Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x

Test Date: 11/1/2023

Test Case ID#: 10

Name(s) of Testers: Logan Watters

Test Description:
An IRElection where no majority is found after all ballots have been redistributed.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using all IR related methods (i.e., anything in a Java file with IR in the name).

Automated: yes___ no__✔__

Results: Pass __X__ Fail________

Preconditions for Test: Election file contains ballots and will lead to a tie in the middle of the election.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Pass in filename.	“electionTest1.csv”	File is opened.	Expected	
2	Create an Election object.	electionTest1.csv	Election object is created.	Expected	
3	Create a HeaderProcessor object	electionTest1.csv	HeaderProcessor object is created.	Expected	
4	Parse the header.	HeaderProcessor object.	Returns an IR voting system with each candidate from the emptyelection.csv file.	Expected	
5	Run election on the voting system	Nothing	If Rosen is removed from the election, then Chou or Kleinberg wins, depending on who wins the final coin flip. If Kleinberg is removed from the election, then Rosen wins. If Chou is removed from the election, then Kleinberg wins.	Expected	

Post condition(s) for Test: If Rosen is removed from the election, then Chou wins. If Kleinberg is removed from the election, then Rosen wins. If Chou is removed from the election, then Kleinberg wins.

Project Name: Project 1: Voting System

Team#5

Test Stage: **Unit __** **System X**

Test Case ID#: 11

Test Description:

An IR Election where no majority is found after all ballots have been redistributed so the candidate with the most ballots is selected as the winner.

Automated: **yes___** **no__✔__**

Test Date:

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using giveBallot().

Results: **Pass __X__** **Fail________**

Preconditions for Test: Election file contains ballots and after all redistributions there is no majority winner.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Pass in filename.	“electionTest3.csv”	File is opened.	Expected	
2	Create an Election object.	electionTest3.csv	Election object is created.	Expected	
3	Create a HeaderProcessor object	electionTest3.csv	HeaderProcessor object is created.	Expected	
4	Parse the header.	HeaderProcessor object.	Returns an IR voting system with each candidate from the emptyelection.csv file.	Expected	
5	Run election on the voting system	Nothing	Rosen is declared the winner with 4/10 votes.	Expected	

Post condition(s) for Test: Rosen is declared the winner.

Project Name: Project 1: Voting System Team#5

Test Stage: **Unit __** **System _x_**

Test Case ID#: 12

Test Description:

Test the OPL system with 0 ballots.

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Election information is stored in BallotRedistributeTester.csv. Election, HeaderProcessor, FileProcessor, and IRVotingSystem.

Automated: yes___ no__✔__

Results: Pass Fail X

Preconditions for Test: No ballots are given to the FileProcessor.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Read file header	“electionTest4.csv”	Reads 0 entries.	Expected	
2	System has no ballots to read in.	Election file	Error.	Expected	
3	VotingSystem has no ballots to loop through for the candidates and displays error.	Terminal Output	0 ballots are to be reassigned to candidates and therefore there is no reordering and no winner.	Divide by 0 error in runElection function because there are no ballots to be read	
4					

Post condition(s) for Test: System alerts there are 0 ballots.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System _x_

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Test Case ID#: 13

Test Description:
Test the OPL system with actual ballots.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Election information is stored in BallotRedistributeTester.csv. Election, HeaderProcessor, FileProcessor, and IRVotingSystem.

Automated: yes___ no__✔__

Results: Pass X Fail

Preconditions for Test: Ballots have been created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Read file header	“electionTest5.csv”	Reads in the header correctly. Then creates the candidate objects and votingSystem.	Expected	
2	System reads through ballots and parses through candidates	Election file	Ballots are assigned to their candidate. runElection() is called to begin votingSystem	Expected	
3	Candidate with lowest amount of ballots is removed (audit file notes this)	AuditFile	Lowest candidate removed	Expected	
4	Loops and next candidate is removed until there is one candidate left	Audit File	Audit file displays candidates removed and shows where votes are now located.	Expected.	
	VotingSystem displays winner in auditFile	Terminal Output	Candidate with the most votes is declared winner after looping through and assigning seats.	Expected	

Post condition(s) for Test: System assigns ballots to candidates and loops through (dropping the candidate with the lowest amount of ballots each round) until there is one declared winner.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System _x_

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Test Case ID#: 14

Test Description:

An IRElection where two candidates are tied at 50% votes each after two candidates are removed.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Election information is stored in BallotRedistributeTester.csv. Election, HeaderProcessor, FileProcessor, and IRVotingSystem.

Automated: yes no ✔

Results: Pass __✔__ Fail________

Preconditions for Test: .csv file is available to the program.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Read file header	“electionTest6.csv”	Reads reads header properly, candidate objects and votingSystem are created.	Expected	
2	System reads through ballots and parses through candidates	Election file	Ballots are assigned to the correct candidate and votingSystem calls runElection().	Expected	
3	VotingSystem removes the candidate with the lowest ballots and logs in audit log	AuditFile	The candidate removed is stated in the audit log	Expected	
4	VotingSystem removes the next candidate with the lowest ballots and logs in audit log	AuditFile	The candidate removed is stated in the audit log and their votes are successfully reallocated.	Expected	
5	VotingSystem uses breakTie() and each candidate remaining wins an equal amount of times	Terminal output	50% of the time candidate 1 is winning, 50% of the time candidate 2 is winning	When run manually 10 times, candidate 1 was selected 6 times and candidate 2 was selected 4 times.	This result is satisfactory in terms of equally picking both candidates (breakTie() was also run 1000 times with results in unit testing)

Post condition(s) for Test: Candidate 1 and candidate 2 are winning an equal amount of elections.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __	**Test Date: 11/12/2023**
Test Case ID#: 15 **Test Description:** Test FileProcessor's ability to return an empty list if no ballots are in the election file.	**Name(s) of Testers:** Logan Watters
Automated: yes___ no__✔__	**Indicate where are you storing the tests (what file) and the name of the method/functions being used:** Storing in test CSV and using processFile().
Results: Pass X Fail	

Preconditions for Test: The election file does not contain any ballots.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Pass in filename.	"emptyelection.csv"	File is opened.	Expected	
2	Create a FileProcessor object.	emptyelection.csv	FileProcessor is created.	Expected	
3	Run the FileProcessor's processFile() method.	emptyelection.csv	Empty list is returned	Expected	

Post condition(s) for Test: File is opened and FileProcessor gives an output of an empty list.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __	**Test Date: 11/12/23**
Test Case ID#: 16 **Test Description:** Test giveBallot's ability to fail to give a ballot to a	**Name(s) of Testers:** Logan Watters

candidate that is not in the running.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using giveBallot().

Automated: **yes** **no** ✔

Results: **Pass ___x_** **Fail________**

Preconditions for Test: IRCandidate has been created and then removed from the election using the removeCandidate() method.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run giveBallot() on the candidate.	Arbitrary IRBallot object.	Ballot is not given to the candidate.	Expected	
2	Run getBallotCount() on the candidate.	N/A	0 is returned.	Expected	

Post condition(s) for Test: Ballot count for the candidate is 0.

Project Name: Project 1: Voting System Team#5

Test Stage: **Unit x** **System __**

Test Date: **11/12/2023**

Test Case ID#: **17**

Name(s) of Testers: Logan Watters

Test Description:
Test FileProcessor’s ability to return an empty list if no election type is specified at the top of the election file.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using processFile().

Automated: **yes___** **no__✔__**

Results: Pass __x__ Fail________

Preconditions for Test: The election file does not contain any ballots.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Pass in filename.	"nooplorir.csv"	File is opened.	Expected	
2	Create a FileProcessor object.	nooplorir.csv	FileProcessor is created.	Expected	
3	Run the FileProcessor's processFile() method.	nooplorir.csv	Empty list is returned	Expected	

Post condition(s) for Test: File is opened and FileProcessor gives an output of an empty list.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Test Case ID#: 18

Test Description:
Test the OPL system so that candidate ballots are accurately counted and eliminated in each round.

Indicate where are you storing the tests (what file) and the name of the method/functions being used:
Storing in test CSV and using processBallots().

Automated: yes___ no__✔__

Results: Pass ___✔__ Fail________

Preconditions for Test: The election file contains ballots and candidates are out of order to be reordered.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Read file header.	Election File	Empty list of ballots is returned.	expected	
2	System reads through ballots and parses through candidate	Election File	Ballots are assigned to their candidate. runElection() is called to begin votingSystem	Expected	
3	Ballots are tallied for candidates and lowest candidate is removed.	Election File.	Candidates receive their votes and Lowest candidate removed.	Expected	
3	Candidates reordered and loops until one candidate is left. Documents in audit file	Audit File.	Audit file displays candidates removed and shows where votes are now located.	Expected	
4					

Post condition(s) for Test: Ballots are accounted for, candidates are ordered based on voting and one is removed.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __

Test Date: 11/1/2023

Test Case ID#: 19

Name(s) of Testers: Logan Watters

Test Description: Test giveBallot()’s ability to give a ballot to the specified candidate.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using giveBallot(), IRCandidate constructor, and IRBallot constructor.

Automated: yes___ no__✔__

Results: Pass _X___ Fail________

Preconditions for Test: Ron Johnson’s ballot count is 0

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create an ArrayList of IRCandidate objects.	Name: “Ron Johnson” Party: “G” Ballot Index: “1” Name: “Jon Rohnson” Party: “R” Ballot Index: “4” Name: “Candice Tyler” Party: “D” Ballot Index: “3” Name: “Audrey Owens” Party: “I” Ballot Index: “2”	IRCandidate ArrayList is created and instantiated with the IRCandidates as described in the test data.	Expected	
2	Create an IRBallot	ArrayList of IRCandidate objects aforementioned.	IRBallot object is created. giveBallot is called on the first candidate in the list (i.e., “Ron Johnson”) with itself as the argument.	Expected	
3	Give ballot to first candidate	IRCandidate object for “Ron Johnson”.	“Ron Johnson”’s ballot ArrayList contains the ballot created above and “Ron Johnson”’s total number of ballots increments to 1.	Expected	

Post condition(s) for Test:
Ron Johnson has been given one ballot and the total ballot count for Ron Johnson is equal to 1.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __

Test Case ID#: 20

Test Description: Test removeCandidate()'s ability to redistribute each ballot currently allocated to the candidate.

Automated: yes___ no__✔__

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using giveBallot(), IRCandidate constructor, and IRBallot constructor.

Results: Pass _X___ Fail________

Preconditions for Test: None

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create an ArrayList of IRCandidate objects.	Name: "Ron Johnson" Party: "G" Ballot Index: "1" Name: "Jon Rohnson" Party: "R" Ballot Index: "4" Name: "Candice Tyler" Party: "D" Ballot Index: "3" Name: "Audrey Owens" Party: "I" Ballot Index: "2"	IRCandidate ArrayList is created and instantiated with the IRCandidates as described in the test dated.	Expected	
2	Create an IRBallot	ArrayList of IRCandidate objects aforementioned.	IRBallot object is created. giveBallot is called on the first candidate in the list (i.e., "Ron Johnson") with itself as the argument.	Expected	
3	Give ballot to first candidate	IRCandidate object for "Ron Johnson".	"Ron Johnson"'s ballot ArrayList contains the ballot created above and "Ron Johnson"'s total number of ballots increments to 1.	Expected	
4	Remove "Ron Johnson" from the race.	IRCandidate object for "Ron Johsnon"	"Ron Johnson"'s ballot ArrayList contains no ballots and is empty. Ron Johnson's total number of ballots is equal to 0. Ron Johnson's 'inRunning' value is equal to false. "Jon Rohnson"'s ArrayList contains the ballot previously held by Ron	Expected	

			Johnson. “Jon Rohnson”’s total number of ballots is equal to 1.		

Post condition(s) for Test:
Ron Johnson does not have any ballots in the ArrayList. Jon Rohnson has 1 ballot in the ArrayList.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit x System __

Test Date: 11/1/2023

Name(s) of Testers: Perrie Gryniewicz, Logan Watters, Matthew Johnson, Bek Allenson

Test Case ID#: 003

Test Description: Test breakTie in IRVotingSystem to make
sure it is. Runs 1000 times and tracks how many times each
position wins.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Automated Test case is in IRVotingSystemTest.java. IRVotingSystem() and breakTie(int numOfCandidate) are used.

Automated: yes_✔_ no____

Results: Pass __X__ Fail________

Preconditions for Test: numOfCandidate != 1 or 0.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a IRVotingSystem object with dummy values	n/a			
2	Call breakTie(2)	num1(number of times spot 1 wins) num2(number of times spot 2 wins)	0.44 < num1 < 0.56 0.44 < num2 < 0.56		
3	Call breakTie(3)	num1(number of times sport 1 wins) num2 (number of times spot 2 wins) num3(number of times spot 3 wins)	0.27 < num1 < 0.39 0.27 < num2 < 0.39 0.27 < num3 < 0.39		

Post condition(s) for Test:
The rate at which each spot is picked is not statistically different from any other spot

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x **Test Date: 11/1/2023**

Test Case ID#: 21 **Name(s) of Testers:** Matthew Johnson

Test Description:
A MPO election with 6 candidates and two seats, There is a tie for the second seat

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using all MPO related methods (i.e., anything in a Java file with MPO in the name).

Automated: yes___ no__✔__

Results: Pass X Fail

Preconditions for Test: None

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile				
2	Pass in “testing/MPOtest1.csv”	MPOtest1.csv	MPO election is run. Pike wins the first seat with 3 votes, and either Borg or Foster win the second seat with 2 votes.	Expected	

Post condition(s) for Test: Pike wins the first seat, either Borg or Foster win the second seat.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x **Test Date: 11/1/2023**

Test Case ID#: 22 **Name(s) of Testers:** Matthew Johnson

Test Description:
A MPO election with 6 candidates and three seats,
There is a tie for second place.

Indicate where are you storing the tests (what file) and the name of the method/functions being used:
Storing in test CSV and using all MPO related methods (i.e., anything in a Java file with MPO in the name).

Automated: yes___ no__✔__

Results: Pass __X__ Fail________

Preconditions for Test: None

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile				
2	Pass in "testing/MPOtest2.csv"	MPOtest2.csv	MPO election is run. Pike wins the first seat with 3 votes, and either Borg or Foster win the second seat with 2 votes, then the other wins the third seat with 2 votes.	Expected	

Post condition(s) for Test: Pike wins the first seat with 3 votes, then either Borg or Foster win the second seat with 2 votes. If Borg wins the second seat then Foster wins the third seat, if Foster wins the second seat then Borg wins the third seat.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x **Test Date: 11/1/2023**

Test Case ID#: 23 **Name(s) of Testers:** Matthew Johnson

Test Description:
A MPO election with 6 candidates and one seat,
There is a tie.

Indicate where are you storing the tests (what file) and the name of the method/functions being used:

Automated: yes___ no__✔__

Storing in test CSV and using all MPO related methods (i.e., anything in a Java file with MPO in the name).

Results: Pass __X__ Fail________

Preconditions for Test: None

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile				
2	Pass in "testing/MPOtest3.csv"	MPOtest3.csv	MPO election is run. Either Pike or Smith win the seat with 4 votes.	Expected	

Post condition(s) for Test: Pike wins 50% of the time with 4 votes, Smith wins the other 50% of the time with 4 votes.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x

Test Date: 11/1/2023

Test Case ID#: 26

Name(s) of Testers: Bek Allenson

Test Description:
An MPO file with 2 candidates and 6 ballots.

Indicate where are you storing the tests (what file) and the name of the method/functions being used:
Storing in test CSV and using all MPO related methods (i.e., anything in a Java file with MPO in the name).

Automated: yes__X__ no_____

Results: Pass X Fail

Preconditions for Test: None

Step	Test Step	Test Data	Expected Result	Actual Result	Notes

#	Description				
1	Compile				
2	Pass in "testing/testHeaderMPO.csv"	MPOParsingTest.csv	MPOVotingSystem is created with candidates Pike (D) and Deutsch ®, expecting 6 ballots and 1 candidate to be seated.	Expected	

Post condition(s) for Test: MPOVotingSystem is ready to process the ballots from the file.

Project Name: Project 1: Voting System Team#5

Test Stage: Unit __ System x

Test Date: 11/1/2023

Test Case ID#: 27

Name(s) of Testers: Bek Allenson

Test Description:
An MPO file with 2 candidates and 2 ballots.

Indicate where are you storing the tests (what file) and the name of the method/functions being used: Storing in test CSV and using all MPO related methods (i.e., anything in a Java file with MPO in the name).

Automated: yes X no

Results: Pass X Fail

Preconditions for Test: None

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile				
2	Pass in "testing/testBallotsMPO.csv"	MPOParsingTest.csv	Pike (D) has 2 votes and Deutsch (R) has 0	Expected	

Post condition(s) for Test: MPOVotingSystem is ready to run the election and determine the winner(s).

PBI	MPO Tie Breaking Functionality
Team Member	Logan
Inputs	A file representing an MPO election
Tests	1. An MPO election where there is only one seat available and there is a tie for that seat. This should be run 1000 times to ensure that it is randomly decided. 2. An MPO election where there are two seats available and there is a tie for the second seat. This should be run 1000 times to ensure that it is randomly decided. 3. An MPO election where there is are three seats available and there is a tie for both seats (i.e. a 3 way tie for two seats). This should be run 1000 times to ensure that it is randomly decided.
Outputs	1. True or false if the winners are equally distributed.
Passed or Failed	All passed. Can run these; they are MPOTieTest1, MPOTieTest2, and MPOTieTest3 in the testing folder.
Date	12/10/2023

PBI	OPL runs without any errors
Team Member	Logan
Inputs	A file representing an OPL election
Tests	1. Test to see if there is a divide by 0 error when there are no ballots in an OPL election
Outputs	1. “There are no ballots in this election”
Passed or Failed	Passed. Can run this by running main with filename OPLEmpty.csv
Date	12/10/2023

System integration tests were just running the previous tests to ensure that the addition of MPO did not break IR and OPL. These all passed again.