OTCR Consulting

Key Statistics

- 41% of all OTCR members have been in the Hoeft Technology & Management minor
- 17% go to the Big 3 consulting out as a first job
- 21% go Deloitte Technology Consulting as a first job
- OTCR-wide technology firm placements: Dell (17%),
- Microsoft (31%), Google (29%), Piazza (13%)

Clients

Information Session

- Monday, February 11th
- 213 Greg Hall
- 7:00pm
- Dress: business casual (slacks and a nice shirt)

About Us

- http://otcr.uiuc.edu/
- Or email me:

  nathaniel.may22@gmail.com

OTCR CONSULTING

Announcements

MP3 available, due 2/22, 11:59p. EC: 2/15, 11:59p.

MP 3.1 will be on Exam 1.

Exam 1: 2/19, 7-10p, in rooms tba. 75min exam, given 3hr.

Class cancelled 2/18.

Review session - 2/18, 12-2p, Siebel 1404.

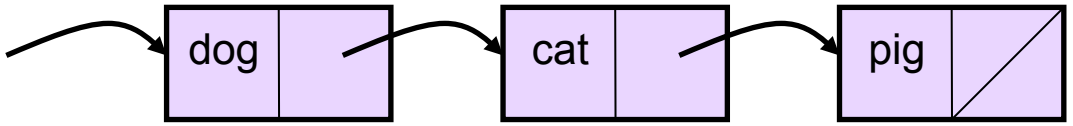
MP2 solution party: Sat, 2/16, 10a, Siebel 1404.

Review session: Sun, 2/17, 5p, Siebel 1404.

TODAY: more linked memory

ADT - lists

Example 1: `insertAtFront<farmAnimal>(head,cow);`




```
void insertAtFront(listNode *  curr, LIT e) {
```

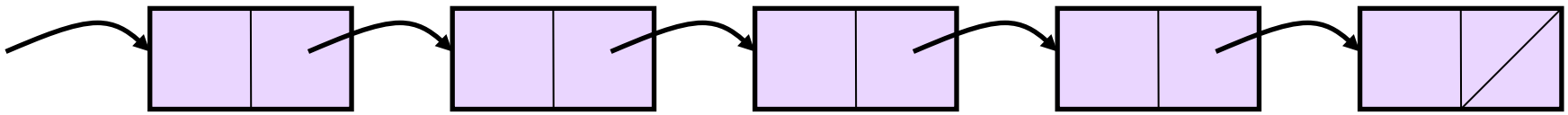
```
}
```

Running time?

```
struct listNode {
   LIT data;
   listNode * next;
   listNode(LIT newData):data(newData),next(NULL){}
}
```

8 4 2 6 3 0 1 2

Example 2:

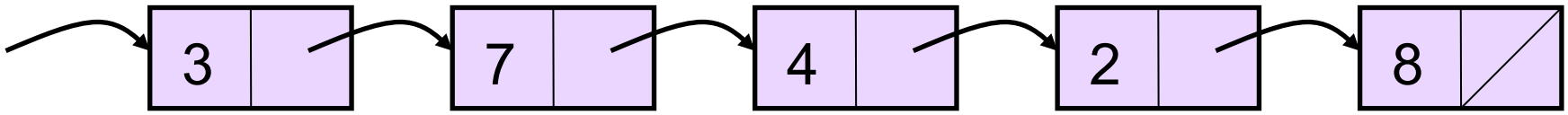


void printReverse(listNode * curr) {

Running time?

```
struct listNode {
   LIT data;
   listNode * next;
   listNode(LIT newData):data(newData),next(NULL){}
}
```

Example 3: Find kth position (we'll need this later)



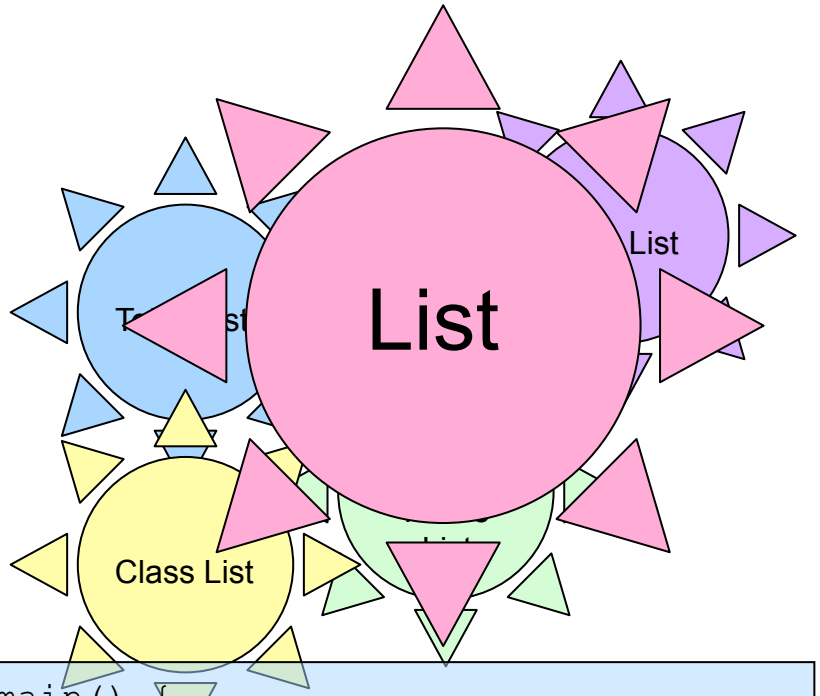

```
//returns pointer to node k steps forward from *curr
listNode * findKth(listNode *  curr, int k) {

}
```

Analysis:

Find kth in array:

Abstract Data Types (an example):




```
template<class LIT>
class List {
public:
     List();
     //~List();
     int getSize() const;
     void insert(int loc, LIT e);
     void remove(int loc);
     LIT const & getItem(int loc) const;
private:
     //my little secret
};
```

```
int main() {
     List<int> myList;
     myList.insert(1,4);
     myList.insert(1,6);
     myList.insert(1,8);
     myList.insert(3,0);
     myList.insert(4,myList.getItem(2));
     cout << myList.getSize() << endl;
     myList.remove(2);
     cout << myList.getItem(3) << endl;
     return 0;
}
```

ADT List, implementation 1:

```
template<class LIT>
class List {
public:
     List():size(0){}
     //~List();
     int getSize() const;
     void insert(int loc, LIT e);
     void remove(int loc);
     LIT const & getItem(int loc) const;
private:
     LIT items[8];
     int size;
};
```

0	1	2	3	4	5	6	7

```
template<class LIT>
int List<LIT>::getSize() const {
   return size;
}
template<class LIT>
void List<LIT>::insert(int loc, LIT e){
if ((size + 1) < 8) {
   LIT go = e;
   int it = loc-1;
   while (it < size+1){
     LIT temp = items[it];
     items[it] = go;
     go = temp;
     it ++;
   }
   size ++;
}}
template<class LIT>
void List<LIT>::remove(int loc) {
if (size > 0) {
   int it = loc-1;
   while (it < size){
     items[it] = items[it+1];
     it ++;
   }
   size --;
}}
template<class LIT>
LIT const & List<LIT>::getItem(int loc)
const {return items[loc -1];}
```

Don't look at this code!

Don't look at this code!

Implementing a list using an array:

0	1	2	3	4	5	6	7

0	1	2	3	4	5	6	7

ADT List, implementation 2:

```
template<class LIT>
class List {
public:
     List():size(0),head(NULL){}
     ~List(); // also copy constructor, assignment op
     int getSize() const;
     void insert(int loc, LIT e);
     void remove(int loc);
     LIT const & getItem(int loc) const;
private:
     listNode * head;
     int size;
     listNode * Find(listNode * place, int k);
     struct listNode {
        LIT data;
        listNode * next;
        listNode(LIT newData
     }
}
```

```
template<class LIT>
listNode * List<LIT>::Find(listNode * place, int k){
if ((k==0) || (place==NULL))
   return place;
else
   return Find(k-1, place->next);
}
```