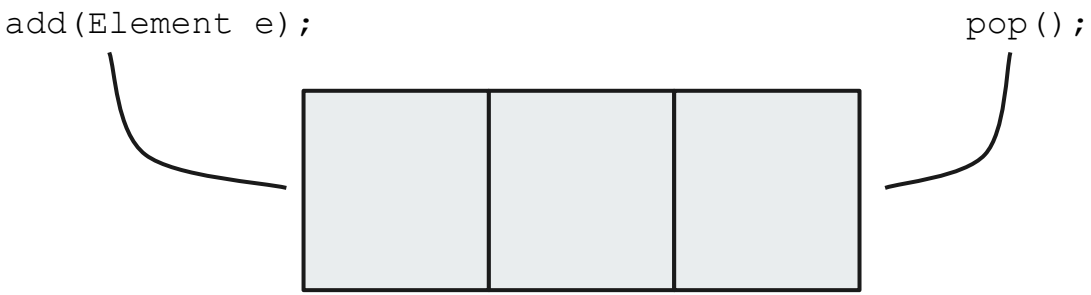
Queues in Java

Intermediate Programming
Leap@CMU 2017

What are Queues?

- Data Structure
- **F**irst **I**n **F**irst **O**ut



Note: Do not use `add(Element e)`, this will treat it as a stack and not a queue

Because Java

- Queues are an interface in Java, so you cannot declare a queue.
- A linked list is an implementation of a queue already available in Java (however, many more do exist)

```
LinkedList<Integer> aRandomQueue = new LinkedList<Integer>();
```

Queue Project

Goal: Create a Radix Sort algorithm

Input: The number of elements in a randomly generated array

Output: The sorted array

Example:

```
Num Elements: 10
Sorted array: [1, 2, 3, 4, 4, 5, 5, 78, 100, 101]
```

Radix Sort Explained

Radix sort is a non-comparison sort based upon the orderings of digits in a number.

1. Take a list of random numbers, say `[8, 2, 101, 984, 32, 67, 115]`
2. We then find the number with the most digits: `[101, 984, 115]`, each has 3 digits. This is the number of times you're going to iterate through the algorithm
3. Now you're going to create 10 bins number 0 - 9
4. For each iterations, you are going to place the numbers in their respective bin based upon their n^{th} digit from the right (e.g. on iteration 1, look at the ones place, on iteration 2, look at the 10s place, etc)

Worked Example

`[8, 2, 101, 984, 32, 67, 115]`

Iteration 1:

	101	32 2		984	115		67	8	
0	1	2	3	4	5	6	7	8	9

Worked Example (cont.)

`[101, 2, 32, 984, 115, 67, 8]`

Iteration 2:

8 2 101	115		32			67		984	
0	1	2	3	4	5	6	7	8	9

Worked Example (cont.)

`[101, 2, 8, 115, 32, 67, 984]`

Iteration 3:

67 32 8 2	115 101								984
0	1	2	3	4	5	6	7	8	9

Done! `[2, 8, 32, 67, 101, 115, 984]`

Project Guidelines

[*] = Implement Radix Sort

[**] = User input (with error catching) and random array generation

[***] = Negative numbers

[****] = Lexicographical (alphabetical) sorting

[*****] = Big O Notation