Ryan Lam

A. One notable obstacle I overcame was ensuring that my program did not try to access an index of a string that was out of bounds. Since C++ does not check whether a given string index is out of bounds, my program relied on undefined behavior, appearing to work correctly until I tested it on the SEASnet server, which revealed the out-of-bounds errors. Another obstacle I overcame was designing an algorithm to correctly parse through a poll data string and check the state codes, party results, and tally up seats. I encountered many bugs where my program would skip over indexes in the string.

B. My hasProperSyntax function calls two other functions, processStateCode and processPartyResults.

processStateCode:

take in one state forecast

create a new string

append uppercase of first 2 letters in state code

if the state code is invalid:

return false

return true

processPartyResults:

take in one state forecast

create a substring of the forecast consisting of everything after the state code

if the resulting substring is empty:

the party result is valid

if the resulting substring only has size of one:

the party result is invalid

repeatedly until the end of the string is reached:

if the current character is not a number:

check if there are two or less numbers before the character

check if the character is a letter

remove the checked party result from the state forecast

hasProperSyntax:

take in a poll data string

if poll data is empty

the syntax is valid

repeatedly until the end of string is reached:

segment the string into sections between commas

check the section’s state code using processStateCode

check the section’s party result using processPartyResults

tallySeats:

take in a poll data string, a party character, and a seat tally integer reference

if the poll data contains no state forecasts:

set number of seats to 0 and end function

if poll data syntax is incorrect:

end function and return 1

if the party character is not a letter:  
 end function and return 2

repeatedly until end of poll data string is reached:

segment string into sections between commas

repeatedly until end of section is reached:

if current index contains desired party letter:  
 add the party result before the letter to the seat tally

C. Test data

{“”, ‘d’, 0} : empty poll data string

{“CA”, ‘d’, 0} : poll data string with one state code

{",ca,ny,ma",'x',0} : checks if hasProperSyntax returns false if poll data starts with comma

{",ca,ny,ma",'x',0} : checks if tallySeats returns 1 if poll data starts with comma

{“ca,ny,ma,” ,’d’ ,0} : poll data ends with comma

{"ca,,ny,ma",'x',0} : checks if hasProperSyntax returns false if poll data contains multiple consecutive commas

{"ca,,ny,ma",'x',0} : checks if tallySeats returns 1 if poll data contains multiple consecutive commas

{"ca,ny,,,ma",'x',0} : multiple commas in middle of poll data

{“ca9i,ny7d, ct8r”, ‘d’, 0} : poll data contains a space

{“ca9i7r8d”, ‘%’, 1} : invalid party code character

{“ca9i7rd”, ‘r’, 0} : no party results between two party codes

{“ca9i77777”, ‘I’, 0} : poll data does not end with a party code

{“ca2r17d,ny9d,co,wa13d17i”, ‘D’, -999} : uppercase party code

{“ca2r17D,ny9D,co,wa13D17i”, ‘d’, -999} : lowercase party code

{“ca2r17d,ny9D,co,wa13d17i”, ‘d’, -999} : mixed upper/lowercase party codes

{“NY7i6R”, ‘R’, 0} : uppercase state code

{“ny7i6R”, ‘R’, 0} : lowercase state code

{“Ny7i6R”, ‘R’, 0} : upper then lowercase state code

{“nY7i6R”, ‘R’, 0} : lower then uppercase state code

{“orange”, ‘I’, 0} : random string in poll data

{“7”, ‘d’, 0} : single number string in poll data

{“$$$”, ‘R’, 0} : random symbol in poll data

{“California”, ‘R’, 0} : invalid poll data string that starts with a state code

{“AZ997D,Tx34D”, ‘D’, 0} : poll data string has more than two consecutive numbers

{“CA,az,tx,nY”, ‘r’, -999} : checks if seatTally is set to zero

{“CA5D,az7R,tx7D”, ‘j’, -999} : party code not contained in poll data