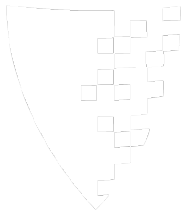
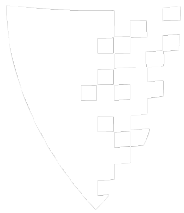


Linked Lists

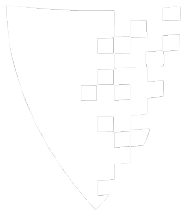
snarf today's code



Announcements

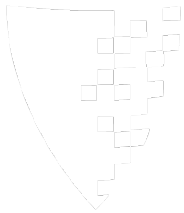
- Jotto – Due Tomorrow
- APT Set 3 – Due Feb 12

- Exam 1 – Feb 15



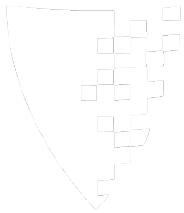
Today

- Linked Lists
- Queues
- Stacks



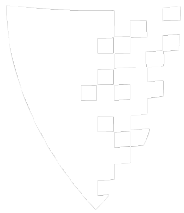
List

- ArrayList



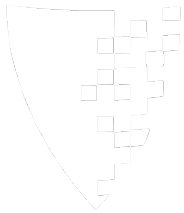
List

- ArrayList



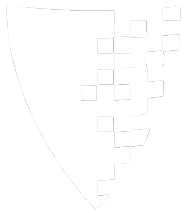
List

- ArrayList
  - List – ordered collection of values
    - List Interface
      - add(E elem)
      - add(int index, E elem)
      - contains(Obeject o)
      - an many more!



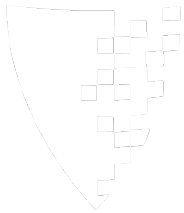
List

- ArrayList
  - List – ordered collection of values
    - List Interface
      - add(E elem)
      - add(int index, E elem)
      - contains(Obeject o)
      - an many more!



List

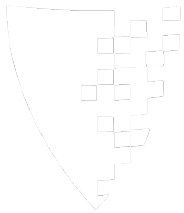
- ArrayList
  - Array – a data structure in which elements may be located by index



List

- ArrayList implements List
  - inherits all methods from List
    - ArrayList implements List's methods using an array

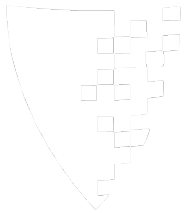
  - ArrayList<String> array = new ArrayList<String>();
  - OR
  - List<String> array = new ArrayList<String>();



List

- What are ArrayLists good for?
  - Constant time O(1)
    - size
    - isEmpty
    - get
    - set
    - add*
  - Linear time O(N)
    - everything else

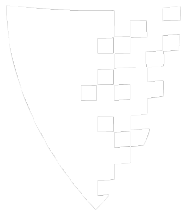
* add is ~O(1)



List

- list.add(“C”);
- list.add(“o”);
- list.add(“m”);
- list.add(“p”);
- list.add(“s”);
- list.add(“c”);
- list.add(“i”);

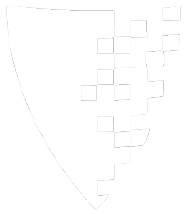
C	o	m	p	s	c	i



List

- How would you
  - add to the end, beginning, middle?
  - remove an element?

C	o	m	p	s	c	i

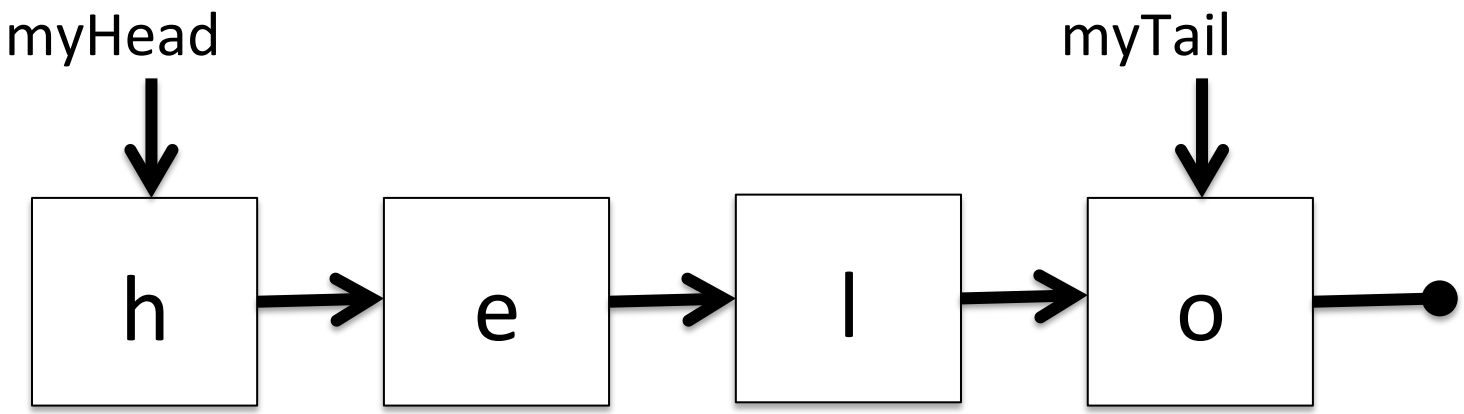


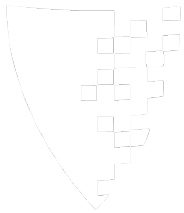
Linked List

- LinkedList
  - List – ordered collection of values
    - List Interface
      - add(E elem)
      - add(int index, E elem)
      - contains(Obeject o)
      - an many more!

Linked List

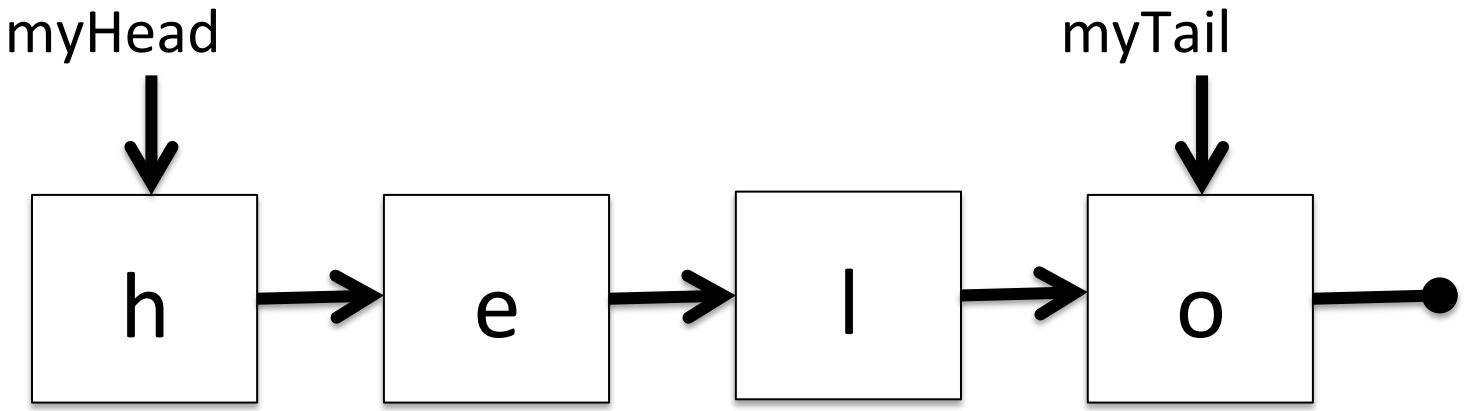
- LinkedList
  - different implementation from ArrayList





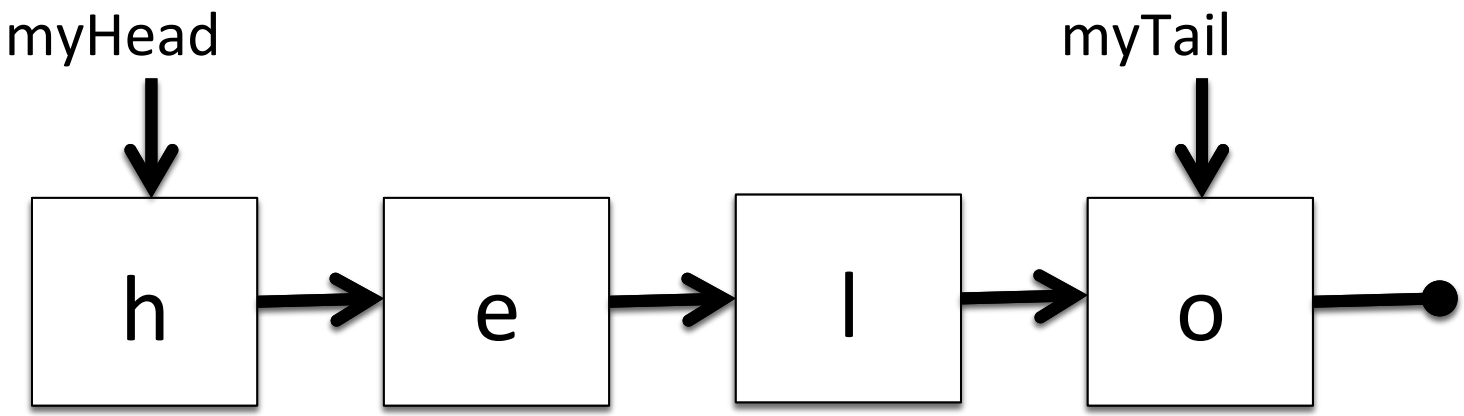
Linked List

- LinkedList
  - Nodes
    - data
    - pointer to the next node
  - Pointer to beginning and (sometimes) end



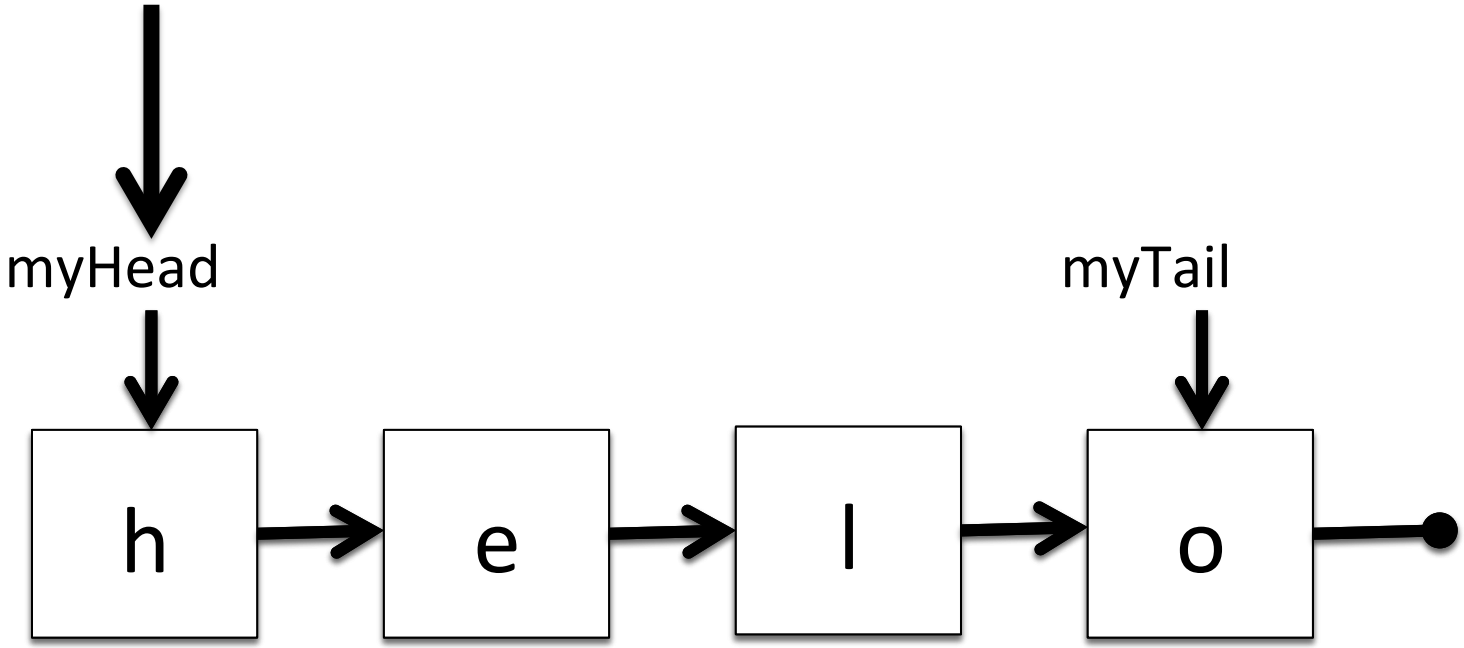
Linked List

- insert “l” after “e”



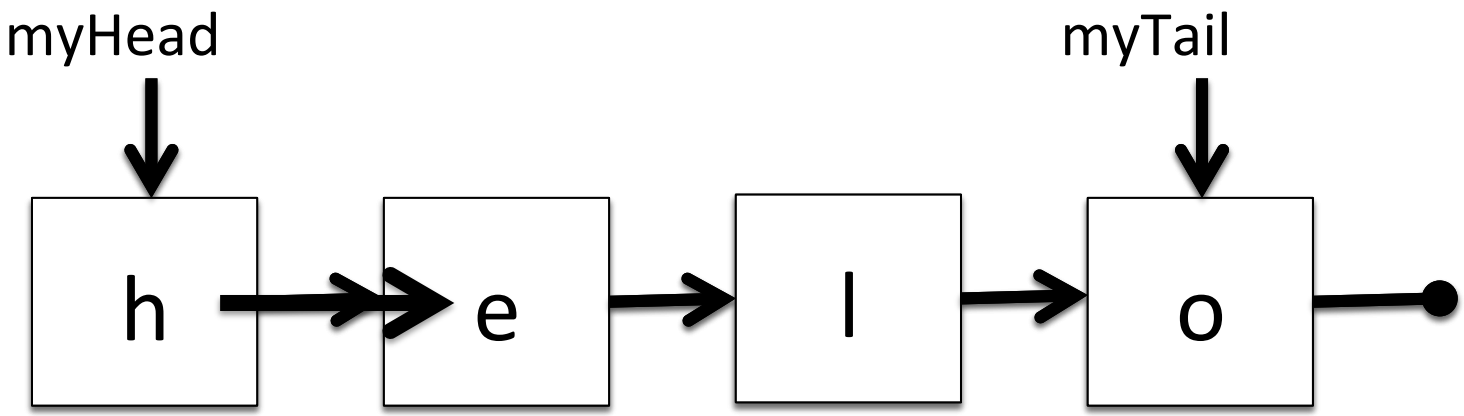
Linked List

- insert “l” after “e”
  - find the location



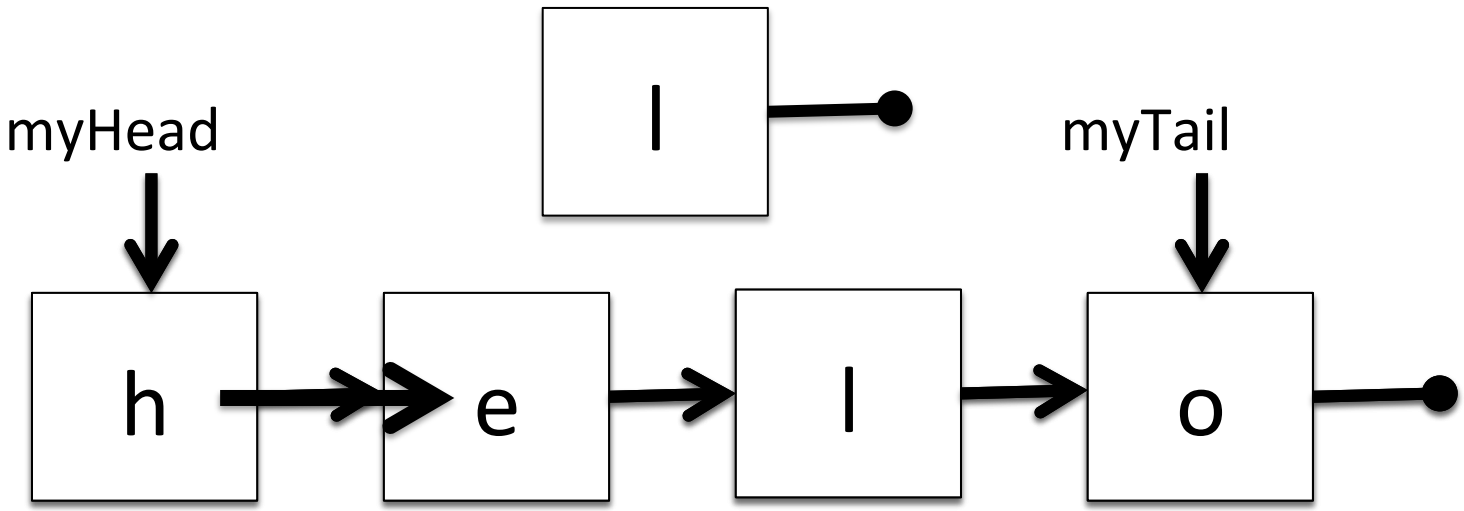
Linked List

- insert “l” after “e”
  - find the location



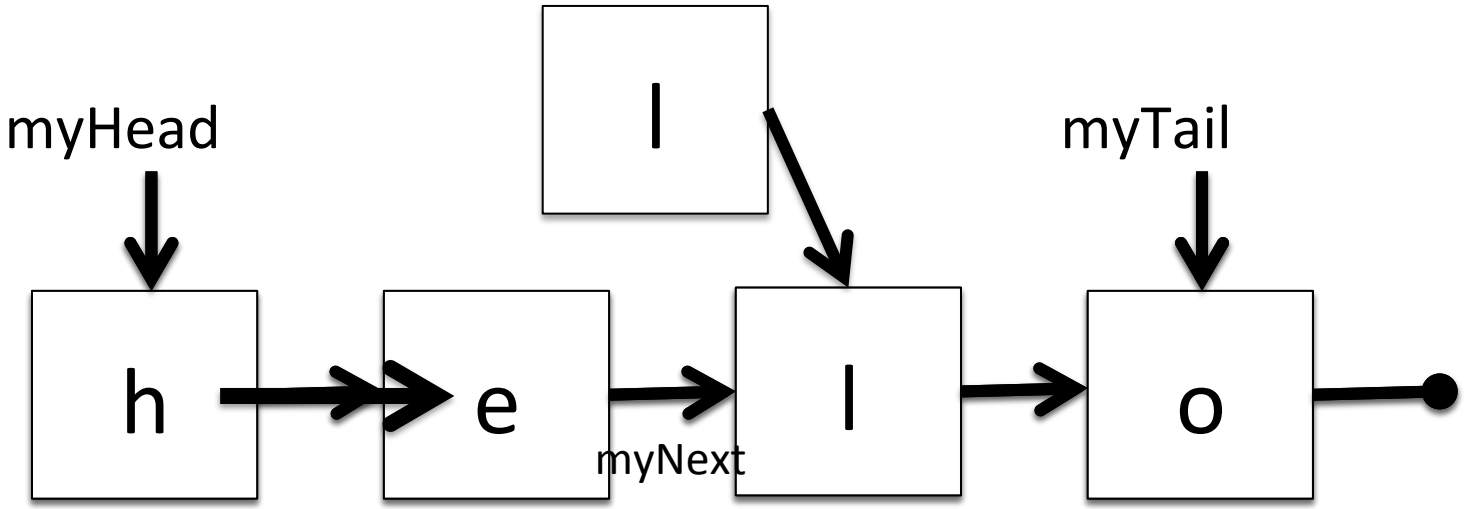
Linked List

- insert “l” after “e”
  - find the location
  - create a new node



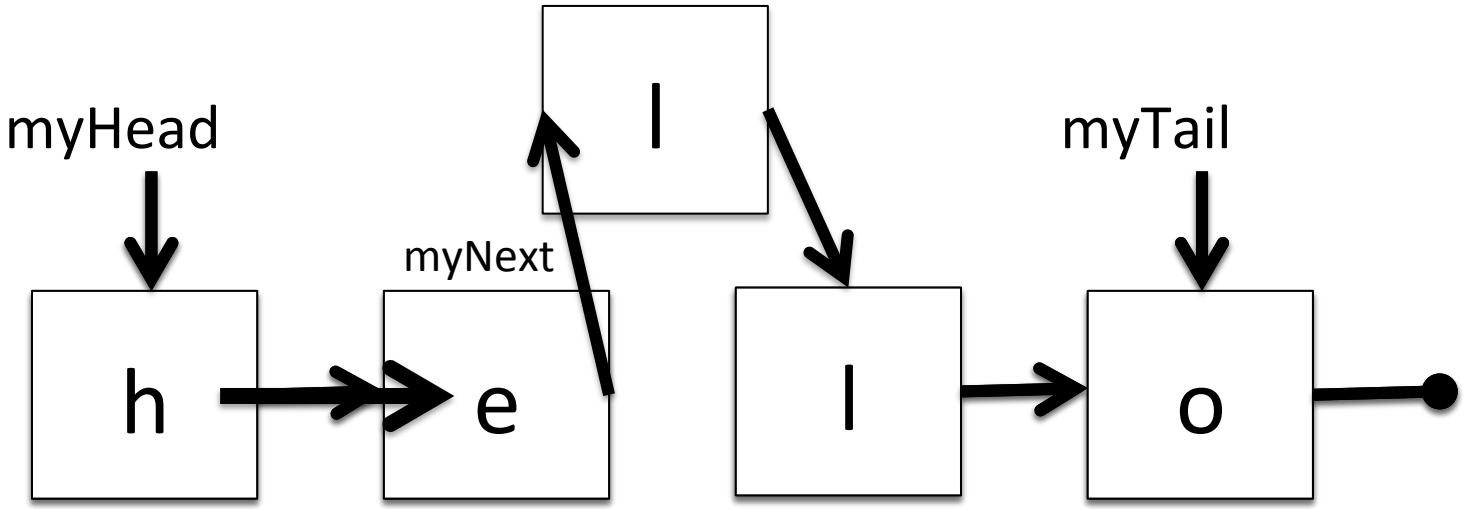
Linked List

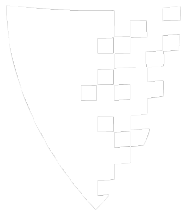
- insert “l” after “e”
  - find the location
  - create a new node
  - update pointer to myNext



Linked List

- insert “l” after “e”
  - find the location
  - create a new node
  - update newNode to myNext
  - update myNext to newNode



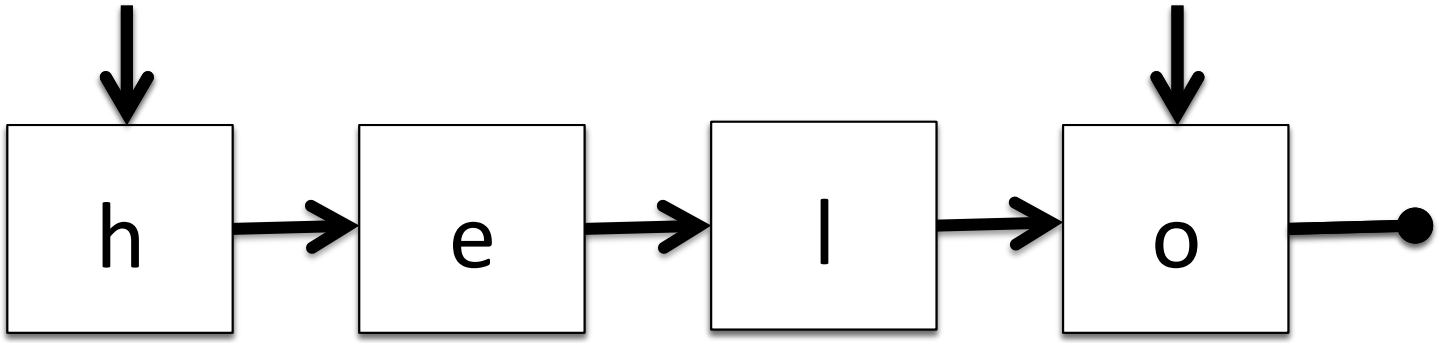


Linked List

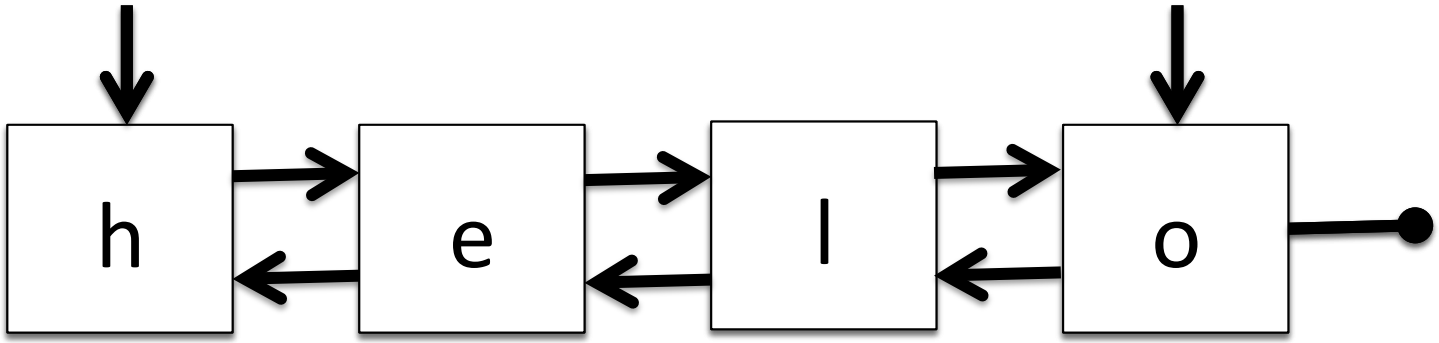
- Code time

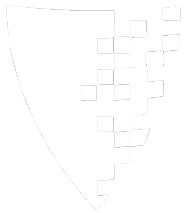
Linked List

- Singly linked list



- doubly linked list



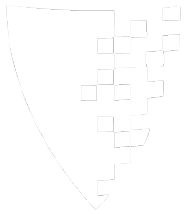


Linked List

- LinkedList vs. ArrayList

	get	**add at end**	**add in middle**
LinkedList	O(N)	O(1)	O(1)
ArrayList	O(1)	O(1) but O(N) worst case	Shift elements

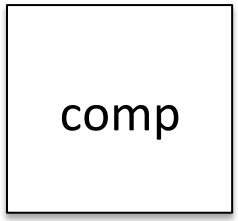
Queue



Queue

- First In First Out (FIFO)

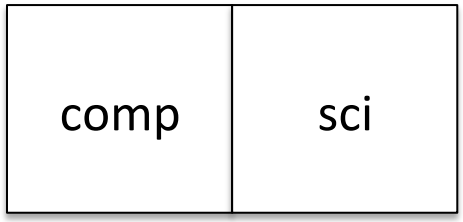
```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

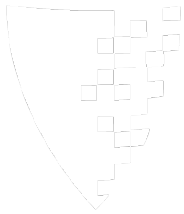


Queue

- First In First Out (FIFO)

```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

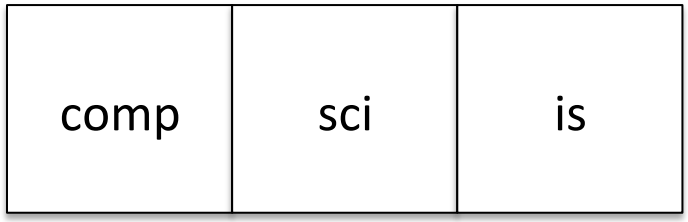




Queue

- First In First Out (FIFO)

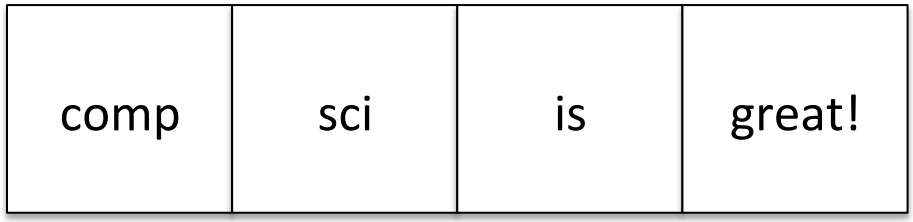
```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```



Queue

- First In First Out (FIFO)

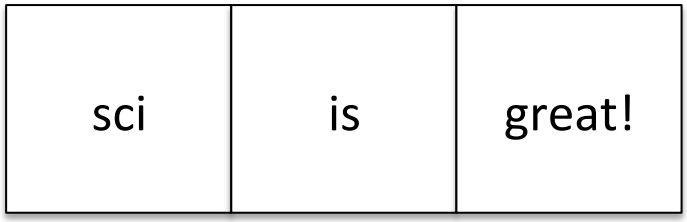
```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```



Queue

- First In First Out (FIFO)

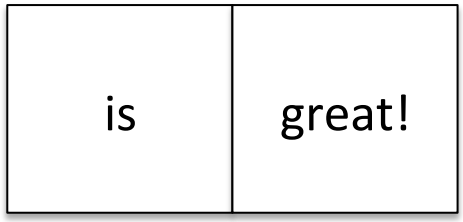
```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

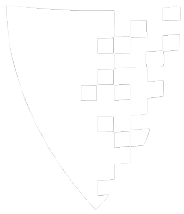


Queue

- First In First Out (FIFO)

```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

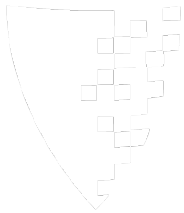




Queue

- First In First Out (FIFO)

```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

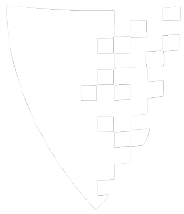


Queue

- First In First Out (FIFO)

```
1   Queue<String> q = new LinkedList<String>();
2           q.add("comp ");
3           q.add("sci ");
4           q.add("is ");
5           q.add("great!");
6           while(!q.isEmpty())
7               System.out.print(q.remove());
```

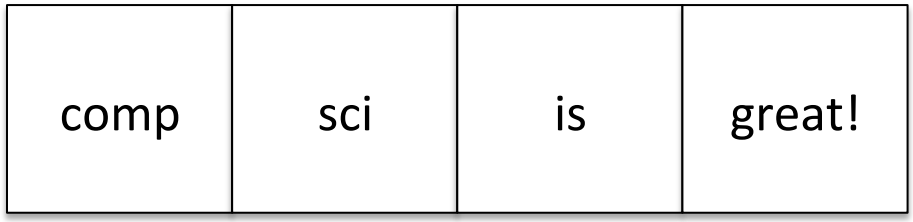
"Comp sci is great!"

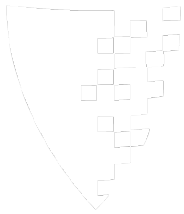


Stack

- Last In First Out (LIFO)

```
1       Stack<String> q = new Stack<String>();
2       q.push("comp ");
3       q.push("sci ");
4       q.push("is ");
5       q.push("great!");
6       while(!q.isEmpty())
7           System.out.print(q.pop());
```

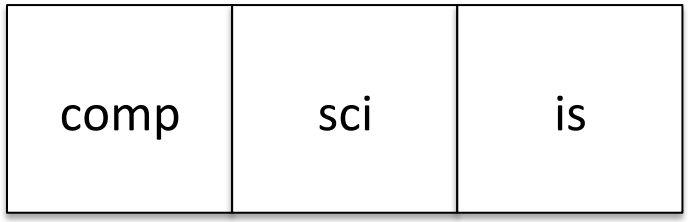


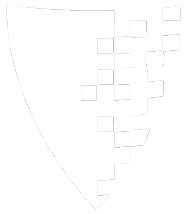


Stack

- Last In First Out (LIFO)

```
1        Stack<String> q = new Stack<String>();
2        q.push("comp ");
3        q.push("sci ");
4        q.push("is ");
5        q.push("great!");
6        while(!q.isEmpty())
7            System.out.print(q.pop());
```

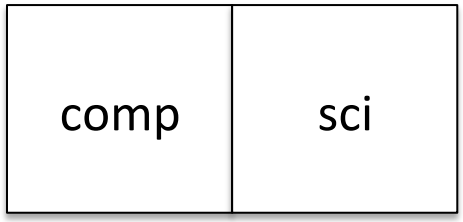


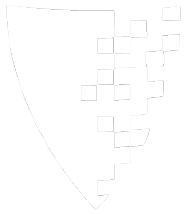


Stack

- Last In First Out (LIFO)

```
1       Stack<String> q = new Stack<String>();
2       q.push("comp ");
3       q.push("sci ");
4       q.push("is ");
5       q.push("great!");
6       while(!q.isEmpty())
7           System.out.print(q.pop());
```

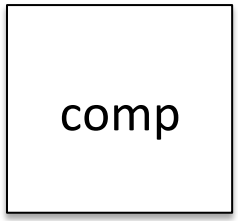


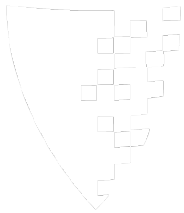


Stack

- Last In First Out (LIFO)

```
1        Stack<String> q = new Stack<String>();
2        q.push("comp ");
3        q.push("sci ");
4        q.push("is ");
5        q.push("great!");
6        while(!q.isEmpty())
7            System.out.print(q.pop());
```





Stack

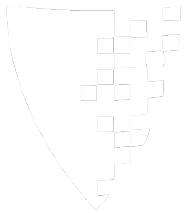
- Last In First Out (LIFO)

```
1       Stack<String> q = new Stack<String>();
2       q.push("comp ");
3       q.push("sci ");
4       q.push("is ");
5       q.push("great! ");
6       while(!q.isEmpty())
7           System.out.print(q.pop());
```

"great! is comp sci"

Questions!!!!

- Google Form
  - http://goo.gl/EkYHh



Today

- Linked Lists
- Queues
- Stacks