Course Announcements

- Bacon due in 11 days.
- Term project teams due today.
- Individuals have a separate google form.
Autocorrect Review

John Jannotti

CSCI032 (cs0320)

March 1, 2017
Today, we’ll look at three implementations of Autocorrect
  ▶ Two by students, Alberta Devor and William Povell. Thanks.
  ▶ And one by me, experimenting with a design that might be too complex, but I hope illustrates an idea.
Nice things we looked for

- README states that Trie cannot handle all unicode chars, OR
- Trie can handle all unicode chars
- Trie implements Collection or Set<String>
- Interface (or abstract superclass) for Suggesters
- Interface for Rankers or implements Comparator<String>
- Ability to make composite suggestion generators (ex: prefix+led suggester)
- Trie implements both contains() and add() methods
- Trie implements a search()/suffixes() method that returns a subtree
- Trie implements delete() (doesn’t have to garbage collect nodes)
- Implemented a Compact Trie
- Node object does not store its own letter (redundant)
- Separates command loop nicely.
- Trie accepts a CharSequence rather than a String
- Closes files properly in a finally block or try with resources
Some bad things we looked for

- Trie stores full String in leaves
- LED is brute force
- Whitespace does not use the Trie
- Prefix does not use the Trie
Lay of the land, Alberta

- 24 sources files, 1961 lines (includes Stars, of course)
- test files, 1716 lines, 134 individual tests.
- Seven packages. Maybe going too far?
  - Looks like a package for every idea, rather than related ideas.
  - Maybe kdtree and trie belong together.
  - Are sortable and ranker “package-level” ideas?
As of last year, so autocorrect only.

21 source files, 979 lines (plus an experiment)

10 more system tests (we gave out 18)

Three packages: autocorrect, collect, text
  collect: Trie, Pair
  text: Corpus, Bigram, Completer
  autocorrect: Main, Rankers, Autocorrect*
Smart sounds interesting. Will look to see if it reimplements, or uses the basic ranker.

Any info on whether trigrams are common enough to matter in reasonable corpora?

Storing the word itself defeats most of the purpose of the Trie.

HashMapValComparator sounds nice. We should see if a lambda expression could do it though.
Notes on README, William

- Doesn’t mention the “getter” commands.
- GUI sounds good (keyboard and mouse)
- Smart is interesting. Any commentary on whether it “works”? 
Notes on build and run, Alberta

- During mvn package, unit tests printed errors.
  - We’ll look at her command loop and error handling.
- prefix gets turned on when whitespace is activated.
- GUI should prefill 0, on interpret blank as 0.
- GUI doesn’t support whitespace.
- GUI didn’t have a way to pick suggestions.
Considerably smaller. Web/Cli has been moved out.
Corpus holds information about a language. (Trie, Bigrams inside)
Next places to look: Corpus, Autocorrector (Gui/Cli)