Oo, Thant Zin (Andy)

CS 31, Section 2F

1. The main obstacle I overcame while implementing poll.cpp was accommodating all the test cases in which hasProperSyntax had to check specifically for errors in the pollData string, such as commas before and after state forecasts. On a more general level, the main difficulties in Project 3 arose from trying to understand exactly what the spec required in terms of what is a syntactically valid pollData string.
2. My program is designed around 5 functions:

makeUpper()

repeatedly:

capitalize current character

stop after last character

hasProperSyntax()

check for empty string

capitalize entire string

repeatedly:

break string into substrings with different state codes

repeatedly:

store non-comma characters in a string

if comma stop

check if string is valid state forecast (calls function below)

check if last character is comma

if all substrings are valid then entire string is valid

validStateForecastSyntax()

calls function below to check if first two characters are valid state codes

checks if string is only state code

repeatedly:

store digits in a string

make sure no more than two digits are allowed per party

looks for party character immediately after last digit

makes sure there is at least one digit in between party characters

validates no characters following state code as true ex: "NY"

isValidUppercaseStateCode()

saves uppercase state codes in constant string

checks if input string is 2 characters long and if it matches any state codes in constant string

tallySeats()

checks for proper syntax of pollData string

checks if party is a letter

if neither of the above, set seatTally to 0

checks for empty string or only state code as pollData string

capitalizes pollData to call function above

capitalizes party letter to match implementation of hasProperSyntax (where every char is capitalized)

repeatedly:

finds party letter(s) in pollData string

checks if 1 or 2 digits

adds digits to seatTally

Due to the design of my program, both hasProperSyntax and tallySeats call the makeUpper function to make use of the provided isValidUppercaseStateCode function. hasProperSyntax also calls ValidStateForecast after calling isValidUppercaseStateCode to validate the substrings of pollData that exist between commas. tallySeats also calls hasProperSyntax to determine which value to return.

1. Some of the test data I used to test my program are included below:

assert(hasProperSyntax("CT5D,NY9R17D1I,VT,ne3r00D"));

Should return true because valid string (even with no state forecast for Vermont)

assert(hasProperSyntax(""));

Should return true because empty string

assert(!hasProperSyntax(" "));

Should return not false because string only contains whitespace

assert(!hasProperSyntax(",CT5D,NY9R17D1I,VT,ne3r00D"));

Should return not false because of comma at the start of the string

assert(!hasProperSyntax("CT5D,NY9R17D1I,VT,ne3r00D,"));

Should return not false because of comma at the end of the string

assert(!hasProperSyntax("CT5D,NY9 R17D1I,VT,ne3r0 0D"));

Should return not false because of spaces between digits

assert(!hasProperSyntax("CT5D,NY9 R17D1I,VT,ne3r00 D"));

Should return not false because of spaces between digits and party

assert(!hasProperSyntax("NY9R17D1I, Vt, NJ3d5r4D, KS4R"));

Should return not false because of spaces after commas

assert(!hasProperSyntax("TY9R17D1I,Vt,NJ3d5r4D,KS4R"));

Should return not false because of invalid state code

assert(!hasProperSyntax("NY922R17D1I,Vt,NJ3d5r4D,KS4R"));

Should return not false because of 3 digit party votes for NY (922R)

assert(hasProperSyntax("NY"));

Should return true because only party code

assert(!hasProperSyntax("CA32"));

Should return not false because no party following digit(s)

assert(!hasProperSyntax("NYd"));

Should return not false because no digits preceding party.

int seats = -999;

assert(tallySeats("CT5d,NY9R17D1I,VT,ne3r00d", 'd', seats) == 0 && seats == 22);

Should return true because valid string and party

seats = -999;

assert(tallySeats("CT5D,NY9R17D1I,VT,ne3r00D", '%', seats) == 2 && seats == -999);

Should return true because party is not a letter, so seats should be unchanged

assert(tallySeats("CA32", 'D', seats) == 1 && seats == -999);

Should return true because invalid pollData string

assert(tallySeats("CA", 'D', seats) == 0 && seats == 0);

Should return true because valid string and party but no change from 0

seats = -999;

assert(tallySeats("", 'D', seats) == 0 && seats == 0);

Should return true because empty string is valid and party is valid

assert(hasProperSyntax("MA9D,KS4R") && hasProperSyntax("KS4R,MA9D"));

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

Should return true for both because program interprets ‘,’ as single separator.