- Academistrative
    - Midterm thoughts (3 comments about the midterm)
        - Attempting to keep to our one-week grading rule
        - Already done some grading, and we have a massive grading session on Wednesday
            - Buying TA's Pizza Luce so your grades will go up
    - HW2 due on Friday
- Today
    - Continuing our discussion of Lists
        - Context: beginning half of the class more dedicated to **traditional** data structures than object-oriented programming
            - Although OOP will still play a role
    - Tomorrow's lab will involve a hand's on version of what we're doing now
        - Expecting it to be a short one after the midterm + upcoming HW
- List ADT
    - Manifest in Java in the List interface
    - SLIDE: key methods in Java's list interface
        - What do a few of these do?
    - We've talked about how to implement a list using an array as a backing data structure
        - Contains on the midterm
    - However, interfaces are COMPLETELY agnostic
        - So is the idea of List Abstract Data Type
    - To get this point across, I thought I'd pose a thought question
        - To provide context for this question, I'd like to briefly introduce Mechanical Turk
            - Define what it is
            - Crowdsourcing central
            - World's first large crowdsourcing marketplaces
            - Q: What are some common use cases?
                - Things that computers need to know, but can't do themselves
                - Tagging images
                - Transcription
                - Categorizing things
                - Writing prose
                - Content moderation
                - Often used to train new algorithms
        - SLIDE: Mechanical has a Java API, meaning we can request tasks directly from within our Java programs
            - Q (Pairs): How might we use this to implement the list interface / implement a list abstract data type?
                - A: Looking for someone who implements a "real-life" list

        - Thought about implementing this for you folks, but didn't have time
            - Good side project, although it will cost a small amount of money
    - Key point: abstract data types are general ideas with many different possible implementations, each with its advantages and disadvantages
        - What are some advantages and disadvantages of our MechanicalTurkList?
- List ADT with Linked Lists
    - Move back into the realm of practicality and talk about a list implementation that is
        - commonly used
        - but that is really fundamentally different from array-backed lists
    - Write on board the key data structures in terms of how it is implemented
    - Q (Pairs): How might we implement add? [5-min]
        - More or less efficient than array-backed data store?
    - Q (Pairs): how do we implement T get(int index)
        - More or less efficient than array-backed store
    - Q (if extra time)
        - How might we implement T get(int index) in a significantly more efficient fashion?
    - Reminder: You'll use Linked ATT in your life as a programmer, and now you've implemented at a relatively low level two of the key methods.
- Next time
    - Focusing on sorting, efficiency and Big-O notation