Report

1. The first obstacle I ran into was figuring out how to properly substring the state forecasts between commas. Unlike Java, in C++ the substring method takes the parameter of length. To solve this issue, I calculated length by subtracting the start index from the end index and then used that as the second parameter. However, this ran into a problem of cutting off the last character in the string. I solved this by adding a comma, so I would not have to create a separate loop for that specific case. The second problem I ran into was returning the incorrect values of seatTally after running multiple assert statements. I realized that this was because I was not resetting the value of seatTally after its 1st call, so I assigned it back to 0 within the else statement. A final error was not recapitalizing everything within tally Seats. I had forgotten that the string was not global, and therefore all the lowercase letters needed to be turned into uppercase again.
2. Declare function and take in parameter

Declare constant state codes

Returns true/false based on if state code is found

Declare function to check state forecast and take in parameter

Checks if first character is alphabet

Returns false if not

Repeatedly adds characters to a stateCode string

Checks if stateCode is valid, returns false if not

Checks first character after stateCode, returns false if not digit

Moves through string, returns false if more than 2 digits

Checks last character, returns false if not an alphabet

Repeatedly does above code until string ends

Returns true

Declare function to check syntax of poll data and take in parameter

For loop to check number of commas

For loop to capitalize lower case alphabets

Returns true if length is zero

If there is no comma counter, call helper function to check syntax

If multiple commas

Add a comma to string

Move through string

When a comma is found

Calculate length based on end index

Substring from the start position and length

Call helper function to check syntax, return false if incorrect

Repeatedly does above code, setting start position to end index

Returns boolean variable

Declare function to tally seats and take 3 parameters

For loop to capitalize lower case alphabets

Capitalizes party character if possible

Checks if string parameter has proper syntax, returns 1 if it doesn’t

Checks if part character is an alphabet, returns 2 if it doesn’t

Checks if string parameter has size of 0 or 2

Move through string starting after stateCode

If there are two numbers before party, multiply the first number by 10 and add it and second one to seat tally

If there is one number, add it to see tally

1. assert(hasProperSyntax("TX")); **//tests if statecode function works**

assert(hasProperSyntax("tX")); **//tests to make sure capitalization occurs**

assert(hasProperSyntax("Tx20R")); **//tests to see if isValidChecker works**

assert(hasProperSyntax("TX1D21R")); **//tests to see if isValidChecker works for multiple parties**

assert(hasProperSyntax("+TX1D") == false); **//tests to see if isValidChecker properly handles characters**

assert(hasProperSyntax("YY") == false); **//tests if statecode functions returns false correctly**

assert(hasProperSyntax("TX9R17D1I,Ny,CA3d5r4D,UT4R"));**//tests if hasProperSyntax works for multiple commas**

assert(hasProperSyntax("ut3r00456D") == false); **//tests if it returns false for size > 2 of number**

assert(hasProperSyntax("Ny9D,Vt\*3,NJ3r") == false); **//tests if hasProperSyntax properly returns false if syntax is wrong**

assert(hasProperSyntax("MA9D,UT4R") && hasProperSyntax("KS4R,MA9D"));**//given by Smallberg to check if commas work**

assert(hasProperSyntax("MA9D,,UT4R") == hasProperSyntax("KS4R,,MA9D"));

assert((tallySeats(" ", 'R', seatTally) == 1));**//checks to see if hasProperSyntax properly called in tallySeats**

assert((tallySeats("CA99D", '2', seatTally) == 2));**//checks to see if tallySeats properly returns if party character not an alphabet**

assert((tallySeats("", 'R', seatTally) == 0) && seatTally==0);**//checks to see if tallySeats works for empty string**

assert((tallySeats("VT", 'D', seatTally) == 0) && seatTally==0);**//checks to see if tallySeats works for just state code**

assert((tallySeats("MA1d", 'D', seatTally) == 0) && seatTally==1);**//checks to see if tallySeats properly tallies for one state forecast**

assert((tallySeats("TN87R", 'R', seatTally) == 0) && seatTally==87);**//checks to see if tallySeats proerly tallies for 2 digits**

assert((tallySeats("NY1D33R", 'R', seatTally) == 0) && seatTally==33);**//checks to see if tallySeats properly tallies with a forecast of multiple parties**

assert((tallySeats("NY12R,MA34R", 'R', seatTally) == 0) && seatTally==46);**//checks to see if tallySeats properly tallies with**

assert((tallySeats("TX9R17D1I,Ny,CA3d5r4D,KS4K", 'K', seatTally) == 0) && seatTally==4);**//checks to see if tallySeats properly avoids error if statecode and party are the same**