Question 5:
initializeCandidates - Time complexity O(n^2)

- Recursion O(n)
- Looping through each candidate in initialized candidates O(n)
- Putting inside voteCount hashmap O(1)

Space complexity O(1)

- We are inserting the candidates in initialized candidates into voteCounts Hashmap O(1)
- Because the Hashmap was not initialized inside of the method I didn't account for it

castVote Time complexity - O(1)

- .contains is O(1) time complexity to find the key, and see if it exits
- To put a vote inside of the HashMap for the candidate is also O(1) time
- Incrementing to totalVotes++ is constant operation

Space complexity - O(1)

- There is nothing being initialized here
- Adding a vote to totalVotes takes constant amount of space
- Incrementing totalVotes takes constant amount of space as well

castRandomVote Time complexity - O(n)

- It takes constant amount of time to generate a random number
- However to retrieve the randomly selected candidate from the ArrayList it takes O(n) because we have to iterate through each element within the list
- And then casting a vote is O(1) time

Space complexity - O(1)

- It takes constant time to initialize the pick variable

- To cast the vote it takes constant amount of space as well

rightElection Time complexity - O(n)

- The first if statement is just a comparison statement between 2 integers O(1)
- The second if statement checking if the candidate does exist in the HashMap is a constant operation.
- Targetvotes is initialized n times until the rigged candidate has won
- The for loop iterates through each key pair value in the voteCounts hashmap, from there on we check if the key and value align with each other. O(n) operation
- E,getvalue gets the current entries vote count, and compares to topOtherVotes and topOtherVotes was initialized to -1, so for any real vote count will be greater tan -1, this means we're looking for does the candidate have more votes than the best so far. When its true, we update the topOtherVotes to the large vote count, and hte we record the key, this is a O(n) time complexity operation, because we have to iterate through until the candidates has the most votes.
- voteCount.put is a constant operation

Space complexity - O(n)

targetVotes takes O(n) space because it has to be initialized n times until while loop stops running and that happens only when the candidate has hightest votes

getTopCandidates

Time complexity - O(n)

- Initiazling pq a maxheap takes O(1) times
- Adding all the voteCounts to maxHeap takes O(logn) for each entry, into the maxHeap
- Creating an arraylist putting inside the top candidates takes O(n) time operations
- looping through until reaching the number of top candidates by votes takes O(n) time

Space Complexity - O(n)

- MaxHeap takes O(n) amount of space for all the candidates organized by vote count
- Initialzing an arraylist and filing it with top candidates for k given takes O(n) amount of space

auditElection - Time complexity O(logn)

- Initiazling pq a maxheap takes O(1) times
- Adding all the voteCounts to maxHeap takes O(logn) for each entry, into the maxHeap
- In the while loop repeatedly removes the current highest voted pq.poll entry Prints the candidates with their vote counts takes takes O(nlog n ) time

Space complexity - O(n)

- MaxHeap takes O(n) amount of space for all the candidates organized by vote count
- Pq takes constant amount of space as well
- however because we iterate through candidates its initialized logn times