Sean Blanchard
9/27/20

Problem Set 4

4-1 a) No, because reordering the words in a string will not produce a different hash.

b) Both are necessary to ~~add~~ to implement a hash table, she definitely needs to implement dynamic resizing, because it will insert everything right on top of eachother, and this would cause your search to be inefficient. You also must have collision resolution as it can be reduced but never avoided altogether.

c) Hash table suffer from O(n) worst time complexity. We must iterate through the each slot of the old table to make sure each slot is copied, this would take O(n) time.
We must create our new table which takes O(m') time.
$O(n) + O(m') \approx \boxed{O(n+m')}$

d)

4-2 Python Dictionary

a) Membership Testing

- Dictionaries of any size. Created once and then rarely changes.
- Single write to each key
- Many calls to __contains__() or has_key()
- Similar access patterns occur w/ replacement dictionaries such as with the % formatting operator.

2) Many insertions right after creation, then only lookups

b) We are going to want a large size and hash function. Since membership testing is lookups it can use a big hash table w/ open spots. thus a large minimum size and a growth rate of 4.

4-3