Douglas Frattini Edwards

Professor David Smallberg

COM SCI 31

26 October 2022

**A.) Notable Obstacles**

Fortunately, creating this program did not cause too many significant problems. The implementation of each idea went fairly smoothly and no bugs caused any more than 5-10min of investigation to resolve. I would say the only significant obstacle was testing and thinking of all the different ways in which a poll string could be invalid, and ensuring that isValidPollString caught them. Catching these problems with code was easy, but originally imagining the specific issue took a very significant amount of time. Some of the problems that I only realized late in the project were as follows:

* A state code with a letter matching the selected party could in some cases cause an out of range index
* A comma being the first character of the pollstring had not been accounted for
* An empty poll string was not treated as a valid poll string

**B.) Design**

**isValidPollString()**

My design of isValidPollString centered around mapping out all comma locations. While recording the location of each comma in an std vector, checks were done to make sure commas were not the first or last characters of the string, and also that commas were far apart from each other and the start/end of the pollstring so that valid state info could be stored between them. I also took this opportunity to capitalize every character and ensure every character is alphanumeric or a comma by using a helper function I built called isValidChar. Once the comma map was complete, I looped through the list of commas, extracting the data on each side and sending them to helper functions.

The first helper function was the one provided, and it was used to make sure that a valid state code was the beginning of every subsection of data. I created the second function (isValidStateResult), which takes in the results (everything between the state code and the comma). It then parses through, returning true if empty, and returning false if it fails to meet any of the following criteria:

* Starts with number
* Ends with letter
* No more than 2 numbers or 1 letter in a row

If the poll string passes all of these tests, it is considered a valid poll string.

**countSeats()**

Comparatively, countSeats() was much simpler to create. First it checks for invalid poll strings using the above function and ensures that the selected party is a valid character. I realized that almost all data including states and commas could be entirely ignored by just instructing the function to loop through the poll string in search of any instance of the party character. Whenever it finds this character, it checks to see if 1 or 2 characters behind it are digits. If 2, then it creates a substring of them, converts them to an int, and adds them to s. If 1, it simply converts that character to an int and adds it to s.

**C.) Test Data**

| **Case** | **Description / Why Important?** | **Return value** | **Value of s** |
| --- | --- | --- | --- |
| countSeats("CA01D,DE23RI56D", 'D', s) | Ensure that state codes with letters matching the selected party do not cause issues | 0 | 57 |
| countSeats("DE01D,CA23RI56D", 'D', s) | State codes matching party letters don’t cause out of range indexing if at start of poll string | 0 | 57 |
| countSeats("dE01D,Ca23RI56d", 'd', s) | Behavior unaffected by capitalization | 0 | 57 |
| countSeats("dE01D,Ca23RI56d,", 'd', s) | Incorrect syntax: comma at end | 1 | 0 |
| countSeats("dE01D,,Ca23RI56d", 'd', s) | Incorrect syntax: commas with nothing between | 1 | 0 |
| countSeats(",dE01D,Ca23RI56d", 'd', s) | Incorrect syntax: Comma at start | 1 | 0 |
| countSeats("SC1R2D5R,VT9R5R", 'R', s) | Repeated party results for a state | 0 | 20 |
| countSeats("XE1R2D5R,VT9R5R", 'R', s) | Invalid state | 1 | 0 |
| countSeats("SC1R-2D5R,VT9R5R", 'R', s) | Invalid character in pollstring | 1 | 0 |
| countSeats("state56,85republican", 'R', s) | Multiple problems handled | 1 | 0 |
| countSeats("SCR2D,VT9R5R", 'R', s) | No number before party in result | 1 | 0 |
| countSeats("SC1R2D,VT9R5R", '@', s) | Non-alphabetical party char | 2 | 0 |
| countSeats("", 'D', s) | Empty poll string must work | 0 | 0 |
| countSeats("VT,CA,DE,CA,NY5D", 'D', s) | Must handle state codes with no party results attached | 0 | 5 |