Review Question

• What does the following print?
  ```java
  public static void main(String[] args) {
    String a = "AAA";
    String b = "BBB";
    String x = a;
    String y = b;
    String temp = x;
    x = y;
    y = temp;
    System.out.println(a + " " + b);
    System.out.println(x + " " + y);
  }
  ```

References to same type

• What would happen if we had a class that declared one of its own type as a field?
  ```java
  public class Strange {
    private String name;
    private Strange other;
  }
  ```

  – Will this compile?
  • If so, what is the behavior of the `other` field? What can it do?
  • If not, why not? What is the error and the reasoning behind it?

Linked data structures

• All of the collections we will use and implement in this course use one of the following two underlying data structures:
  
  – an array of all elements
    • ArrayList, Stack, HashSet, HashMap

      | 42 | -3 | 17 | 9 |

  – a set of linked objects, each storing one element, and one or more reference(s) to other element(s)
    • LinkedList, TreeSet, TreeMap

      front → 42 → -3 → 17 → 9 → null

Linked Lists

• A linked list is a type of list

• But instead of a contiguous block of memory, like this:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

• Linked list elements are scattered throughout memory:

  | 8 | 9 | 7 | 5 | 6 |

• But now the elements are unordered. How do linked lists keep track of everything?
Linked Lists

• Each element must have a reference to the next element:

- 8
- 9
- 7
- 5
- 6

• Now, so long as we keep track of the first element (the front), we can keep track of all the elements.

Linked node problem 1

• What set of statements turns this picture:

```
list.next.next = new ListNode(30);
```

• Into this?

```
list = new ListNode(30, list);
```

Linked node problem 2