The message board as a linked list

What if: instead of using an array, we added a pointer to each struct that pointed to the next struct in a sequence.

It can be done.

Then removing an item from the message board would just require changing the pointers to bypass the removed item and freeing the memory used by that item.

Linked Lists

Probably what Vectors use so you can call add, remove, push back. Also true for Python lists

Data structure where objects arranged in linear order. Order established by pointer to next object instead of by index

Today: Singly-linked lists
    Object points to next object, but not to previous object.
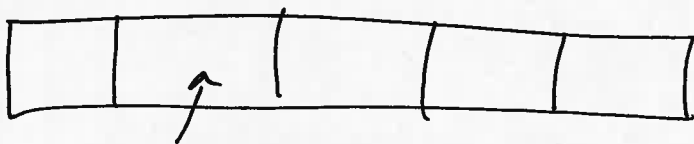   Using or message board, each item on the

Filling a gap in an array

b f 10	b w 5	c w 2	f f 500	f f 1000

messageBoard
array of structs

read in bicycle for sale, $4

Matches with 2ND item in array

want to remove this one

b f 10		c w 2	f f 500	f f 1000
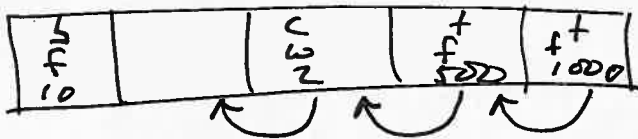
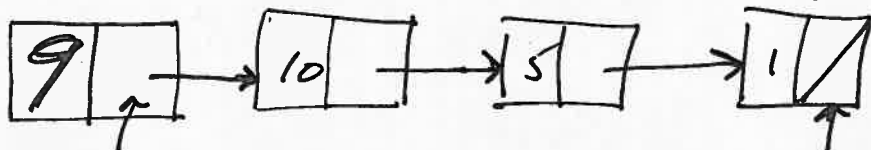3 operations
Costly for large data sets

the reason we have to shift is the array is a contiguous block of memory and its the index of each item that establishes order of the array.

We also know that each item in the array has an address. And, we can access a variables address. And, we can create a pointer to reference an address.

Show array example w/ addresses, and pointer example

board would include a pointer to next item on the board, but not to previous item on the board.

Linked List with one property, or key

```
┌───┬──┐    ┌────┬──┐    ┌───┬──┐    ┌───┬──┐
│ 9 │  ┼──→ │ 10 │  ┼──→ │ 5 │  ┼──→ │ 1 │╱ │
└───┴──┘    └────┴──┘    └───┴──┘    └───┴──┘
      ↑                                  ↑
    this                          the final pointer
    contains                      is null, indicating
    address of next object        the end of the
                                  list
```
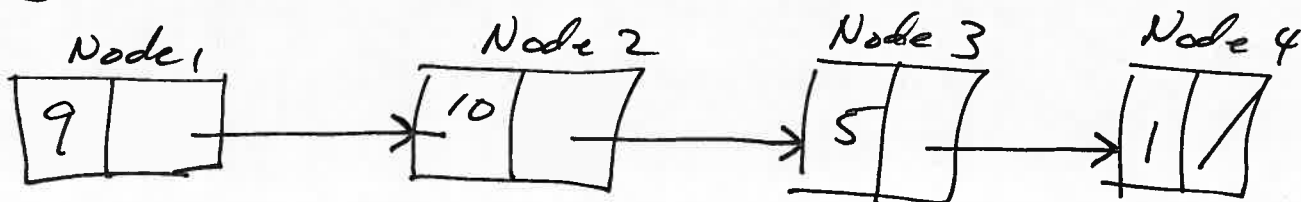
If that block diagram were a struct, it would look something like:

```
struct node {
    int x;
    node *next;    //pointer to object of same
                   //type
}
```

```
┌───┬──┐    ┌────┬──┐    ┌───┬──┐    ┌───┬──┐
│ 9 │  ┼──→ │ 10 │  ┼──→ │ 5 │  ┼──→ │ 1 │╱ │
└───┴──┘    └────┴──┘    └───┴──┘    └───┴──┘
 Node 1      Node 2      Node 3      Node 4
```

Node 1 next is node 2
Node 2 next is node 3
Node 3 next is node 4
Node 4 next is NULL

The object in this example is a struct w/ an integer and a pointer. But, the object could be anything. It could be a struct of your message board items. Or, it could be a class. You could create a linked list of Battleships. You would just need to modify the message board or the Battleship to include that next pointer.

Memory Allocation

With arrays, we allocate memory all at once. Here, we will allocate it dynamically as we add new nodes to our linked list.

Today we're going to discuss dynamic memory allocation using the new keyword, and then how we free this memory.