CS 162
Intro to CS 2

Doubly Linked lists

Circular Lists

OSU Oregon State University

Limitations

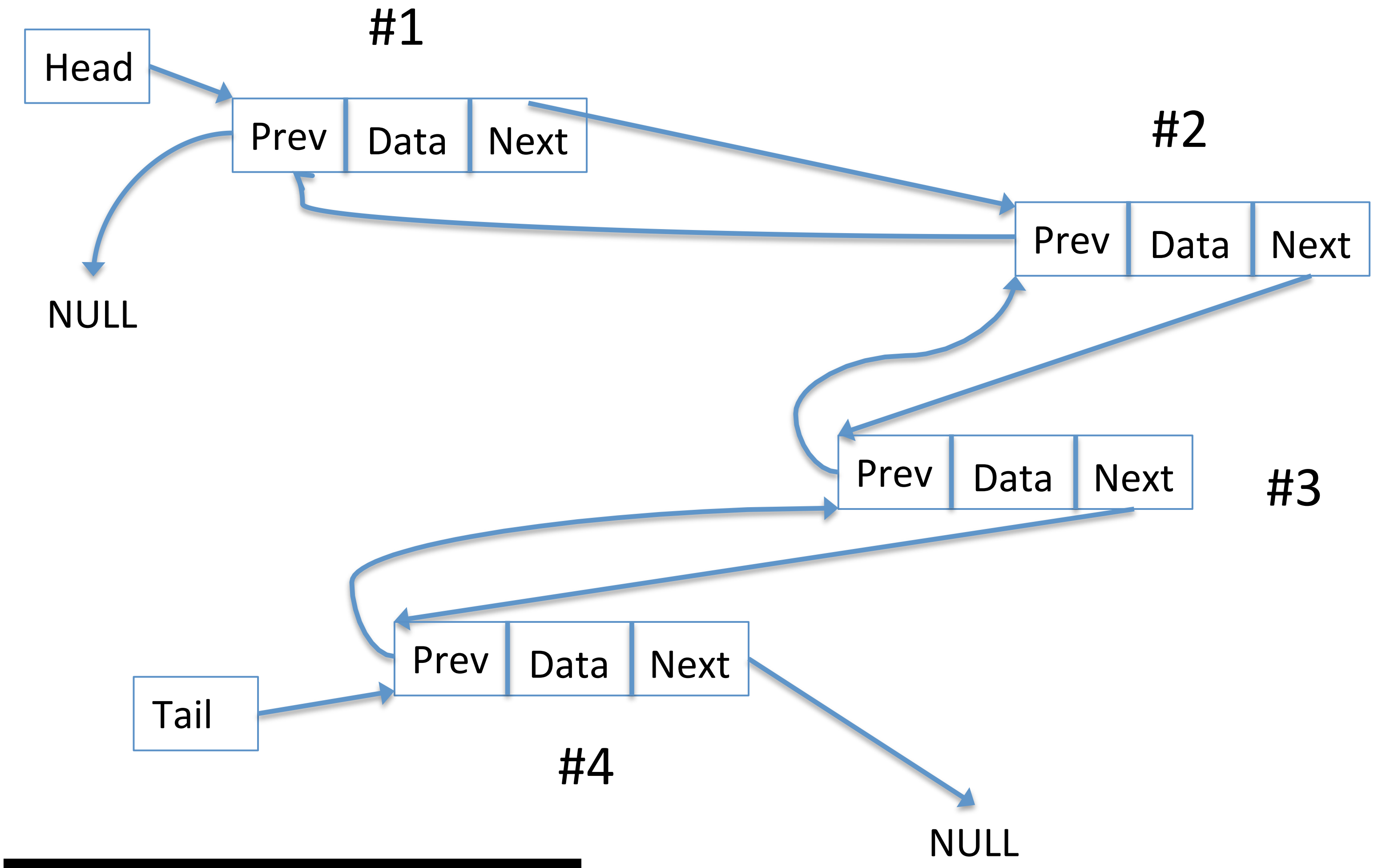
- Singly linked lists
  - You can only go from first towards last
  - Have fixed first and last elements

- Solutions?
  - Double links so you have a node after AND before
  - Circular structure so there is no fixed first and last elements

Doubly Linked List

- A node has:
  - A data element
  - A pointer (link) to the next node
  - A pointer (link) to the previous node
- Still requires a pointer to the head of the list
- May require a pointer to the end of the list
- You can go backwards and forwards through list elements

Graphically



Using a Struct

```
 Struct  IntDoubleNode
 {
 private:
  int data;
  IntDoubleNode *previous;
  IntDoubleNode *next;
};
```

Necessary functions are written separately

Doubly Linked List Class

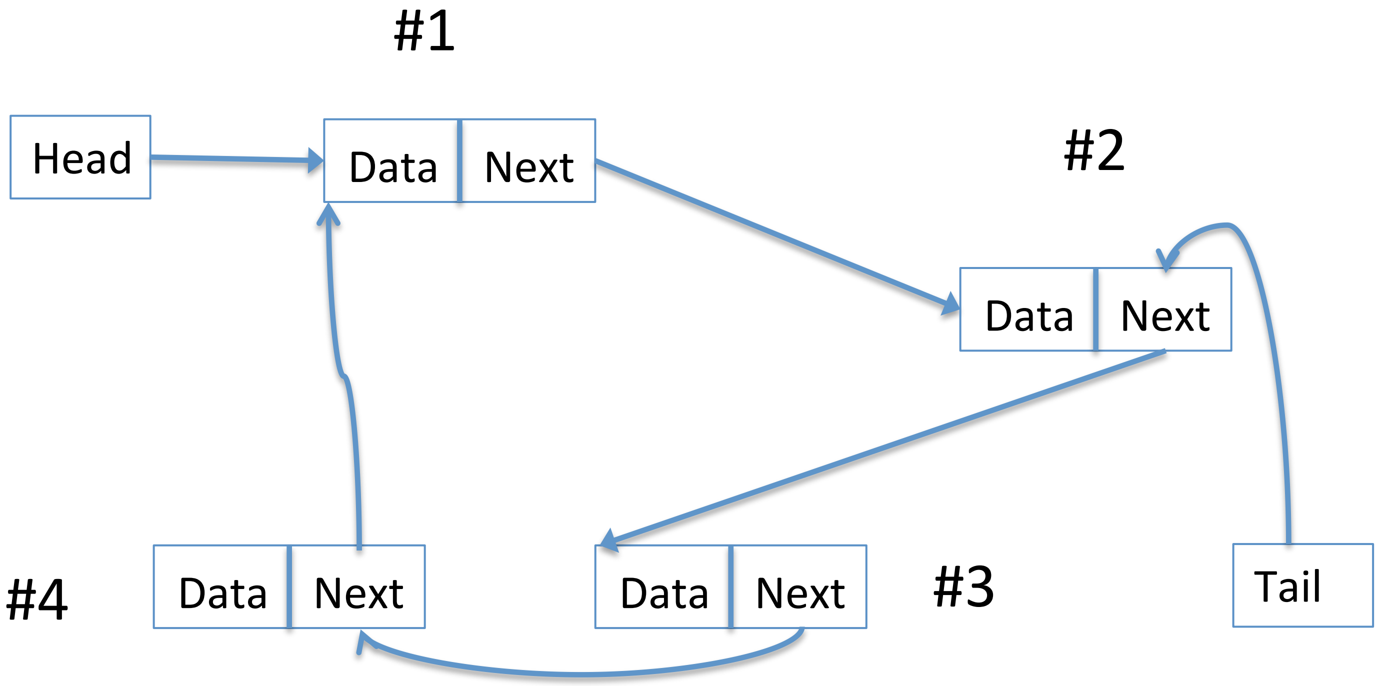
```
class DoublyLinkedIntNode
{
public:
    DoublyLinkedIntNode ( ){}
    DoublyLinkedIntNode (int theData, DoublyLinkedIntNode* previous,
                DoublyLinkedIntNode* next)
            : data(theData), nextLink(next), previousLink(previous) {}
    DoublyLinkedIntNode* getNextLink( ) const { return nextLink; }
    DoublyLinkedIntNode* getPreviousLink( ) const { return previousLink; }
    int getData( ) const { return data; }
    void setData(int theData) { data = theData; }
    void setNextLink(DoublyLinkedIntNode* pointer) { nextLink = pointer; }
    void setPreviousLink(DoublyLinkedIntNode* pointer)
    { previousLink = pointer; }
private:
    int data;
    DoublyLinkedIntNode *nextLink;
    DoublyLinkedIntNode *previousLink;
};
typedef DoublyLinkedIntNode* DoublyLinkedIntNodePtr;
```

From the publisher's slides

Circular Linked List Overview

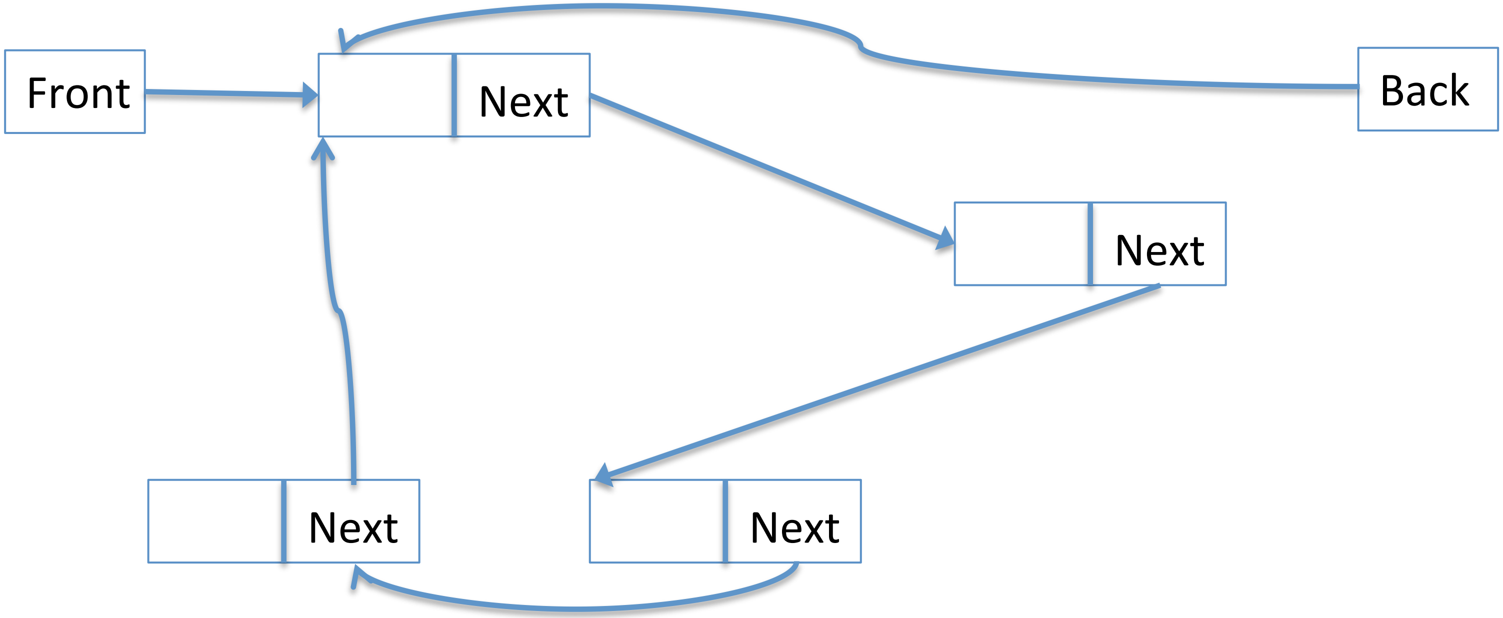
- To make a circular list, the last link points to head rather than NULL
- Head pointer now points to start of data rather than first element
- May need a tail pointer to indicate end of data
- At this point there is no first or last element

Circular Linked list



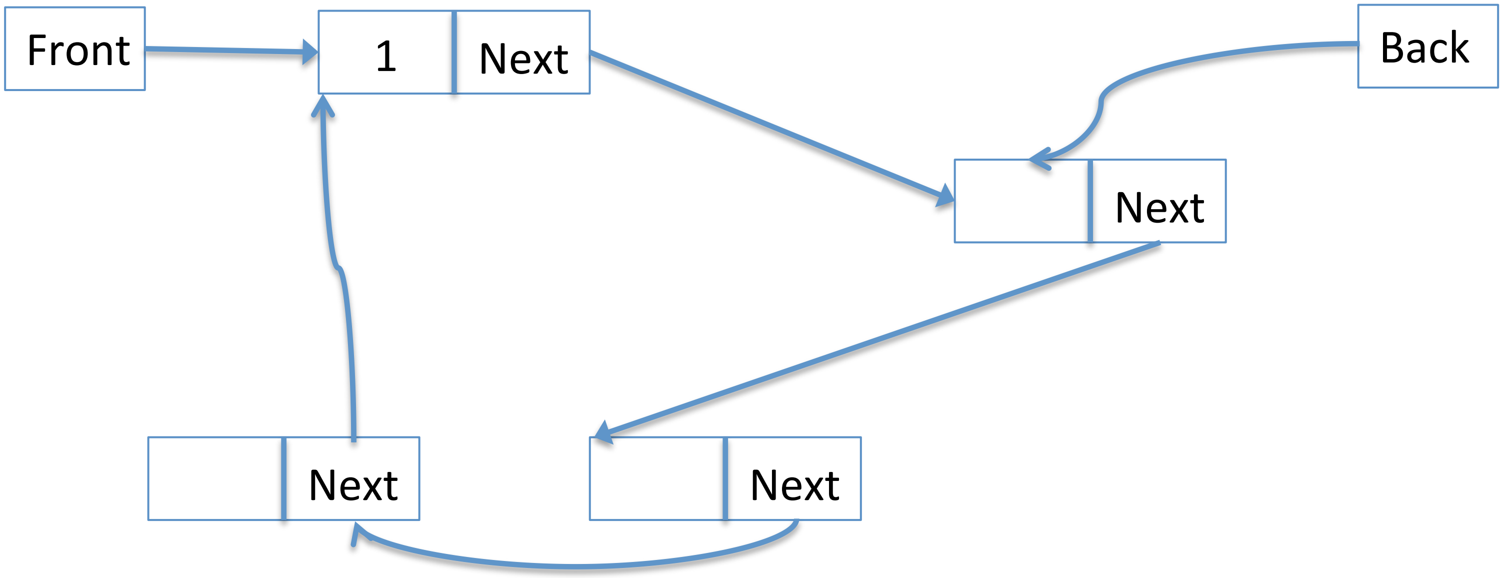
Head and Tail now refer to the contents and not the physical structure

Queue Example



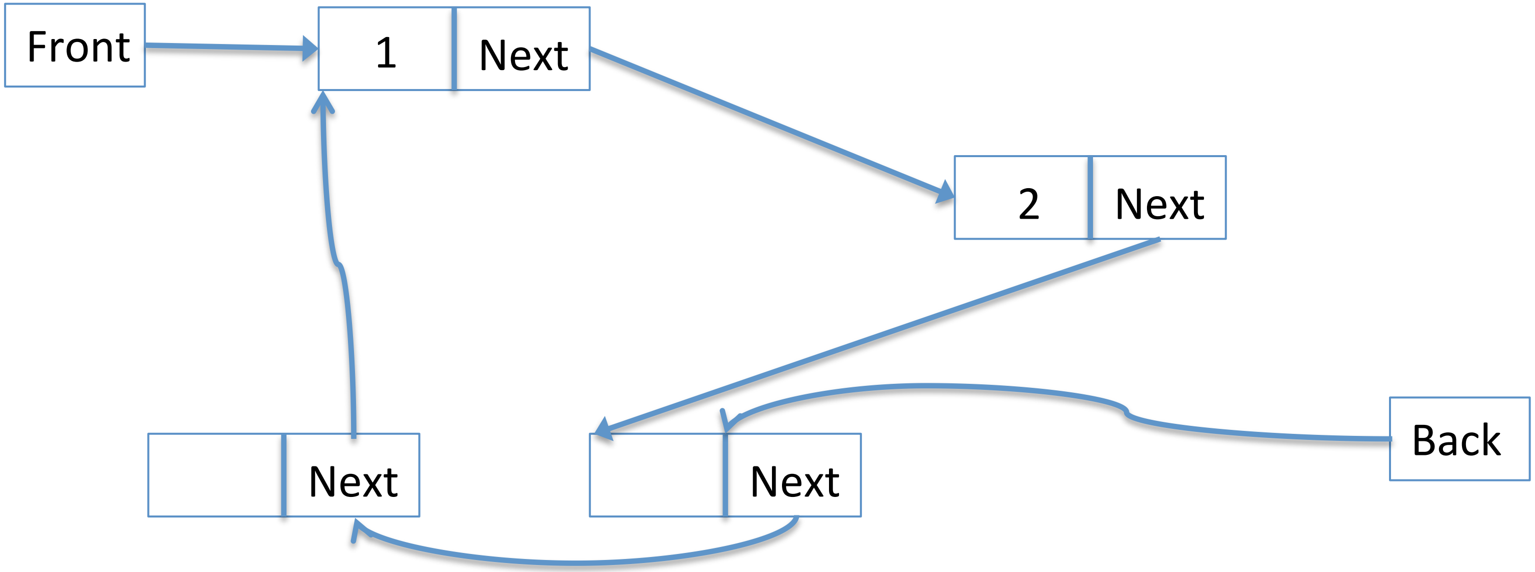
Queue is empty

Queue Example



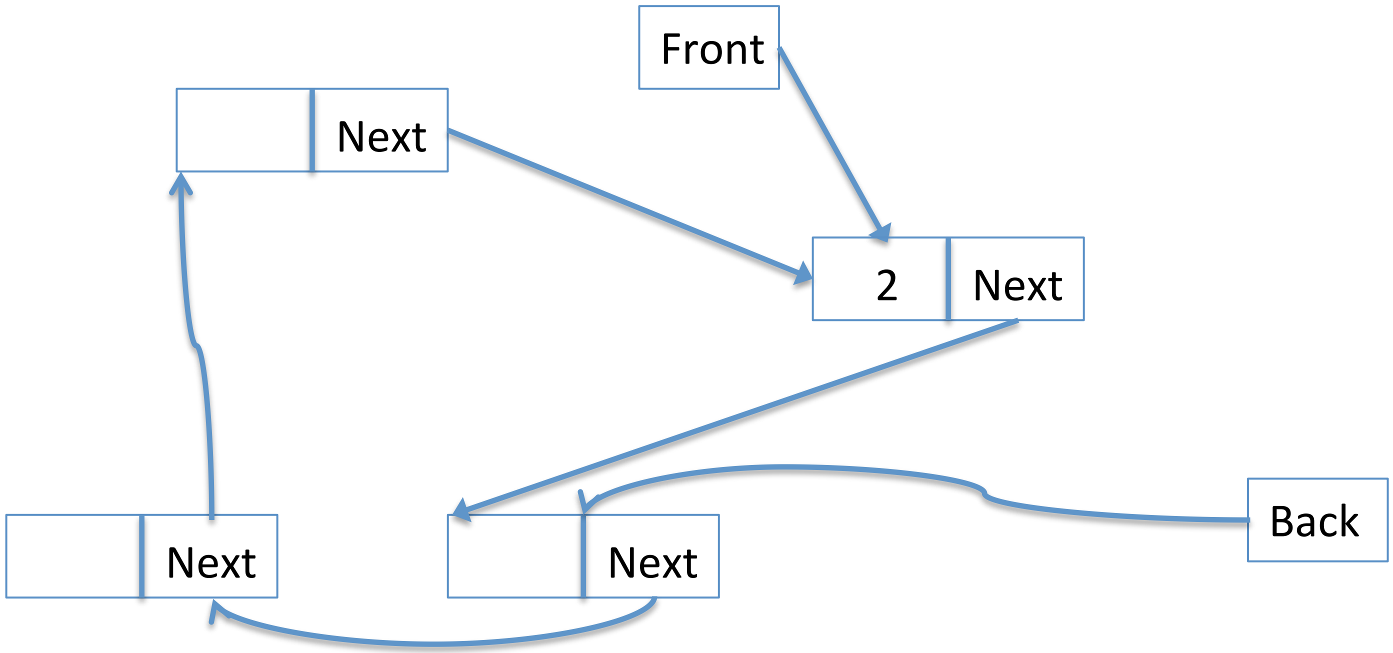
Queue has one number

Queue Example



Queue has two numbers

Queue Example



Remove front of queue

Better!

- Consider using an array as DS for a queue ADT
  - You fill and take off

    123-- > 1234- > -234- > --345 > 6-345
  - You need to handle the wrap around

- Using a circular list avoids the wrap around
  - Add nodes as you need them
  - It naturally wraps around!!

Summary

- Simple changes can have big impact
- Point to the first node rather than NULL
- 2 link elements allows you to go both ways
- In a later class you’ll see that 2 link elements will also allow you to create binary trees
  - A hierarchical structure where each node can have 0, 1, or 2 children