**Class LexiconNode:**

Managing Trie Node Children:

* Use a LinkedList to store children references to children nodes as it will help with adding and removing a word in between the trie.

Instance Variables

* Char letter = letter stored in the node
* Boolean isWord = whether at this node all the letters of the nodes in the path combine to form a word

Methods:

* Constructor (char letter, boolean isWord):
  + Initializes doubly linked list
  + Sets up char letter and Boolean isWord
* compareTo (LexiconNode o):
  + returns o.letter – letter
* addChild(LexiconNode ln):
  + create a doublylinked node called finger.
  + While finger != null… finger.next()
  + If ln.compareTo(finger) < 0, insertafter(finger.previous())
* getChild(char ch):
  + While finger.next() != null…finger.next()
  + If (ch == finger.letter), return finger
* removeChild(char ch):
  + While finger.next() != null…finger.next()
  + If(ch == finger.letter), remove(finger)
* Iterator():
  + Return children.iterator(), which is a linkedlist iterator.