

## **Tips and Common Points of Confusion**

# How to Get Started

---

Stuck on how to even get started on LinkedListDeque and/or ArrayDeque? One great first step is implementing SLList and/or AList. Starter code can be found in:

- SLList: [lec4/DIY](#)
- AList: lec6/DIY

To make this more time efficient:

- Work with a friend or two or three.
- See solutions (available on github) (for when you get stuck or just want to compare).

## How to Get Started (Part 2)

---

Try implementing just the empty Deque, and compare your data structure to the idea developed in class.

- e.g. your `LinkedListDeque()` constructor should probably build either:
  - The top figure [from this slide](#) (if you're doing two sentinel nodes).
  - The top figure [from this slide](#) (if you're going circular).

Compare the output of your code using the visualizer.

- You can use the visualizer from inside IntelliJ. See the [61B Plugin](#) guide.

## Other General Tips

---

Take things a little at a time.

- Writing tons of code all at once is going to lead to misery and only misery.
- If you wrote too much stuff and feel overwhelmed, comment out whatever is unnecessary.

If your first try goes badly, don't be afraid to scrap your code and start over.

- The amount of code for each class isn't actually that much (my solution is about 130 lines for each .java file, including all comments and whitespace).

Consider not doing resizing at all until you know your code works without it.

- Resizing is a performance optimization (and is required for full credit).

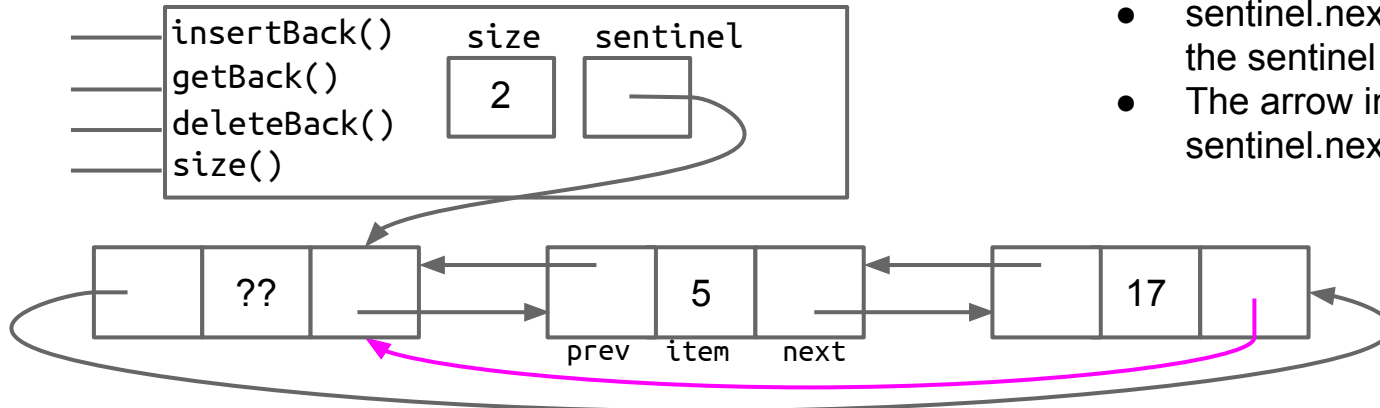
# Common Point of Confusion

Any time I drew an arrow in class that pointed at an object, the pointer was to the ENTIRE object, not a particular field of an object.

- For example, the bits in the “sentinel” box do not point at the “next” field of the leftmost node. They instead point to the ENTIRE node.
- (in fact it is impossible for a reference to point to the fields of an object in Java)

Examples:

- `sentinel.next.next` is the node with `item=17`.
- `sentinel.next.next.next` is the sentinel node.
- The arrow in purple is `sentinel.next.next.next`

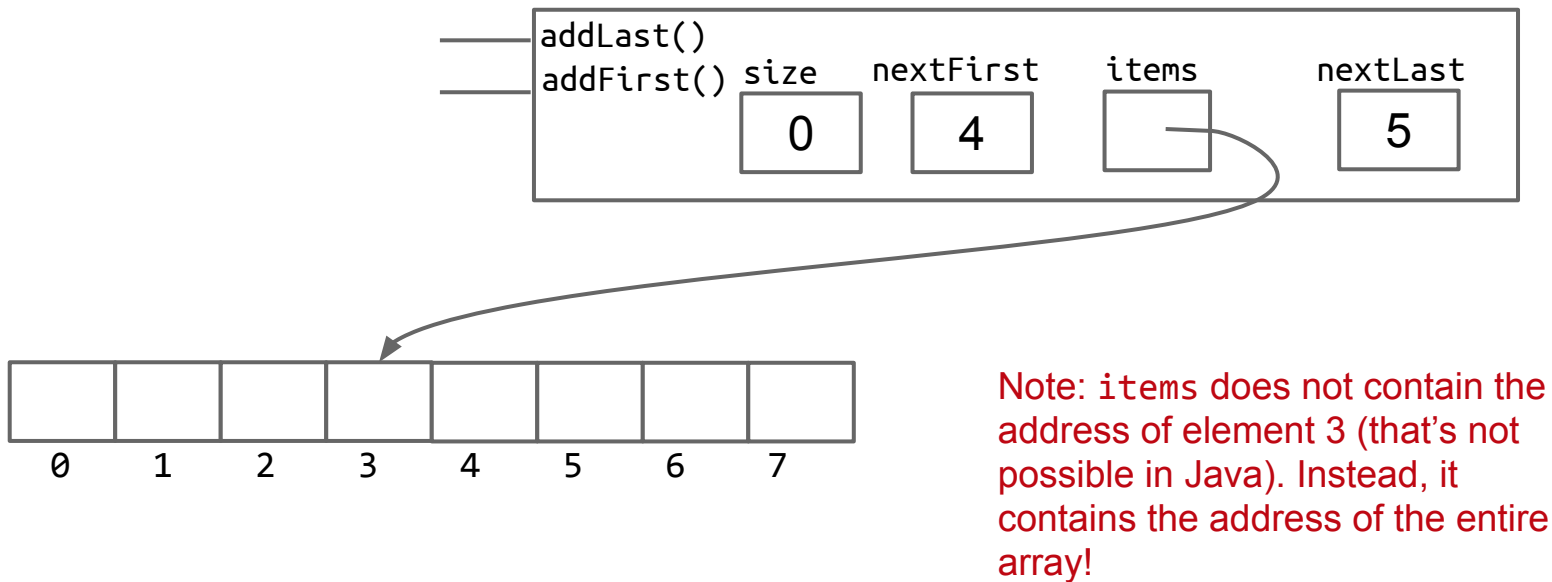


## Circular ArrayDeque Demo

# ArrayDeque Tips

---

Starting from an empty ArrayDeque:

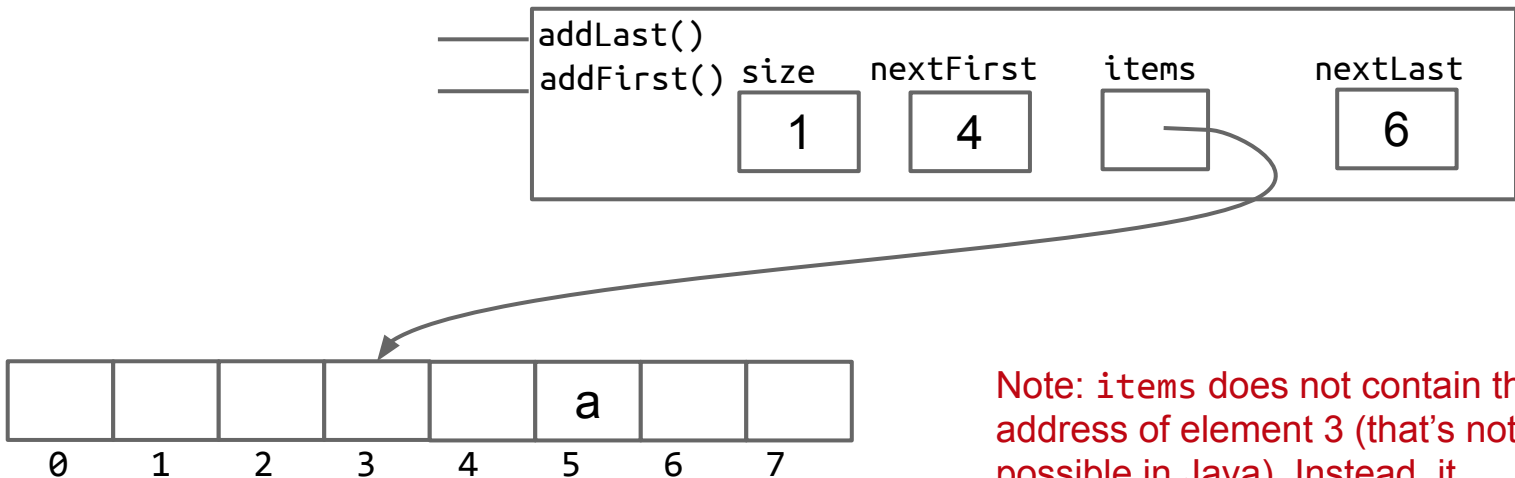


You don't have to start your array at 4 or 5. I just picked these arbitrarily.

# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`



Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

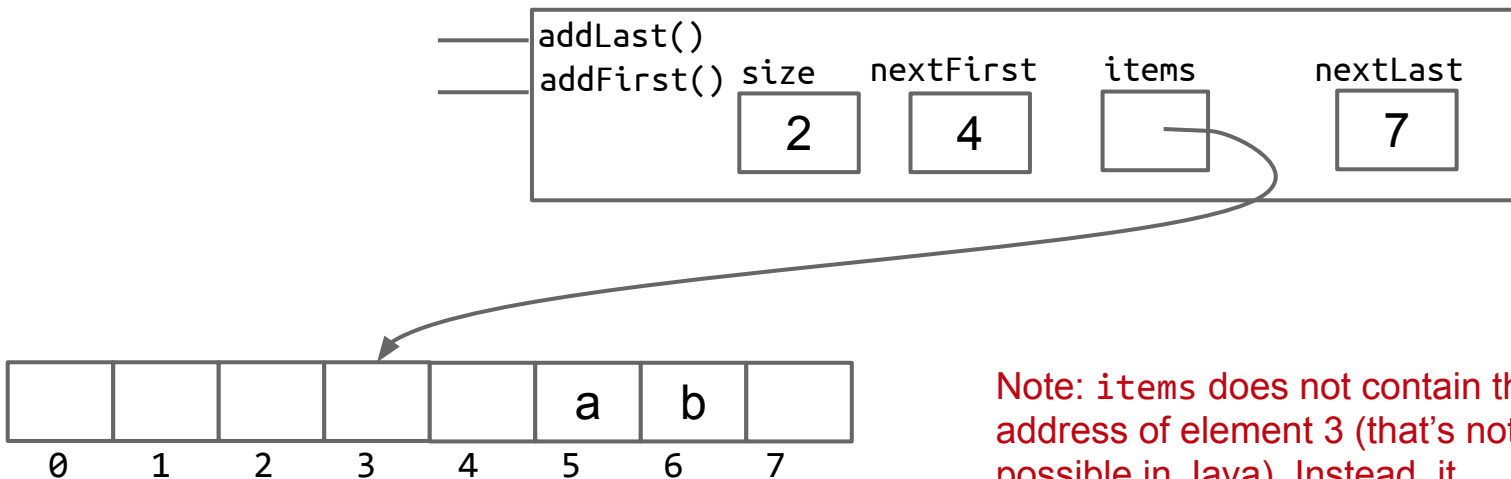
You don't have to start your array at 4 or 5. I just picked these arbitrarily.



# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`
- `addLast("b")`



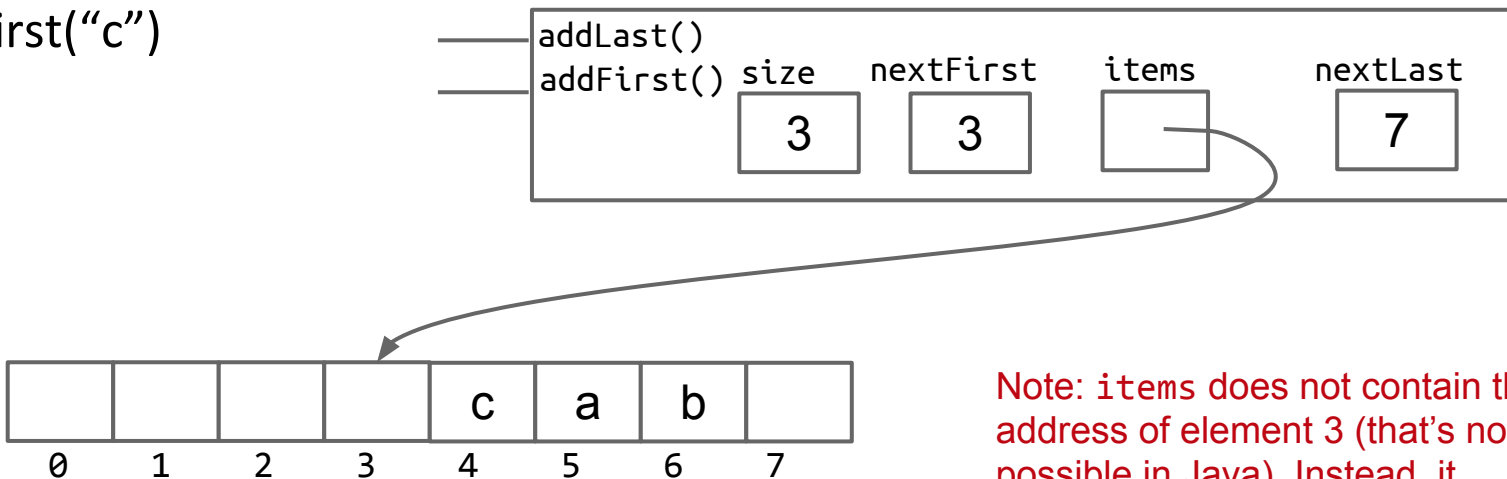
Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

You don't have to start your array at 4 or 5. I just picked these arbitrarily.

# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`
- `addLast("b")`
- `addFirst("c")`



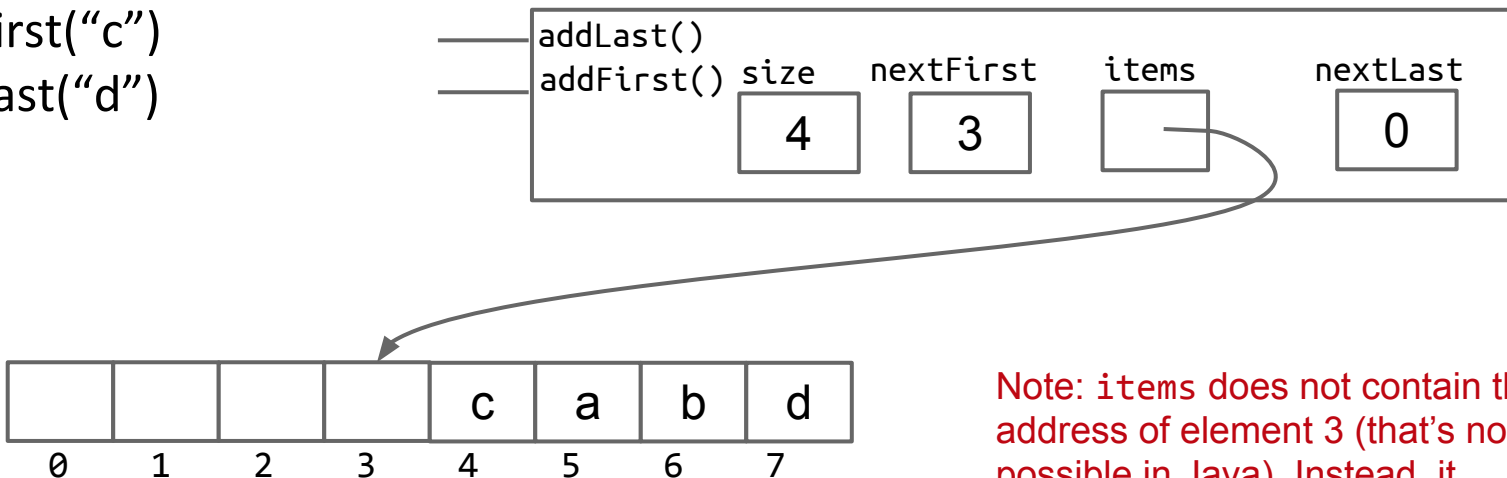
Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

You don't have to start your array at 4 or 5. I just picked these arbitrarily.

# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`
- `addLast("b")`
- `addFirst("c")`
- `addLast("d")`



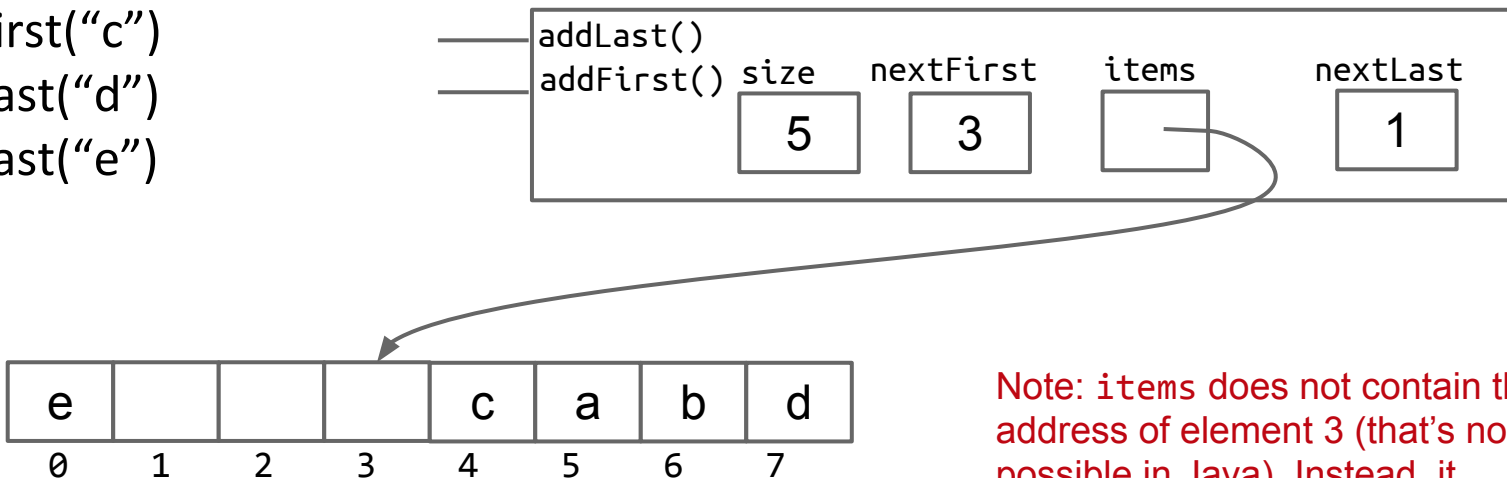
Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

You don't have to start your array at 4 or 5. I just picked these arbitrarily.

# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`
- `addLast("b")`
- `addFirst("c")`
- `addLast("d")`
- `addLast("e")`



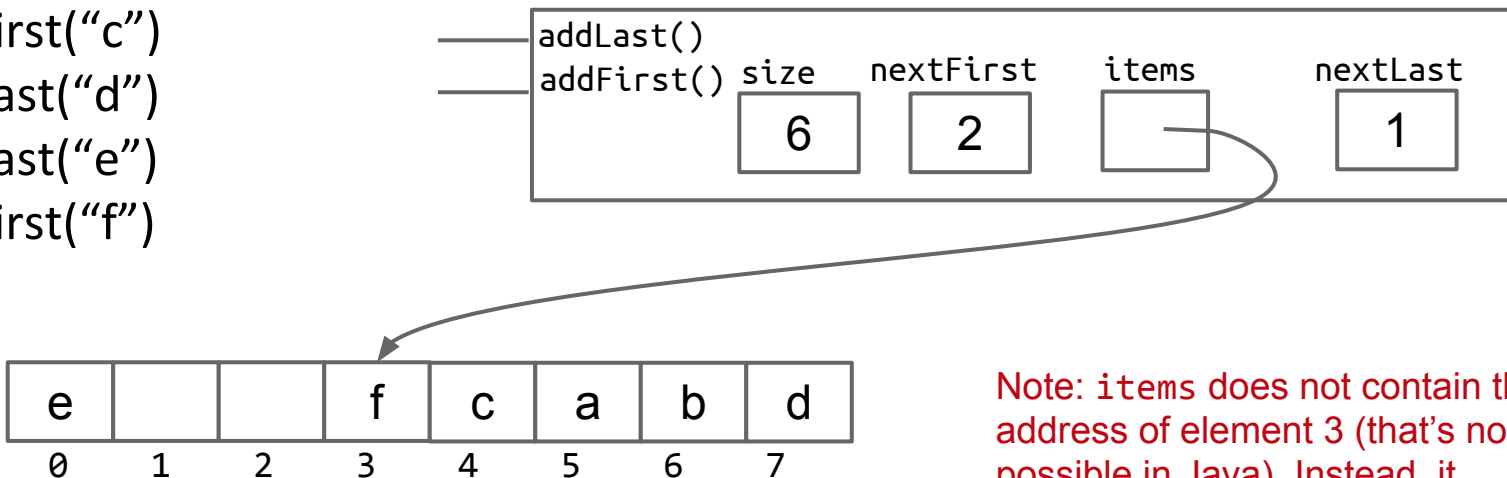
Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

You don't have to start your array at 4 or 5. I just picked these arbitrarily.

# ArrayDeque Tips

Starting from an empty ArrayDeque:

- `addLast("a")`
- `addLast("b")`
- `addFirst("c")`
- `addLast("d")`
- `addLast("e")`
- `addFirst("f")`



Note: `items` does not contain the address of element 3 (that's not possible in Java). Instead, it contains the address of the entire array!

You don't have to start your array at 4 or 5. I just picked these arbitrarily.