Linked List Implementation of the Deque
Deque Interface (Review)

```c
int   isEmpty();
void  addFront(TYPE val);  // Add value at front of deque.
void  addBack (TYPE val);   // Add value at back of deque.
void  removeFront();        // Remove value at front.
void  removeBack () ;       // Remove value at back.
TYPE  front();             // Get value at front of deque.
TYPE  back();              // Get value at back of deque.
```
• What if we want to add and remove elements from both front and back?

Linked List Deque
Modification #3: Double Links

- Point forward to the **next** element
- Point backwards to the **previous** element

```c
struct DLink {
    TYPE val;
    struct DLink *next; /* Link to prev node. */
    struct DLink *prev; /* Link to next node. */
};
```
struct linkedList {
    int size;
    struct dlink * frontSentinel;
    struct dlink * backSentinel;
};
void LinkedListInit (struct linkedList *q) {
    q->frontSentinel = malloc(sizeof(struct dlink));
    assert(q->frontSentinel != 0);
    q->backSentinel = malloc(sizeof(struct dlink));
    assert(q->backSentinel);
    q->frontSentinel->next = q->backSentinel;
    q->backSentinel->prev = q->frontSentinel;
    q->size = 0;
}
• Consider a deque, with two sentinels:
  – Pointer to front sentinel: frontSent
  – Pointer to back sentinel: backSent

• Add to front and add to back are now special cases of more general “add before” operation

This is similar to most standard library Deque implementations (Java LinkedList)
Adding to the LL Deque

```c
void addBackListDeque(struct ListDeque *q, TYPE val) {
    _addBefore(q->lastLink, val);
}

void addFrontListDeque(struct ListDeque *q, TYPE val) {
    _addBefore(q->firstLink->next, val);
}
```
void removeFirstListDeque(struct ListDeque *q) {
    assert(!isEmptyListDeque(q));

    _removeLink(q->firstLink->next);
}

void removeLastListDeque(struct ListDeque *q) {
    assert(!isEmptyListDeque(q));

    _removeLink (q->lastLink->prev);
}
Deque: DynArray vs. Linked List

- Remember: Finite length undo
- Which would you use?
- Which would support this kind of history operation?
Your Turn...

Worksheet #19: `_addBefore, _removeLink`  

<table>
<thead>
<tr>
<th></th>
<th>Dynamic Array Deque</th>
<th>Linked List Deque</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>best, ave, worst</td>
<td>best, ave, worst</td>
</tr>
<tr>
<td>addLast</td>
<td>O(1), O(1+), O(N)</td>
<td>O(1), O(1), O(1)</td>
</tr>
<tr>
<td>removeLast</td>
<td>O(1), O(1), O(1)</td>
<td>O(1), O(1), O(1)</td>
</tr>
<tr>
<td>addFirst</td>
<td>O(1), O(1+), O(N)</td>
<td>O(1), O(1), O(1)</td>
</tr>
<tr>
<td>removeFirst</td>
<td>O(1), O(1), O(1)</td>
<td>O(1), O(1), O(1)</td>
</tr>
</tbody>
</table>