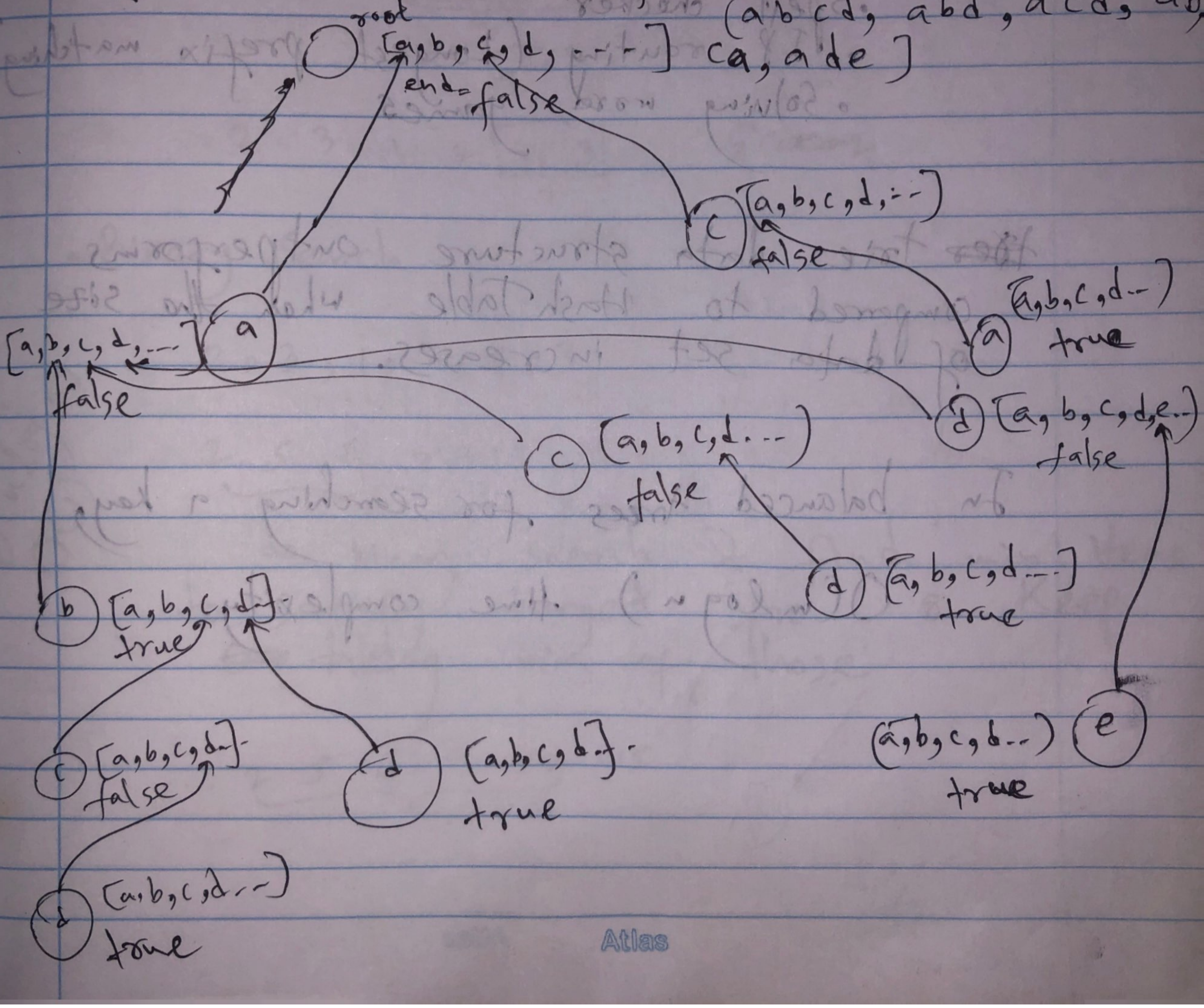
*Implement Trie (prefix Tree).

[abcd, abd, acd, ab, ca, ade]



key insertion to trie → $O(m)$ time
m - key length
$O(m)$ Space

Search for a key → $O(m)$ time
$O(1)$ space

search for a prefix → $O(m)$
$O(1)$

applications;

- auto complete
- Spell checker
- IP routing (Longest prefix matching)
- Solving word games

trie data structure outperforms compared to HashTable when the size of data set increases.

In balanced trees for searching a key,
$O(m \log n)$ time complexity.