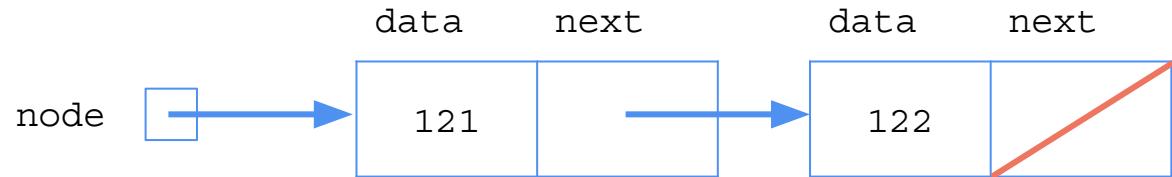


Lesson 7

Linked Lists

Linked Nodes So Far

- Small sequences of nodes connected together
- Using the `ListNode` class
- Traversals over these sequences with `while` loops



Introducing the LinkedList!

- New collection named LinkedList

Introducing the LinkedList!

- New collection named `LinkedList`
- Same kinds of methods as the `ArrayList`
 - `add`, `add`, `get`, `indexOf`, `remove`, `size`, `toString`

Introducing the LinkedList!

- New collection named LinkedList
- Same kinds of methods as the ArrayList
 - add, add, get, indexOf, remove, size, toString
- Implemented with chain of linked nodes

Introducing the LinkedList!

- New collection named `LinkedList`
- Same kinds of methods as the `ArrayList`
 - `add`, `add`, `get`, `indexOf`, `remove`, `size`, `toString`
- Implemented with chain of linked nodes
 - Keeps reference to its `front` as a field
 - `null` is the end of the list
 - If `front` is `null`, list is empty

Introducing the `LinkedList`!

- Implemented with chain of linked nodes
 - Keeps reference to its `front` as a field
 - `null` is the end of the list; if `front` is `null`, list is empty

`LinkedList`

`front`

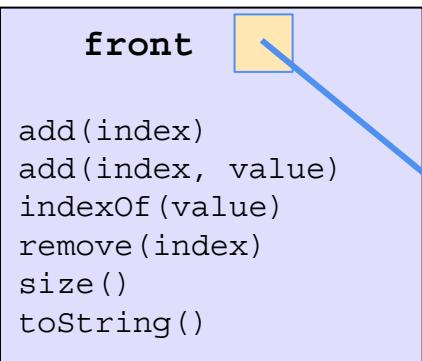


```
add(index)
add(index, value)
indexOf(value)
remove(index)
size()
toString()
```

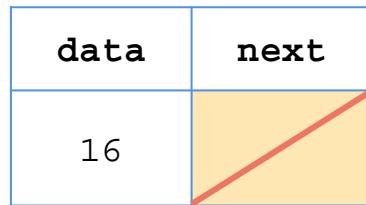
Introducing the `LinkedList`!

- Implemented with chain of linked nodes
 - Keeps reference to its `front` as a field
 - `null` is the end of the list; if `front` is `null`, list is empty

`LinkedList`

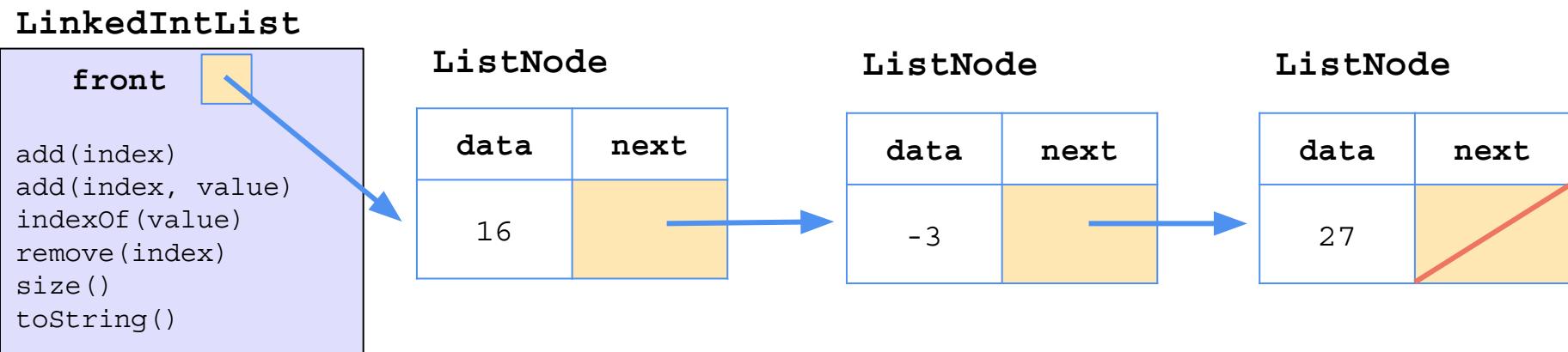


`ListNode`



Introducing the `LinkedList`!

- Implemented with chain of linked nodes
 - Keeps reference to its `front` as a field
 - `null` is the end of the list; if `front` is `null`, list is empty



printList Revisited

```
public class LinkedList {  
  
    private ListNode front;  
    private int size;  
    ...  
  
    public void printList() {  
        if (front == null) {  
            System.out.println("[]");  
        } else {  
            System.out.print("[ " + front.data);  
            front = front.next;  
            while (front != null) {  
                System.out.print(", " + front.data);  
                front = front.next;  
            }  
            System.out.println("]");  
        }  
    }  
}
```

printList Revisited

```
public class LinkedList {  
  
    private ListNode front;  
    private int size;  
    ...  
  
    public void printList() {  
        if (front == null) {  
            System.out.println("[]");  
        } else {  
            System.out.print("[ " + front.data);  
            front = front.next;  
            while (front != null) {  
                System.out.print(", " + front.data);  
                front = front.next;  
            }  
            System.out.println("]");  
        }  
    }  
}
```

Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
    l1.printList();  
    l1.printList();  
}
```

printList Revisited

```
public class LinkedList {  
  
    private ListNode front;  
    private int size;  
    ...  
  
    public void printList() {  
        if (front == null) {  
            System.out.println("[]");  
        } else {  
            System.out.print("[ " + front.data);  
            front = front.next;  
            while (front != null) {  
                System.out.print(", " + front.data);  
                front = front.next;  
            }  
            System.out.println(" ]");  
        }  
    }  
}
```

Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
  
    l1.printList();  
    l1.printList();  
}
```

**What will be the output of
this program?**

printList Revisited

```
public class LinkedList {  
  
    private ListNode front;  
    private int size;  
    ...  
  
    public void printList() {  
        if (front == null) {  
            System.out.println("[]");  
        } else {  
            System.out.print("[ " + front.data);  
            front = front.next;  
            while (front != null) {  
                System.out.print(", " + front.data);  
                front = front.next;  
            }  
            System.out.println(" ]");  
        }  
    }  
}
```

Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
    l1.printList(); // [16, -3, 27]  
    l1.printList(); // []  
}
```

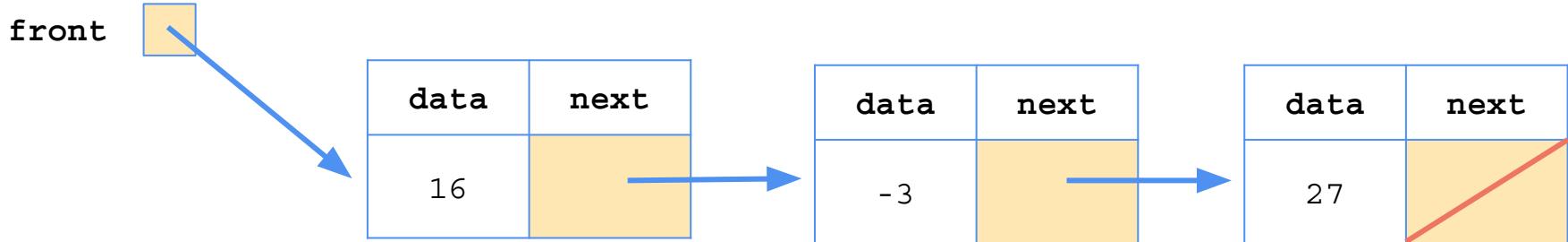
What will be the output of this program?

We're losing the `front` of our list!

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```



Client Code:

```

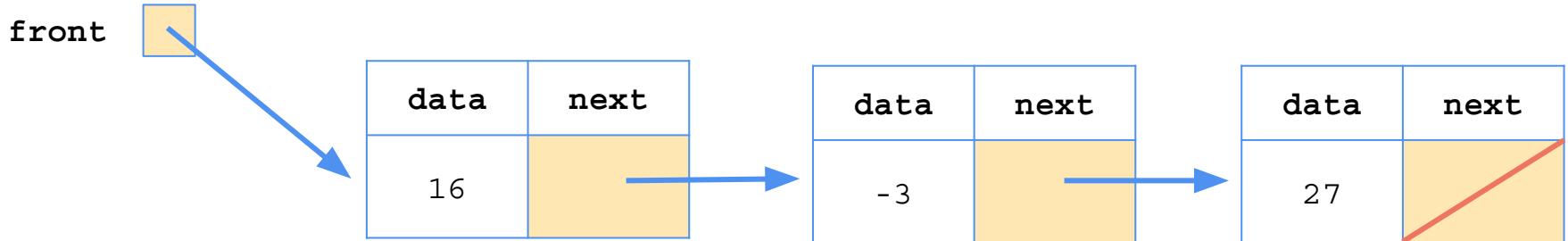
public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList();
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```



Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

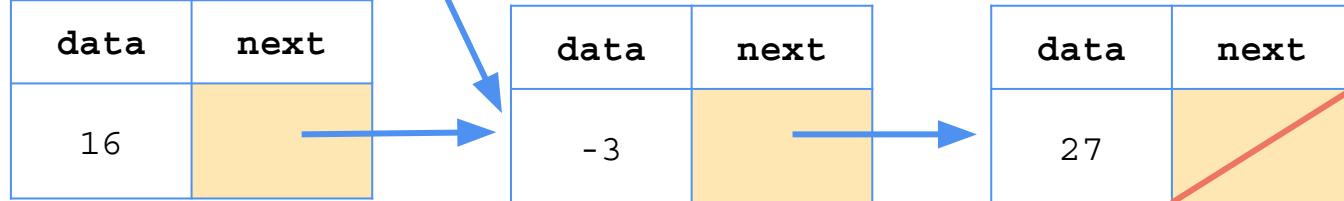
```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```

front



Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```

front

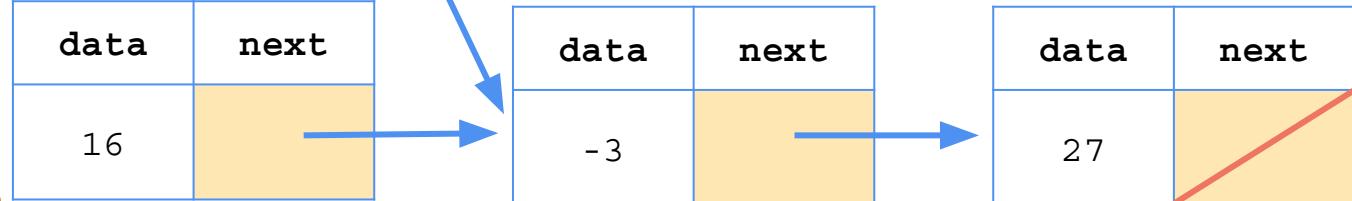


Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

```



```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```

front

A blue arrow points from the variable 'front' to the first node of a linked list. The 'front' variable is highlighted with a yellow square.

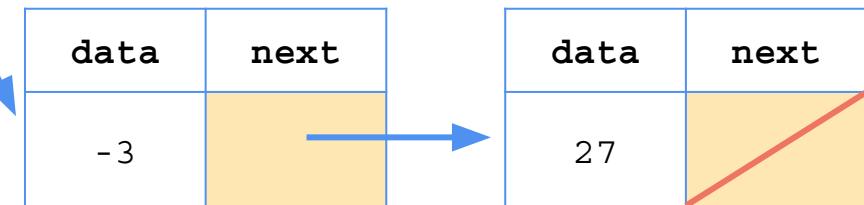
Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

```

}



```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```

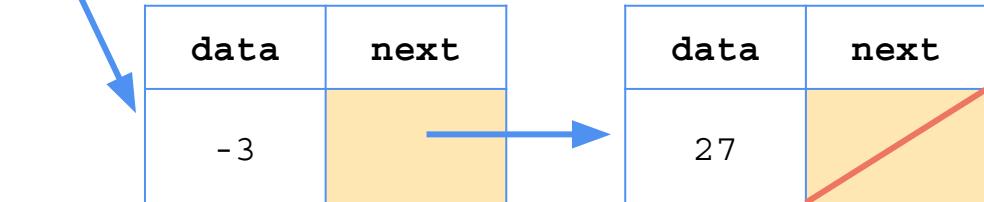
front

Client Code:

```

public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3
    l1.printList();
}

```



```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

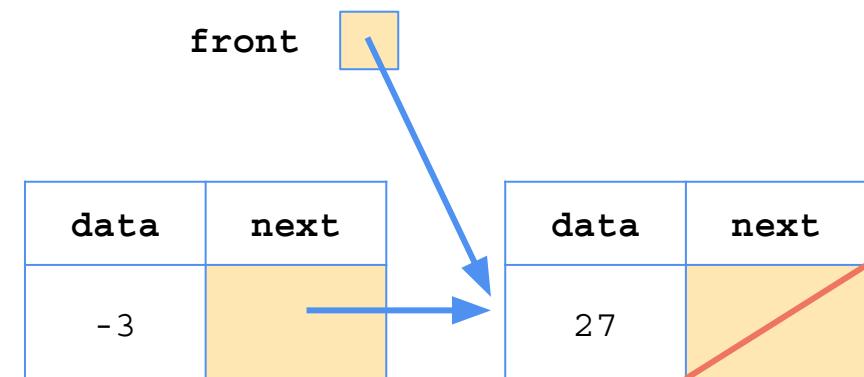
```

Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3
    l1.printList();
}

```



```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```



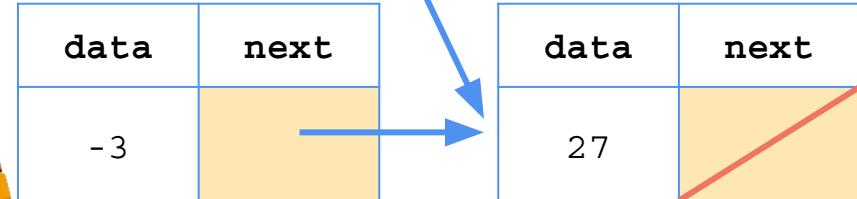
Client Code:

```

public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3
    l1.printList();
}

```

`front`



```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("]");
    }
}

```

Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3
    l1.printList();
}

```

`front`



data	next
27	

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        front = front.next;
        while (front != null) {
            System.out.print(", " + front.data);
            front = front.next;
        }
        System.out.println("] ");
    }
}

```

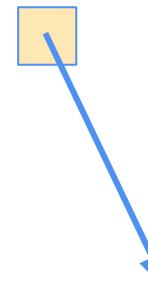
Client Code:

```

public static void main(String[] args) {
    LinkedIntList l1 = new LinkedIntList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3, 27]
    l1.printList();
}

```

`front`



<code>data</code>	<code>next</code>
27	

```
public void printList() {  
    if (front == null) {  
        System.out.println("[]");  
    } else {  
        System.out.print("[ " + front.data);  
        front = front.next;  
        while (front != null) {  
            System.out.print(", " + front.data);  
            front = front.next;  
        }  
        System.out.println("]");  
    }  
}
```

Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
    l1.printList(); // [16, -3, 27]  
    l1.printList();  
}
```

front



```
public void printList() {  
    if (front == null) {  
        System.out.println("[]");  
    } else {  
        System.out.print("[ " + front.data);  
        front = front.next;  
        while (front != null) {  
            System.out.print(", " + front.data);  
            front = front.next;  
        }  
        System.out.println("]");  
    }  
}
```

Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
    l1.printList(); // [16, -3, 27]  
    l1.printList(); // []  
}
```

front



```
public void printList() {  
    if (front == null) {  
        System.out.println("[]");  
    } else {  
        System.out.print("[ " + front.data);  
        front = front.next;  
        while (front != null) {  
            System.out.print(", " + front.data);  
            front = front.next;  
        }  
        System.out.println("]");  
    }  
}
```

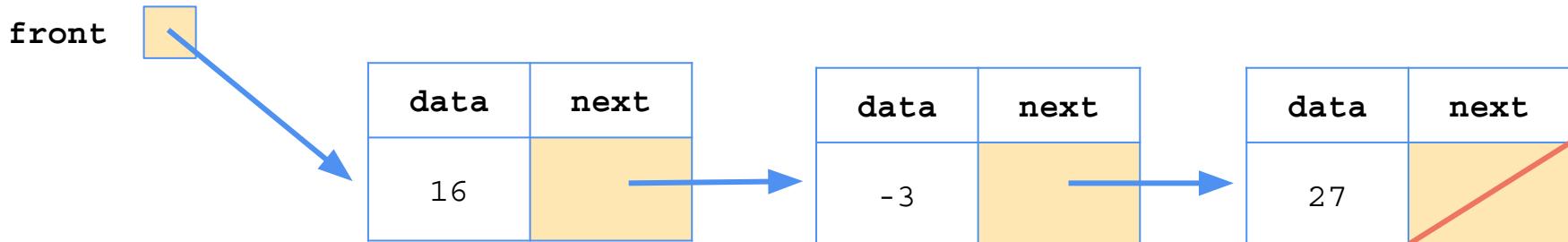
Client Code:

```
public static void main(String[] args) {  
    LinkedList l1 = new LinkedList(  
        new int[]{16, -3, 27});  
    l1.printList();  
    l1.printList();  
}
```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

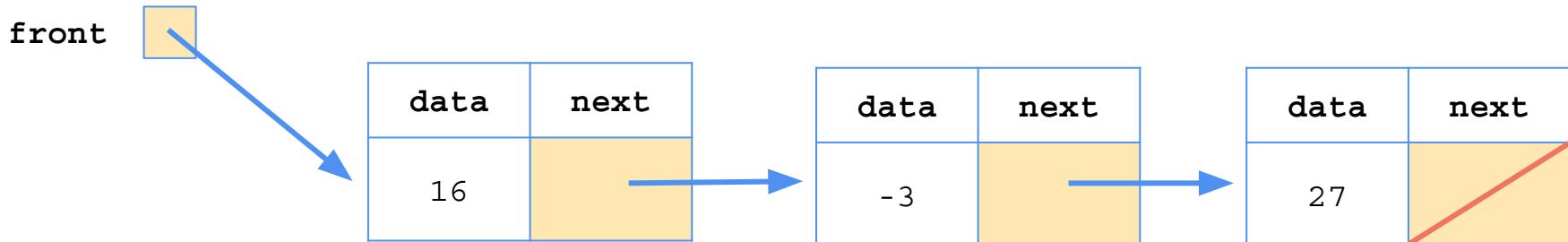
public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList();
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

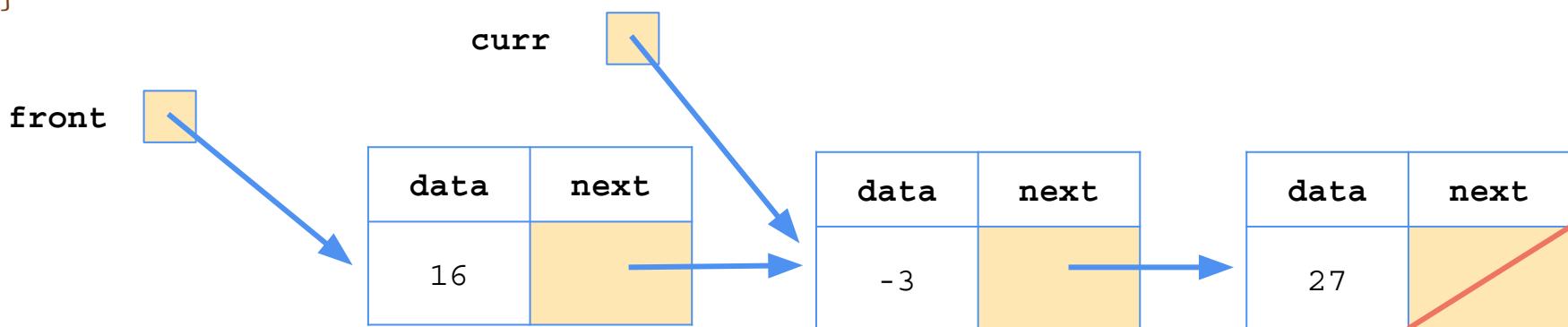
public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

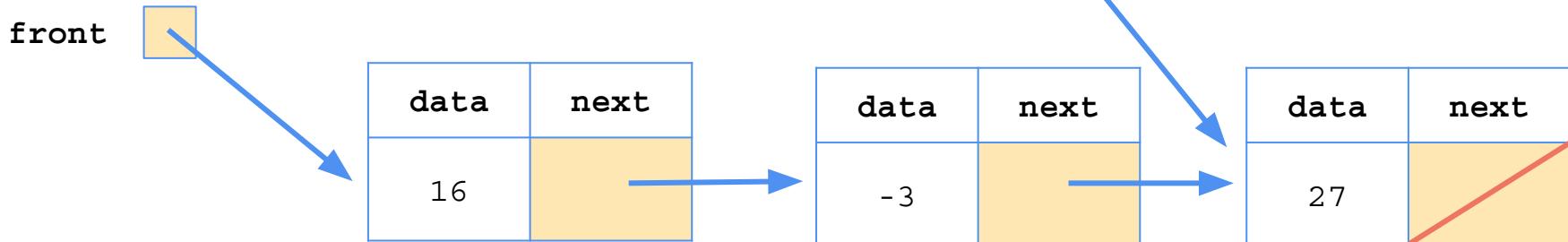
public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

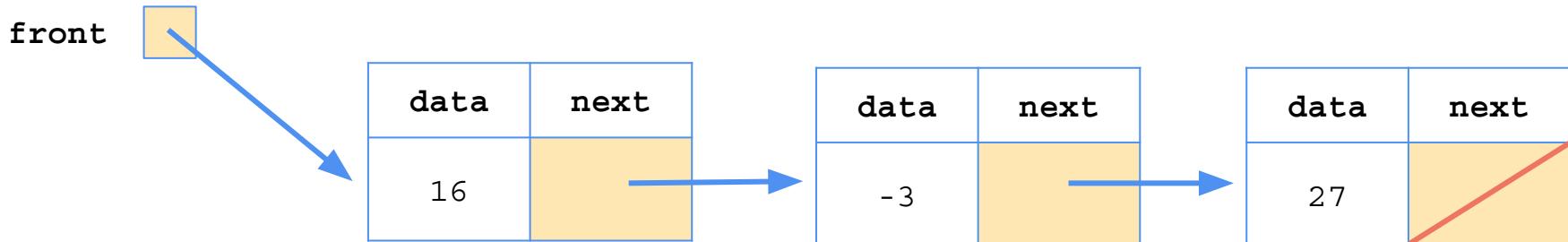
public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

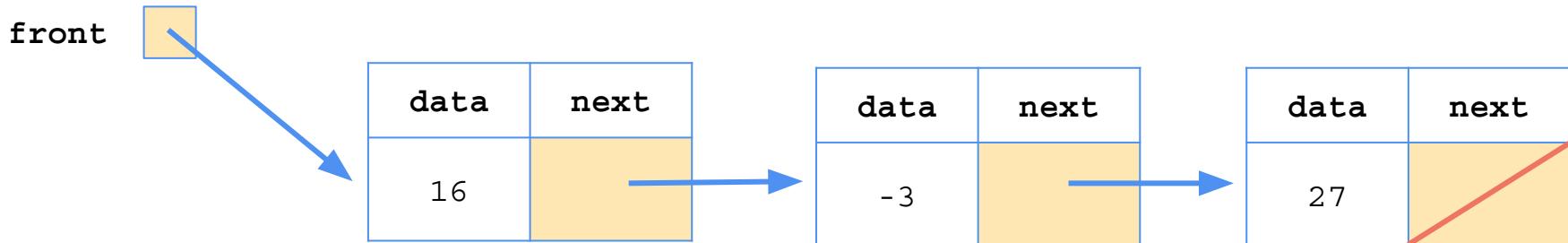
public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3, 27]
    l1.printList();
}

```

```

public void printList() {
    if (front == null) {
        System.out.println("[]");
    } else {
        System.out.print("[ " + front.data);
        ListNode curr = front.next;
        while (curr != null) {
            System.out.print(", " + curr.data);
            curr = curr.next;
        }
        System.out.println(" ]");
    }
}

```



Client Code:

```

public static void main(String[] args) {
    LinkedList l1 = new LinkedList(
        new int[]{16, -3, 27});
    l1.printList(); // [16, -3, 27]
    l1.printList(); // [16, -3, 27]
}

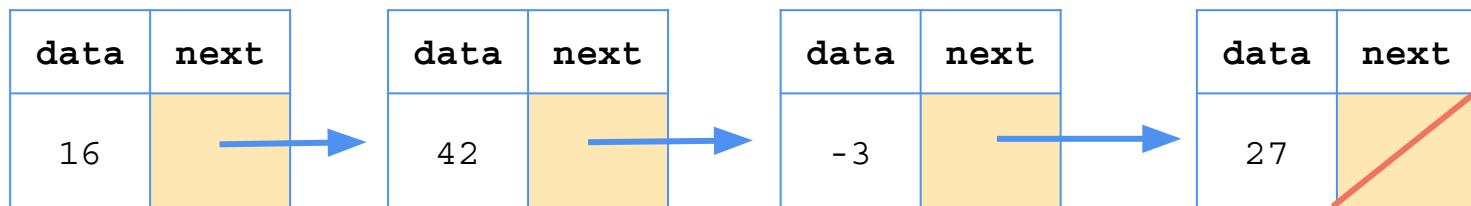
```

remove (int value) Visualization

⚠️ Without “Stopping One Early” ⚠️

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```

Client calls `list.remove(-3)`



⚠️ Without “Stopping One Early” ⚠️

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}  
}front
```

A red arrow points to the line `ListNode curr = front;`. Below the code, there are four nodes represented as tables:

data	next
16	

data	next
42	

data	next
-3	

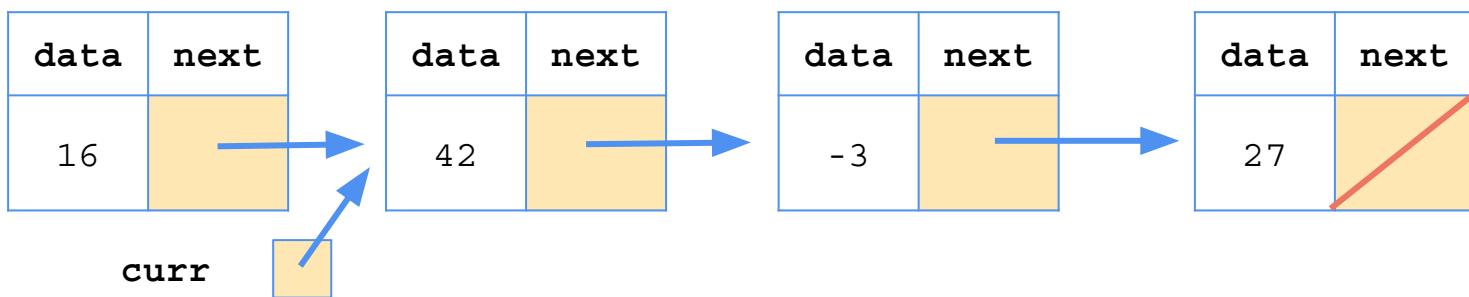
data	next
27	

The `curr` pointer is shown pointing to the first node (data 16). The `front` pointer is shown pointing to the second node (data 42).

Client calls `list.remove(-3)`

⚠️ Without “Stopping One Early” ⚠️

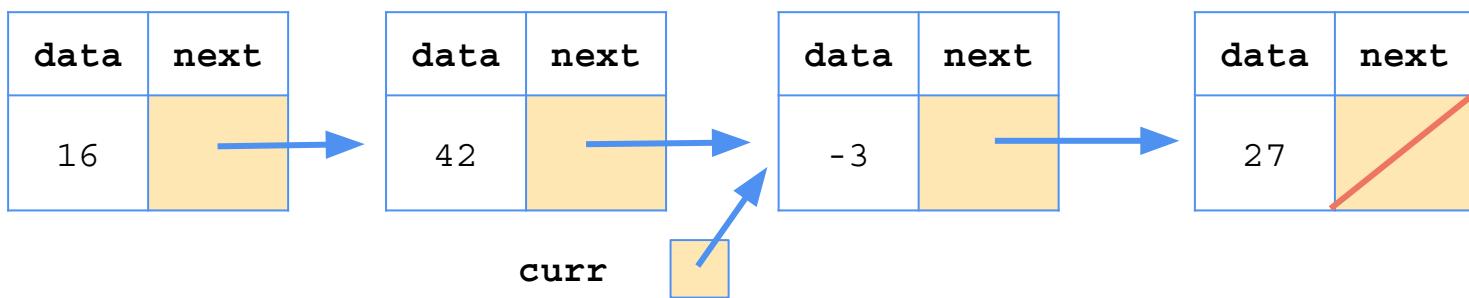
```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```



Client calls `list.remove(-3)`

⚠️ Without “Stopping One Early” ⚠️

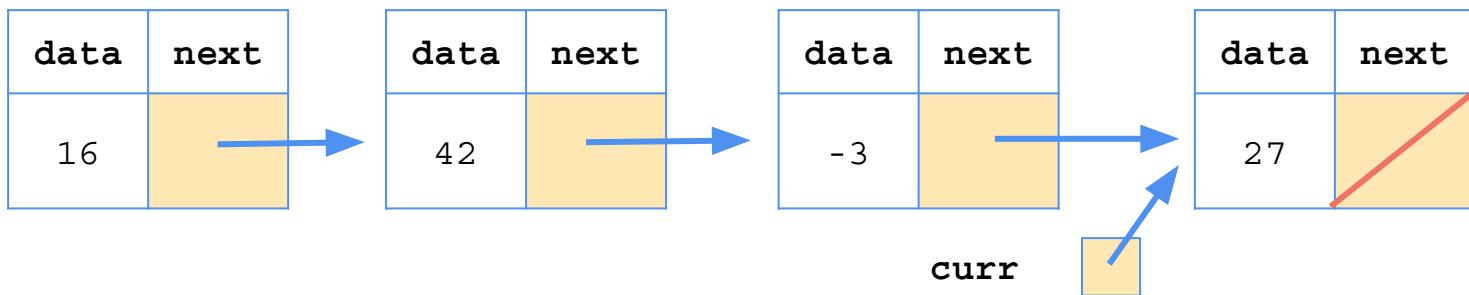
```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```



Client calls `list.remove(-3)`

⚠️ Without “Stopping One Early” ⚠️

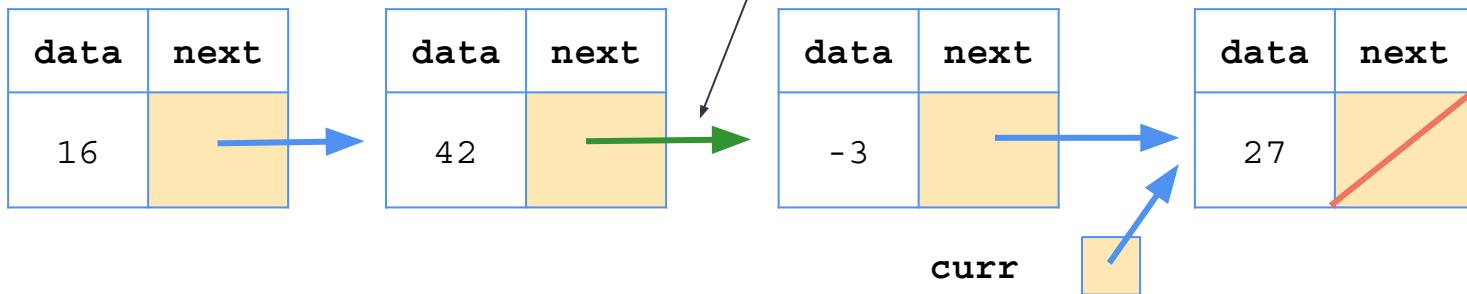
```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```



Client calls `list.remove(-3)`

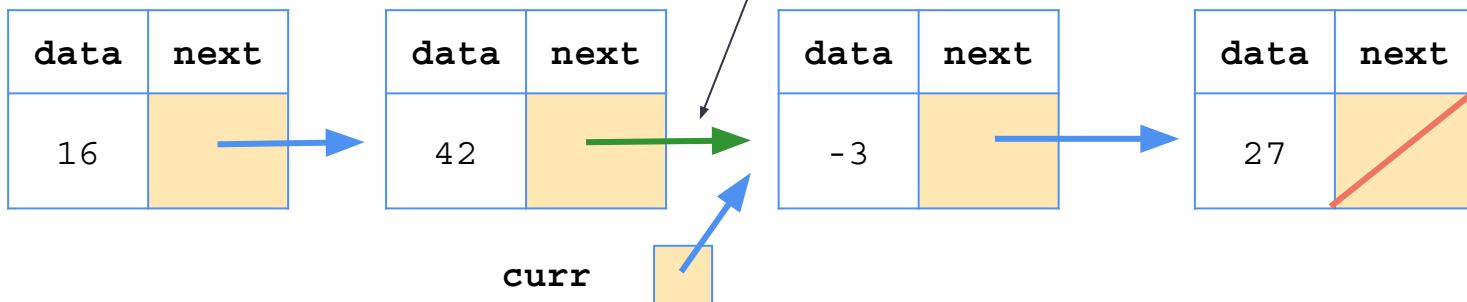
⚠️ Without “Stopping One Early” ⚠️

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```



⚠️ Without “Stopping One Early” ⚠️

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr != null && curr.data != value) {  
            curr = curr.next;  
        }  
        if (curr != null) {  
            curr = curr.next;  
            size--;  
        }  
    }  
}  
}front
```



Client calls `list.remove(-3)`

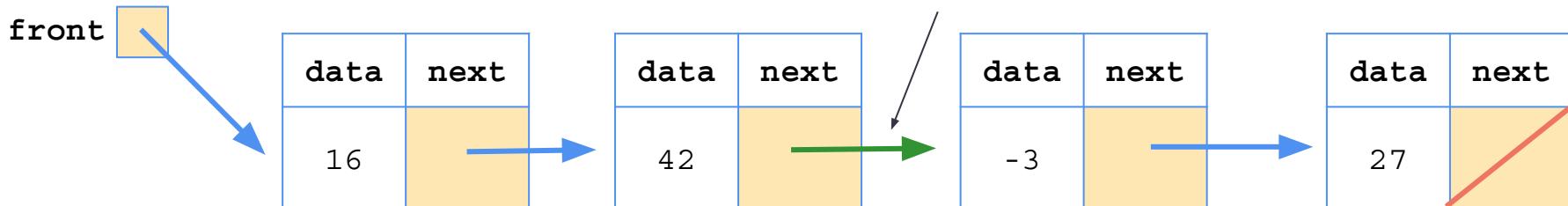
This is the reference we need to change!

With “Stopping One Early”

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr.next != null && curr.next.data != value) {  
            curr = curr.next;  
        }  
        if (curr.next != null) {  
            curr.next = curr.next.next;  
            size--;  
        }  
    }  
}
```

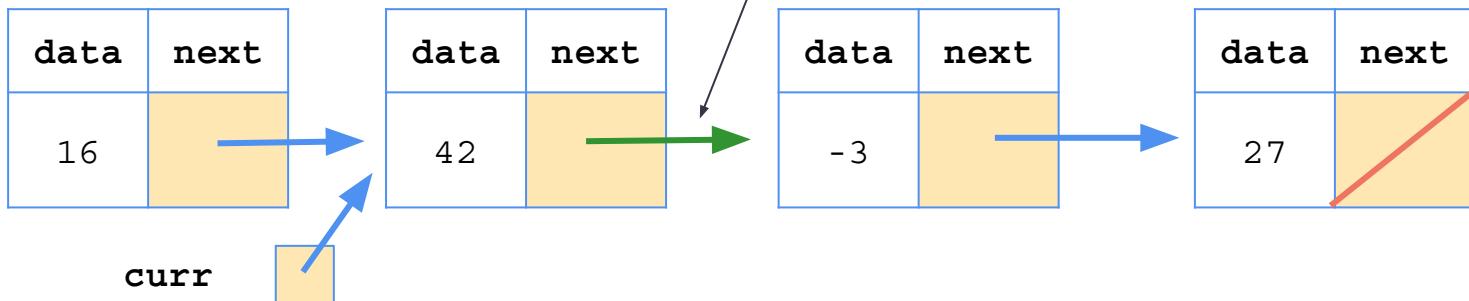
Client calls `list.remove(-3)`

This is the reference we need to change!



With “Stopping One Early”

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr.next != null && curr.next.data != value) {  
            curr = curr.next;  
        }  
        if (curr.next != null) {  
            curr.next = curr.next.next;  
            size--;  
        }  
    }  
}  
front
```

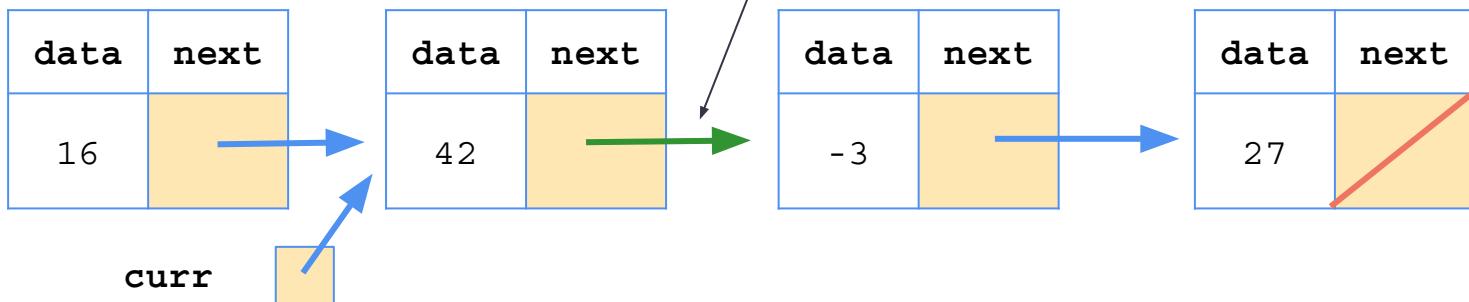


Client calls `list.remove(-3)`

This is the reference we need to change!

With “Stopping One Early”

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr.next != null && curr.next.data != value) {  
            curr = curr.next;  
        }  
        if (curr.next != null) {  
            curr.next = curr.next.next;  
            size--;  
        }  
    }  
}  
front
```

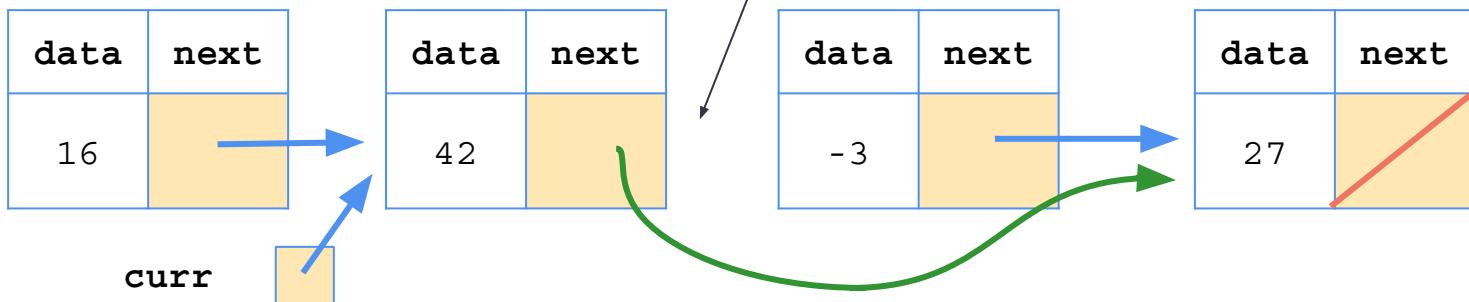


Client calls `list.remove(-3)`

This is the reference we need to change!

With “Stopping One Early”

```
public void remove(int value) {  
    if (front != null) {  
        // ... front case  
    } else {  
        ListNode curr = front;  
        while (curr.next != null && curr.next.data != value) {  
            curr = curr.next;  
        }  
        if (curr.next != null) {  
            curr.next = curr.next.next;  
            size--;  
        }  
    }  
}  
} front
```



Client calls `list.remove(-3)`

This is the reference we need to change!

Changing a list

- There are only two ways to change a linked list:
 - Change the value of `front` (modify the front of the list)
 - Change the value of `<node>.next` (modify middle or end of list to point somewhere else)
- Implications:
 - To add in the middle, need a reference to the *previous* node
 - Front is often a special case