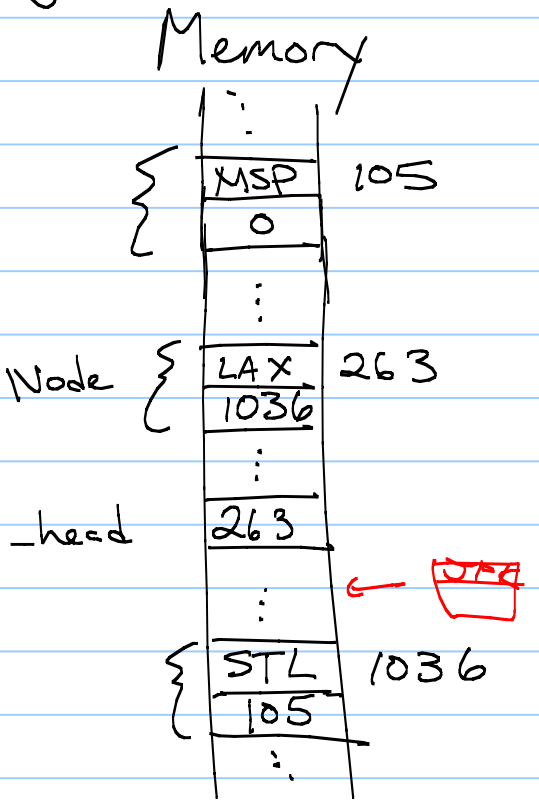


Singly Linked Lists

A collection of nodes that together form a linear ordering.







Functions

What might we want to do?

- append
- insert
- remove
- find
- extend (to combine lists)
- sort
- search

Implementation

We want nodes which store what?

Node class : data ← could be any type
pointer to next

templates

What will our private data be?

(in Linked List class)

- head (a pointer to a node)

Code - Node class & private data

in book, use E

```
template <typename Object>
class SLinkedList {

  private:

    class SNode {
      private:
        Object _elem;
        SNode<Object> * _next;
        // no functions or public data
    };

    SNode<Object> * _head;
```

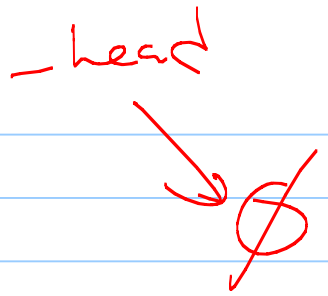
Functions — .h file

```
public:

    SLinkedList();
    ~SLinkedList();
    bool empty() const;
    const Object& front() const;
    void addFront(const Object& e);
    void removeFront();

};
```

Constructor – cpp file




```
template <typename Object>
SLinkedList<Object>::SLinkedList() {

    _head = NULL;

}
```

```
_head = 0;
```

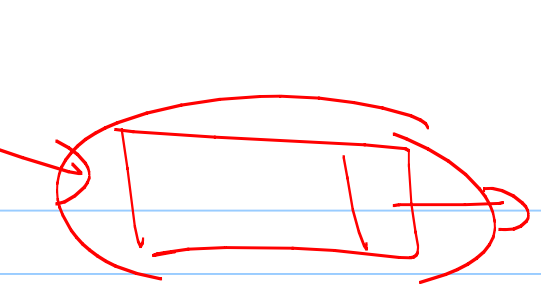
OR:

```
template <typename Object>
SLinkedList<Object>::SLinkedList() : _head(0) { }
```

empty

```
template <typename Object>
bool SLinkedList<Object>::empty() {
    return (_head == NULL);
}
```

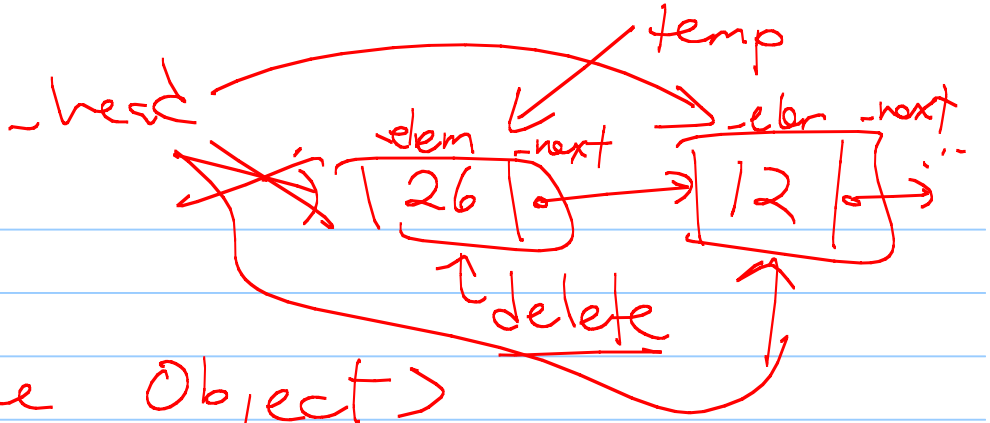
front :




```
template <typename Object>
const Object& SLinkedList<Object>::front() {
    return _head->_elem;
}
```

(*_head)._elem

removeFront




```
template <typename Object>
void SLinkedList<Object>:: removeFront() {
    SNode<Object> * temp;
    temp = _head;
    _head = _head -> _next;
    delete temp;
}
```

Destructor

addFront