This lecture is about representing words. In text processing, dealing with orthography and morphology go hand in hand, particularly for English. Since this class is about text, we'll mainly talk about how each of these appears in written language.

Tokenization: breaking raw text into useful units. In English, punctuation provides partial information, but most punctuation is ambiguous. Related, slightly easier(??) problem: sentence segmentation. Related, much harder problem: Chinese word segmentation (not always clear even for native Chinese speakers!).

Morphological parsing: breaking a word into its morphemes, or its root and the morphological **features** that its morphemes represent. Ingredients:

- Lexicon (database of all the morphemes and types)
- Morphology rules (how morphemes can be put together)
- Orthography rules (what happens when two morphemes join)

Example: to analyze *kitties*, we need morphemes *kitty* and *s*/+PL (from the lexicon), we need to know that the plural marker comes after the root (morphology rule), and we need to know that *kittys* is properly spelled *kitties* (orthography rule).

Finite-state automaton (FSA): finite set of states Q, finite set of symbols Σ, special initial state $q_0 \in Q$, set of final states $F \subseteq Q$, and set of directed edges between states, labeled with Σ^* (sometimes called "transitions" or "arcs"). Recognizer for regular **languages**. Lots more in chapter 2, if you want a refresher.

Finite-state transducer (FST): finite set of states Q, finite set of input symbols Σ, finite set of output symbols Δ, special initial state $q_0 \in Q$, set of final states $F \subseteq Q$, and set of directed edges between states, labeled with $\Sigma^* : \Delta^*$. Recognizer for regular **relations**.

FSTs provide a powerful way to represent morphological systems, including the lexicon and the morphological and spelling rules, including irregular forms. They can be combined (e.g., in **cascades** and through **intersection**). They also let us represent ambiguity and optionality (two kinds of non-determinism).

Stemming: a simple, context-independent mapping of words to simpler strings (which may be non-words) so that related words are mapped together. The Porter stemmer is the most widely known stemmer; it is defined by a set of substitution rules. Stemming is sometimes useful for simplifying our data, but it's not very linguistically satisfying!

To-do list:

- Turn in your signed cheating form to your TA.
- If you have not yet, use **Piazza** to tell us your andrew account so we can add you to the course page.
- Look out for Assignment 1 which is to be released shortly. **Note: You can only submit assignments if you have done the first two items in the list.**
- Intro survey so we can get to know you: `http://tiny.cc/NLParty`

- Readings for this lecture: *SLP* 3.1, 3.9 (and any other parts of chapter 3 you're interested in).
- Start forming your team.