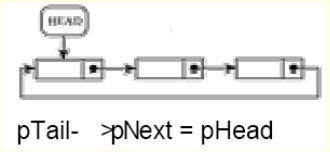
Lecture 9.3

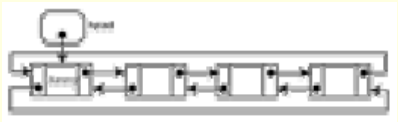
Linked List (continuation)

Circular lists



- The last node links back to the first.
- Traversing the list stops when you arrive back at the head of the list.

Circular doubly linked lists



- Reduces all special cases to inserting or removing between nodes

Implementing Linked Lists with classes

- Convert your struct into a class.
- Create a class for your list, and the functions of the previous implementation into methods for the list class.

The Node class

```
class Node{
    friend class List; // gives List class access to private
                       // members of Node class
    private:
        int nData;
        Node* pNext;
        Node* pPrev;
    public:
        Node ( int data ) { nData = data;
        pNext = NULL;
        pPrev = NULL; }
        int getData( ) { return nData; }
};
```


The List class

```
class List{
    private:
        Node* pHead;
        Node* pTail;
        Node* createNode ( int data ) ;
    public:
        List ();
        ~List();
        void appendNode ( int value );
        void insertNode ( int value, Node *pAfter);
        void removeNode (Node *pNode);
        bool isEmpty ( );
        void printList ( );
};
```


List Constructor and Destructor

```
List::List ( ) {
    pHead=NULL;
    pTail=NULL;
}
List::~List ( ) {
    while ( !isEmpty ( ) ) // keep on removing until the
                           // head points to NULL
    removeNode( pHead );
    cout << "List deleted\n";
}
```


Extra methods

```
Node* List::createNode ( int data ) {
    //allocate memory for new node and
    //intialize value to data
    Node* pNode = new Node ( data ) ;
    return pNode;
}

bool List::isEmpty ( ) {
    return pHead == NULL;
}
```

Changes to Append method

```
void List::appendNode ( int value )
{
   Node* pNode = createNode ( value ) ;
   if ( isEmpty ( ) ) { // if list is empty
        pHead = pNode; // make head point to pNode
        pNode->pPrev = NULL;
        …
        …
}
```


The main

```
int main()
{
        List sampleList;
        // Add items to linked list
        for (int i = 0; i < 100; i++)
          sampleList.appendNode ( i ); //add node to list
                                       //print the list
        sampleList.printList();
        system ("pause");
        return 0;
}
```