Team 18

Noreen Si, Lane Enget, Pyrenees Gavois, Jonathan Haak

Pbi: Display an IR table that records the vote totals, differences per round in an IR election
Task Description (from Sprint Log): Write code to implement a simple table that can hold IR round results

Unique Testing Number: IR_Table_Tests -> test_tableDisplay_oneRound(), test_tableDisplay(), test_tableDisplay_mutipleEmptyCells(), test_tableIntegration_1()

Team Member(s) Responsible: Noreen Si, Lane Enget

Inputs:

1. A list of lists of each candidate's vote differences and totals per round as a string, including the candidate name, party at the beginning of each list
2. A list of exhausted ballots totals per round
3. A list of the totals for the first round
4. A number representing the total number of rounds in a sample election
5. A FileReader that represents an election voting file such as ballots.csv

bet

Tests:

1. Test for input where there is only one round run in an election
2. Test for inputs where there are multiple rounds in an election
3. Test for inputs where there will be empty cells in the rows, such as there being eliminated candidates before the election runs to completion
4. Test for an election with multiple eliminations, rounds, and candidates

Outputs: A GUI that shows up on the screen that has the correct number of columns (depending on the input for total number of rounds) with properly formatted candidate information where each candidate has the vote difference per round in the correct column as well as the exhausted pile and totals being recorded in their own separate rows

Passed or Failed: Passed

Date: 4/18/2023

Pbi: Calculate votes added, subtracted with IR per round with exhausted ballots
Task Description (from Sprint Log): Write code to keep track of IR votes difference per candidate and exhausted ballots

Unique Testing Number: IR_Vote_Tracking_Tests

Team Member(s) Responsible: Noreen Si

Inputs:

1. A FileReader that represents the csv testing file to use for a test -> "IRTrackExhaustedVotes.csv", ""IRMajority1.csv", "IRConductAlgorithm2.csv"
2. A String[] of candidate names, parties for IR

Tests:

1. Test for proper initialization of a candidate's list of vote updates per round, such as having their name, party be specified
2. Test for inputs where there is only a single round with no eliminations
3. Test for inputs where there are multiple rounds with eliminations
4. Test for proper tracking of exhausted ballots across different rounds, with the exhausted totals staying consistent/correctly increasing with each round
5. Test for proper tracking of the total at the beginning of the election

Outputs: Each value within each list of each candidate per round matches with the proper vote difference and total across all the rounds. For instance, the tests check that within the first round position of a candidate's vote differences list is the correct addition/subtraction of votes (if eliminated). The tests also assert that there are properly tracked totals and tracked exhausted ballots from round to round.

Passed or Failed: Passed

Date: 4/27/2023

Pbi: PO to display the winners and losers (the outcome) of a PO election
Task Description (from Sprint Log): Write code to see if PO header was read
Unique Testing Number: test_parseHeader()

Team Member(s) Responsible: Pyrenees Gavois

Inputs:

1. A FileReader that represents an election voting file
2. TestPO.csv, which is a pretty standar PO formatted file

Tests:

1. Tests to see if the parseHeader function works, that is testing to see if the candidates were read in correctly, as well as the number of ballots and number of candidates. Individual ballots not read yet

Outputs: The three candidates, Park, Forst, and Borg at stored in the PO object that was

just created, and the number of total votes which is 10 is also stored
Passed or Failed: Passed
Date: 4/27/2023

Pbi: PO to display the winners and losers (the outcome) of a PO election **Task Description (from Sprint Log):** Write code to see if PO file was processed **Unique Testing Number:** test_processFile()
Team Member(s) Responsible: Pyrenees Gavois
Inputs: 1. A FileReader that represents an election voting file 2. TestPO.csv, which is a pretty standar PO formatted file
Tests: 1. Tests to see if the processFile function works. Sees if all ballots following the header information are properly incrementing the vote count for each candidate. parseHeader must work before this is to be executed.
Outputs: An internal array representing the amount of votes each candidate has earned by the end of processing the entire file. In this particular case, the array's values should first be 3, then 3, then 4.
Passed or Failed: Pass
Date: 4/27/2023

Pbi: PO to break ties in the election fairly among those with the most votes **Task Description (from Sprint Log):** Write code to break tie in a PO election **Unique Testing Number:** test_twoTie(), test_multiTie(), test_allTie()
Team Member(s) Responsible: Pyrenees Gavois
Inputs: 1. A FileReader that represents an election voting file 2. TwoTiePO.csv, which is a particular PO scenario where two candidates are tied for winning 3. MultiTiePO.csv, which is a particular scenario where many candidates are tied for

winning
4. AllTiePO.csv, which is a particular scenario where all candidates are tied for winning

Tests:
1. Tests to see if the two winners are stored in a winners array, and then to see if the final winner is indeed selected from that pool
2. Tests to see if the many winners are stored in a winners array, and then to see if the final winner is indeed selected from that pool
3. Tests to see if all winners are in the winners array
4. Note, everything above is based on the assumption that breakTie function works, which is should based on Part1 of the project

Outputs:
For the first test, the winners array should contain Foster and Bourg
For the second, the winners array should contain Pike, Foster and Bourg (and NOT Rob)
For the last one, the winners array should also contain Pike, Foster and Bourg

Passed or Failed: Passed

Date: 4/27/2023

Pbi: PO to display the winners and losers (the outcome) of a PO election
Task Description (from Sprint Log): Write code to see if PO algorithm was conducted
Unique Testing Number: test_conductAlgorithm(), test_testBasic(), test_superSmall()

Team Member(s) Responsible: Pyrenees Gavois

Inputs:
1. A FileReader that represents an election voting file
2. TestPO.csv, just your average generic PO file
3. SuperSmallPO.csv, PO file that has only one vote
4. Basic.csv, another, slightly larger generic PO file

Tests:
1. Tests to see if the winner can be correctly in a variety of rather simple scenario. One scenario is a miniscule edge case, the other two are a lot more normal

Outputs:
The first csv file should yield Bourg as the winner
The second csv file should yield Pike as the winner
The third should also yield Pike as the winner

Passed or Failed: Passed

Date: 4/27/2023

Pbi: System to be able to be passed in multiple election files
Task Description (from Sprint Log): Write code that allows to pass multiple files and execute them
Unique Testing Number: pass_multipleIR_files()

Team Member(s) Responsible: Jonathan Haak

Inputs:

For manual tests many different combinations of IR testing files were used, for automated tests the ArrayList of files "IRTestBallot.csv", "IRTestBallot2.csv", "IRTestBallot3.csv", "IRTestBallot4.csv", "IRTestBallot5.csv", "IRTestBallot6.csv" was used.

Tests:

Tests the general ability of the system to run multiple IR files and caculate the results correctly, involves testing full functionality of ability to take IR files in, process IR headers, process IR files, and conduct IR algorithm.

Outputs:
The IR test should yield Candidate B as the winner for automated tests, for manual tests various outputs were produced and verified as correct

Passed or Failed: Passed

Date: 4/30/2023

Pbi: System to be able to be passed in multiple election files
Task Description (from Sprint Log): Write code that allows to pass multiple files and execute them
Unique Testing Number: pass_multipleCPL_files()

Team Member(s) Responsible: Jonathan Haak

Inputs:
Many different combinations of files were used for manual tests, for automated tests the ArrayList of files "CPLTestBallot.csv", "CPLTestBallot2.csv", "CPLTestBallot3.csv", "CPLTestBallot4.csv", "CPLTestBallot5.csv" were used

Tests: Tests the general ability of the system to run multiple CPL files and caculate the results correctly, involves testing full functionality of ability to take CPL files in, process CPL headers, process CPL files, and conduct CPL algorithm
Outputs: For manual tests results were verified to be correct, for automated tests the results are technically random, since some of the seats are distributed via lottery.
Passed or Failed: Passed
Date: 4/30/2023

Pbi: System to be able to be passed in multiple election files **Task Description (from Sprint Log):** Testing that System can handle multiple PO files **Unique Testing Number:** pass_multiplePO_files()
Team Member(s) Responsible: Jonathan Haak
Inputs: For manual tests, various combinations of different PO test files. For Automated tests, input was a ArrayList of Strings containing "POTestBallot1.csv", "POTestBallot2.csv", "POTestBallot3.csv", "POTestBallot4.csv", "POTestBallot5.csv"
Tests: Tests the general ability of the system to run multiple PO files and caculate the results correctly, involves testing full functionality of ability to take PO files in, process PO headers, process PO files, and conduct PO algorithm.
Outputs: For the automated tests, PO test should yield Candidate A as the winner, for manual tests results were verified by hand
Passed or Failed: Passed
Date: 4/30/2023