Point $p = \text{new } \text{Point}(1,2);$

Point $q = p;$
Road Map

**CS Concepts**
- Client/Implementer
- Efficiency
- Recursion
- Regular Expressions
- Grammars
- Sorting
- Backtracking
- Hashing
- Huffman Compression

**Java Language**
- Exceptions
- Interfaces
- References
- Comparable
- Generics
- Inheritance/Polymorphism
- Abstract Classes

**Data Structures**
- Lists
- Stacks
- Queues
- Sets
- Maps
- Priority Queues

**Java Collections**
- Arrays
- ArrayList
- LinkedList
- Stack
- TreeSet / TreeMap
- HashSet / HashMap
- PriorityQueue
Memory for a List

- Array (contiguous in memory)
  
  
  | 42 | -3 | 17 | 9 |
  
- Spread in memory
  
  
  [42] [9] [-3] [17]

Front
pseudo code

```python
ListNode
  data (int)
  next (ListNode)
```
A list node class

public class ListNode {
    public int data;
    public ListNode next;
}

• Each list node object stores:
  • one piece of integer data
  • a reference to another list node

• ListNode objects can be "linked" into chains to store a list of values:

- Data: 42
- Next: -3
- Data: 17
- Next: 9
- End
• Suppose we had the following `ListNodes`:

```
```

• What would the lists look like if we ran the code?

```java
list1.next = list2.next;
```
Reassigning references

• when you say:
  • \(a.\text{next} = b.\text{next};\)

• you are saying:
  • "Make variable \(a.\text{next}\) store to the same value as \(b.\text{next}.\)"
  • Or, "Make \(a.\text{next}\) refer to the same place as \(b.\text{next}.\)"

![Diagram showing reassignment of references]

10  20
30  40
References vs. objects

`variable = value;`

- **variable** (left side of `=`) place to put a reference (where the phone number goes; where the base of the arrow goes)
- **value** (right side of `=`) is the reference itself (the phone number; the destination of the arrow)

- adjust
- For the list at right:
  - `a.next = value;` means to `t` where points
  - `variable = a.next;` means to make `variable` point at
Linked node problem 3

• What set of statements turns this picture:

- list1
  - data: 10
  - next: data

- list2
  - data: 30
  - next: data

• Into this?

- list1
  - data: 10
  - next: data

- list2
  - data: 40
  - next: data
Linked node problem 3

- How many ListNode variables?

- Which variables change?

C = D;
list1.next.next = list2;
Linked node problem 3

- How many ListNode variables?

- Which variables change?

```
list1
  A
B
  data: 10
  next: C

list2
  D
E
  data: 30
  next: F

list1.next.next = list2

D = E;
13 + 2 = 13 + 2.next;
or
13 + 2 = 13 + 1.next.next.next;
```
Linked node problem 3

- How many ListNode variables?

```
list1
  A
  □
  □

list2
  D
  □
  □
```

- Which variables change?

```
list1
  □
  □

list2
  □
  □
```
Linked node problem 3

- How many ListNode variables?

- Which variables change?

```
list1.next.next = list2
list2 = list2.next
list1.next.next.next = null
```
Linked node problem 4

- What set of statements turns this picture:

  list1
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

  list2
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

- Into this?

  list1
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

  list2
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>