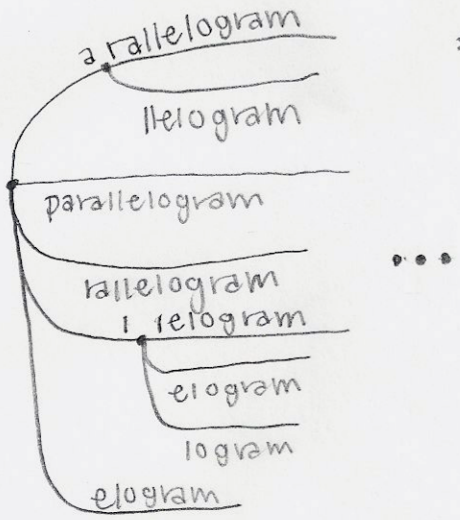
CS157 Lecture #12

SUFFIX TREE: stores every "suffix" (ending) of an input

INPUT: parallelogram



...

total # letters = $\sum_{i=1}^{n} i = O(n^2)$ [whoops!]

* But what about the structure? n leaves, one for each suffix
  - ↳ compactly store by storing index of starting & ending letter of suffix = O(n) storage [yay!]
  - actually a mess to store in practice (but that's fine — more important to think of what we can do with it)

★ **SEARCHING FOR SUBSTRINGS**: only 1 branch at each step storing a given level, so we can search for a substring in linear time

★ **LONGEST COMMON SUBSTRING**: (IDEA 1) "overlay" 1 tree on the other: find the node that has children from both trees & is longest

(IDEA 2) build a suffix tree for [word 1]"X"[word 2]. run regular suffix tree algo + post-processing