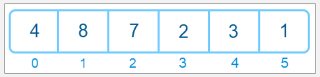
## Data Structures

There are two main kinds of data structures –

- *contiguous structures* occupy consecutive memory locations
  - e.g. arrays
- *linked structures* consist of separate chunks of memory connected by references or pointers
  - e.g. linked lists, many implementations of trees, graphs

## Array Refresher

array concepts



– insert/remove element involves shifting elements (order-preserving)
– dynamic array requires resizing when full

```
/**
 * Add the element at the specified index in the array, preserving the order
 * of the remaining elements.
 *
 * @param array
 *          the array
 * @param elt
 *          element to add
 * @param index
 *          index where to add the element
 */
public static void insert ( int[] array, int elt, int index ) {

}
```

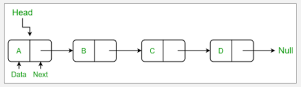
strategy – use examples!
- draw before and after pictures
- identify what changes
- get convenient references for those things
- update the values

be sure to consider several cases to make sure your solution works in general

be sure to consider typical special cases such as the first and last things, empty or one-element array

## Linked List Refresher

linked list concepts
– singly-linked list



– insert/remove node involves re-linking rather than shifting or changing elements

```
/**
 * Remove the specified node from the list.
 *
 * @param head
 *          head of the list
 * @param todel
 *          node to remove
 * @return the head of the list after the removal
 */
public static ListNode remove ( ListNode head, ListNode todel ) {
  return null;
}
```

```
public class ListNode {

  // what goes here?

}
```

strategy – use examples!
- draw before and after pictures
- identify what changes
- get convenient references for those things
- update the values

be sure to consider several cases to make sure your solution works in general

be sure to consider typical special cases such as the first and last things, empty or one-element list, `null` values