**Notable obstacles overcome:**

I tried using a for loop to loop through the entire string when checking if it’s syntactically correct, but I couldn’t logically figure out how to increment the counter by the correct number of characters depending on whether the tally was one or two digits without rereading certain characters. I overcame this by using a while loop instead and just breaking out when any of the conditions were not satisfied. Furthermore, cutting off each part of the string once it’s been checked felt more organized logically to me as it ensures no part of the string is being read twice in a logical error in my code.

When checking the test cases provided on the linux server, my code wasn’t producing the correct voteTally, however it was producing the correct number when I was outputting from my own main function. This was because I wasn’t resetting voteTally within the tallyVotes function and so it was just adding onto the value that tallyVotes was initialized as (which in the test cases, wasn’t 0).

**Pseudocode:**

**main**

call the two functions

output error messages or vote tally

**Check state code function**

Match parameter with valid state code

Return true/false

**Syntactically correct function**

If empty string

return valid

Repeatedly:

Check if character is an alphabetical letter

Set letter to uppercase

Repeatedly:

Find start of state forecast with one or two digits

If not 1 or 2 digits, return false

find/check valid state

If not, return false

Verify next character is a party (letter)

If not, return false

Cut off part of string that was checked

When length of word is 0, return true

**Tally votes function**

Repeatedly:

Check if character is an alphabetical letter

Set letter to uppercase

make party char uppercase

Check syntactically correct

If not, return 1

Check party parameter is an alphabetic letter

If not, return 2

Repeatedly:

Check if vote # is 0 or 00

If so, return 3

Set voteTally to 0

Repeatedly:

Find beginning of each state forecast

Convert tally # to int

Add to voteTally

Return 0

**Test data**

Testing isSyntacticallyCorrect

Bool b = isSyntacticallyCorrect (““);

Checks that an empty string would return true/valid

Bool b = isSyntacticallyCorrect (“3CAD“);

Checks for one digit vote #, valid state, and valid character for party

Would return true

Bool b = isSyntacticallyCorrect (“3rrd5CAD“);

Checks for one digit vote #, invalid state, and valid character for party

Would return false

Bool b = isSyntacticallyCorrect (“3CA\*5war“);

Checks for one digit vote #, valid state, and invalid character for party

Would return false

Bool b = isSyntacticallyCorrect (“33CAD“);

Checks for two digit vote #, valid state, and valid character for party

Would return true

Bool b = isSyntacticallyCorrect (“33rrd5cad“);

Checks for two digit vote #, invalid state, and valid character for party

Would return false

Bool b = isSyntacticallyCorrect (“33CA\*33wad“);

Checks for two digit vote #, valid state, and invalid character for party

Would return false

Bool b = isSyntacticallyCorrect (“3rr&5cad“);

Checks for one digit vote # and both invalid state and party

Would return false

Bool b = isSyntacticallyCorrect (“33rr&“);

Checks for two digit vote # and both invalid state and party

Would return false

Bool b = isSyntacticallyCorrect (“123123132rr\* “);

Invalid vote #, invalid state, invalid party

Would return false

Bool b = isSyntacticallyCorrect (“1231CAD“);

Checks for more than two digits vote #, valid state, and valid character for party

Would return false

Bool b = isSyntacticallyCorrect (“3CAD4waD6ILD9IDD“);

Checks valid case with all one digit votes

Would return true

Bool b = isSyntacticallyCorrect (“4 “);

Checks valid case with all two digit votes

Would return true

Bool b = isSyntacticallyCorrect (“3CAD99MED8MDD55MAD“);

Checks valid case with mix of one and two digit votes

Would return true

Testing tallyVotes

Int v = 0; int r = tallyVotes (“12321391439ca\*“, ‘d ‘, v)

Checks for a non syntactically correct pollData

Would return 1 w/ unchanged voteTally

* Other invalid poll data from above cases that would all return 1 w/ unchanged voteTally

Int v = 0; int r = tallyVotes (“1CAD2WAD3MDD“, ‘\* ‘, v)

Checks if party is a non valid character

Would return 2 w/ unchanged voteTally

Int v = 0; int r = tallyVotes (“1cad00wad “, ‘d ‘, v)

Checks for two zeros

Would return 3 w/ unchanged voteTally

Int v = 0; int r = tallyVotes (“1cad0wad0ild “, ‘d ‘, v)

Checks for single zero

Would return 3 w/ unchanged voteTally

Int v = 0; int r = tallyVotes (“22cad33ilr33wad “, ‘d‘, v)

Only two digit votes

Would return 0 w/ updated voteTally value

Int v = 0; int r = tallyVotes (“2wad3ilr3mdd “, ‘d‘, v)

Only one digit votes

Would return 0 w/ updated voteTally value

Int v = 0; int r = tallyVotes (“2wad33ilr3mdd “, ‘d‘, v)

Both one and two digit votes

Would return 0 w/ updated voteTally value

Int v = 9999; int r = tallyVotes (“1CAD2WAD3ILD“, ‘\* ‘, v)

Checks if party is a non valid character

Would return 2 w/ unchanged voteTally 9999

Int v = 9999; int r = tallyVotes (“1cad00wad “, ‘d ‘, v)

Checks for two zeros

Would return 3 w/ unchanged voteTally 9999

Int v = 9999; int r = tallyVotes (“1cad0wad0mdd “, ‘d ‘, v)

Checks for single zero

Would return 3 w/ unchanged voteTally 9999

Int v = 9999; int r = tallyVotes (“22wad33car33ild “, ‘d‘, v)

Only two digit votes

Would return 0 w/ updated voteTally value, must start tallying from 0

Int v = 9999; int r = tallyVotes (“2cad3war3mdd “, ‘d‘, v)

Only one digit votes

Would return 0 w/ updated voteTally value, must start tallying from 0

Int v = 9999; int r = tallyVotes (“2cad33war3mdd “, ‘d‘, v)

Both one and two digit votes

Would return 0 w/ updated voteTally value, must start tallying from 0

Int v = 0; int r = tallyVotes (“2cad33car3cad “, ‘d‘, v)

Case with several forecasts for the same state

tallyVotes may not work correctly

Int v = 0; int r = tallyVotes (“99cad99war99ILd “, ‘d‘, v)

Case with more than the valid # of electoral votes for a state

tallyVotes would work, but it would not be realistic