Project Name: Project 1: IR/OPL Voting System   Team#24

Test Stage: System

Test Date: March 14th, 2021

Test Case ID#: SUT1

Name(s) of Testers: Sai Tallapragada

Test Description:

In this Test, We simply just testing if the Program is able to understand that we are handling an IR Elections and if the program is able generate the correct output.

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

We are manually compiling the program and running it and passing in our test csv file called IRTest3.csv. This file is located in the src folder with the rest of the other files.

Automated: yes___   no __X_

Results: Pass __X__ Fail________

Preconditions for Test: The Main Program needs to be compiled and it needs to start running. You will be asked for a user input at that point for the test file IRTest3.csv

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Compile Main Program javac Main.java				
2	Run the main() in Main CLass where you will prompted for User Input java Main				
3	Input the file named IRTest3.csv	IRTest3.csv	It should display in the terminal screen “Chou(I) has won the election.”	“Chou(I) has won the election.”	This testfile was supposed to just run the Ir ELection and the test data was written in a way where

Post condition(s) for Test:

After we put in the file and the program finished running, it should clearly display on the screen who won the IR election correctly, it should output information into a media file and audit file which show the election process.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: System

Test Date: March 14th/2021

Test Case ID#: SUT2

Name(s) of Testers: Sai Tallapragada

Test Description:

For OPL ELection, We are going to be just passing in a sample election file called OPLTest1.csv and we will be see if it correctly outputs the results

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

We are storing the OPLTest1.csv in the testing folder which is located inside the Project1 folder of our repo.

Automated: yes___ no _X__

Results: Pass __X__ Fail________

Preconditions for Test: We need to compile the program with javc Main.java and run program with java Main. Please make sure to put in the file path OPLTest1.csv when the program asks you to.

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Compile Main Program javac Main.java				

2	Run the main() in Main CLass where you will prompted for User Input java Main				
3	Input the file named OPLTest1.csv	OPLTest1.csv	Party, Seats Won, Candidates: D , 2 , Pike , Foster R , 1 , Borg I , 0	Party, Seats Won, Candidates: D , 2 , Pike , Foster R , 1 , Borg I , 0	This testfile was supposed to just run the OPL Election and the test data was written in a way where we are trying to just run the election and see if we get the correct results at the end.

Post condition(s) for Test:

After the test is run, the system should give the right results for the OPL election based on the test data in OPLTest1.csv as specified above in the steps. The media and audit files should also be created with Election Information.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: System

Test Date: March 14th, 2021

Test Case ID#: SUT3

Name(s) of Testers: Sai Tallapragada

Test Description:

In this system integration test, we are testing to see if our program can handle OPL election data where we will end up with a tie between the remainders of two different parties with the highest number of ballots. We will see if the program is able to distribute all the seats correctly and break the tie between the two highest parties with the equal number of votes.

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

We are storing the test file OPLTest3.csv in the testing folder which is located inside the Project1 folder of our repo. When you input the file into the main program to do system Integration testing, please use the file path to the file.

Automated: yes___ no _X__

Results: Pass _____ Fail____X____

Preconditions for Test:

Compile the program with javac Main.java and run the program with java Main. When the program asks you for input, put in the file OPLTest3.csv. Please make sure to put in the file path.

Step	**Test Step**	**Test**	**Expected**	**Actual**	
#	**Description**	**Data**	**Result**	**Result**	**Notes**
1	Compile Main Program javac Main.java				
2	Run the main() in Main CLass where you will prompted for User Input				

	java Main				
3	Input the file OPLTest3.csv. Make sure to input the file path to that csv for the input.	OPLTest3.csv	Party, Seats Won, Candidates: D , 0 R , 2 , Borg I , 1 , Smith	Party, Seats Won, Candidates: D , 1 , Pike R , 1 , Borg I , 1 , Smith	This Program is testing whether our program can handle distributing seats to the different parties when their remainders have the same value(There is a tie) We are trying to see if our program is able to give us the correct output and distribute the seats correctly when we see if we get the correct results at the end.

Post condition(s) for Test:

We are supposed to have either the Republican party or Independent party winning the OPL Election at the end of the test. However, we ended up with a three way tie and this went wrong.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: System

Test Date: March 14th, 2021

Test Case ID#: SUT4

Name(s) of Testers: Samuel Wong

Test Description:

In this system integration test, we are testing to see if our program can handle IR election data where we will end up with a tie between two candidates. We will see if the program is able to break the tie using breakTie().

We are storing the test file IRTie.csv in the testing folder which is located inside the Project1/testing folder of our repo. When you input the file into the main program to do system Integration testing, please use the file path to the file.

Automated: yes___ no _X__

Results: Pass X Fail

Preconditions for Test:

Compile the program with javac Main.java and run the program with java Main. When the program asks you for input, put in the file IRTie.csv. Please make sure to put in the file path.

Step \#	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile Main Program javac Main.java				
2	Run the main() in Main CLass where you will prompted for User Input java Main				
3	Input the file IRTie.csv. Make sure to input the file	IRTie.csv	Roughly half the time we should get Sam wins and half the time we get Andy wins	Right around 50% of the time it was ran we got different results.	

	path to that csv for the input. Run multiple times to see if it really is giving different results				

Post condition(s) for Test:

Audit file containing information about the election as well as media file have been created in directory.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: System

Test Date: March 26th, 2021

Test Case ID#: SUT5

Name(s) of Testers: Samuel Wong

Test Description:

In this system integration test, we are testing to see if our program can handle IR election data where we will end up with a tie betweenthree candidates. We will see if the program is able to break the tie using breakTie().

We are storing the test file IRTie2.csv in the testing folder which is located inside the Project1/testing folder of our repo. When you input the file into the main program to do system Integration testing, please use the file path to the file.

Automated: yes___ no _X__

Results: Pass X Fail

Preconditions for Test:

Compile the program with javac Main.java and run the program with java Main. When the program asks you for input, put in the file IRTie.csv. Please make sure to put in the file path.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes

1	Compile Main Program javac Main.java				
2	Run the main() in Main Class where you will prompted for User Input java Main				
3	Input the file IRTie2.csv. Make sure to input the file path to that csv for the input. Run multiple times to see if it really is giving different results	IRTie2.csv	When running the program multiple times we should expect to see an even distribution between the amount of times each of the three candidates won	After running multiple times we saw a near even distribution of winners	

Post condition(s) for Test:

Audit file containing information about the election as well as media file have been created in directory.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: System

Test Date: March 26th, 2021

Test Case ID#: SUT6

Name(s) of Testers: Samuel Wong

Test Description:

In this system integration test, we are testing to see if our program can handle OPL election data where there's more seats than candidates to receive them

We are storing the test file OPLTest4.csv in the testing folder which is located inside the Project1/testing folder of our repo. When you input the file into the main program to do system Integration testing, please use the file path to the file.

Automated: yes___ no _X__

Results: Pass Fail X

Preconditions for Test:

Compile the program with javac Main.java and run the program with java Main. When the program asks you for input, put in the file IRTie.csv. Please make sure to put in the file path.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Compile Main Program javac Main.java				
2	Run the main() in Main Class where you will prompted for User Input java Main				
3	Input the file OPLTest3.csv. Make sure to input the file path to that csv for the input.	OPLTest3.csv	We should expect to see party D win 2 seats, R win 2 seats, and I win one	Index out of bounds error was given.	This error is most likely due to the fact that we don’t handle this case where there’s more seats to be assigned than candidates to assign them to.

Post condition(s) for Test:

Audit file containing information about the election as well as media file have been created in directory.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit **Test Date: 3/14**

Test Case ID#: UT1 **Name(s) of Testers: Samuel Wong**

Test Description: Test found in IR_Ballot_Tests.java file, functions used include IR_Ballot constructor, getID(), getChoice, and getCandidates.

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

All needed classes are present, IR_Ballot and Candidate, and are able to be instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Tests IR_Ballots constructor correctly setting the ID Field	Input will be a IR_Ballot object with ID field set to 1	getID() should return 1	getID() returned 1	
2	Test IR_Ballot's constructor correctly setting candidates field	Input will be an empty candidates array by default.	getCandidates should return an empty array.	getCandidates returned an empty array.	
3	Test IR_Ballot's constructor correctly setting choice field	Input will be a IR_Ballot object with Choice field 0	getChoice() should return 0	getChoice() returned 0	

Post condition(s) for Test:

A IR_Ballot object has been created with default fields of ID=1, Candidates=[], choice=0;

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT2

Name(s) of Testers: Samuel Wong

Test Description: Tests IR_Ballot's getChoice method. Uses IR_Ballot's constructor,getChoice,setChoice;

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

All needed classes are present, IR_Ballot and Candidate, and are able to be instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Tests IR_Ballots getChoice function correctly retrieving choice field	Input will be a IR_Ballot object choice field set to 0	getChoice should return 0	getChoice returned 0	
2	Tests IR_Ballots settChoice function correctly setting choice field	Input will be an IR_Ballot object with choice field set using setChoice to 1	After calling setChoice(1), getChoice should return 1	getChoice returned 1	

Post condition(s) for Test:

A IR_Ballot object has been created with default fields of ID=1, Candidates=[], choice=1

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT3

Name(s) of Testers: Samuel Wong

Test Description: Test IR_Ballot's getCurrentChoice method, uses Candidate objects, IR_Ballot constructor, setChoice, and getCurrentChoice

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

All needed classes are present, IR_Ballot and Candidate, and are able to be instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Test IR_Ballot’s getCurrentChoice method on default choice of 0	Input will be a IR_Ballot object with choice 0 and Candidates[c1,c2]	getCurrentChoice should return c1	getCurrentChoice returned c1	Will break if given an index out of bound, but that will never happen without intentionally trying to break it
2	Test IR_Ballot’s getCurrentChoice method after changing current choice	Input will be a IR_Ballot object with choice changed to 1 and Candidates[c1,c2]	getCurrentChoice should return c2	getCurrentChoice returned c2	

Post condition(s) for Test:

A IR_Ballot object has been created with default fields of ID=1, Candidates=[c1,c2], choice=1

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT4

Name(s) of Testers: Samuel Wong

Test Description: Test OPL_Ballot’s constructor, using getID,getCandidate, and the constructor itself

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

All needed classes are present, OPL_Ballot and Candidate, and are able to be instantiated.

Ste p #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Test OPL_Ballots constructor making sure it sets the ID field correctly	Input will be a OPL_Ballot object ID field set to 1	getID should return 1	getID returned 1	
2	TestsOPL_Ballots constructor making sure it sets the candidate field correctly.	Input will be an OPL_Ballot object with candidate field set to c1	getCandidate should return c1	getCandidate returned c1	

Post condition(s) for Test:

A OPL_Ballot object has been created with default fields of ID=1, Candidates=c1

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT5

Name(s) of Testers: Samuel Wong

Test Description: Tests OPL_Ballots set and get Candidate methods using the constructor and the setter and getter respectively

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

All needed classes are present, OPL_Ballot and Candidate, and are able to be instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Tests OPL_Ballots getCandidate method	Input will be an OPL_Ballot with candidate field set to c1	getCandidate should return c1	getCandidat returned c1	
2	Tests OPL_Ballots setCandidate method	Input will be an OPL_Ballot with candidate field changed to c2	After calling setCandidate(c2) getCandidate should return c2	getCandidate returned c2	

Post condition(s) for Test:

A OPL_Ballot object has been created with fields of ID=1, Candidate=c2

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit **Test Date: 3/14**

Test Case ID#: UT6 **Name(s) of Testers: Andy Chen**

Test Description: Tests Candidate’s constructor to ensure its initialising its name field properly by calling getName(), and checking to see if ArrayList is initialized

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot class needs to be present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes

1	Tests Candidate’s constructor on String “test”	Input will be a String “test”	Calling the getName() method will return an instance of a String	getName() returned an instance of a String	
2	Tests Candidate’s initialization of its ‘votes’ field after calling the constructor	Input will be a String “test”	Calling getBallots() will not return null	getCandidate returned != null	

Post condition(s) for Test:

A Candidate object has been instantiated with its ‘name’ field set to a String and ‘votes’ field instantiated.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit **Test Date: 3/14**

Test Case ID#: UT7 **Name(s) of Testers: Andy Chen**

Test Description: Tests Candidate’s getNumVotes() method returns the number of Ballots after calling appendBallot() to add Ballots to the object.

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot class needs to be present, a Candidate object has been instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create 3 Ballot objects, and call appendBallot() to add each to the test Candidate	Inputs for each call of appendBallot() will be 3 different Ballot objects	Calling the getNumVotes() will return the number of Ballots in the Candidate object, being 3.	getName returned 3	

Post condition(s) for Test:

A Candidate object has been given 3 Ballot objects into its ArrayList<Ballot> via a working appendBallot() and will return the number of Ballots it holds in its list when getNumVotes() is called.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT8

Name(s) of Testers: Andy Chen

Test Description: Tests Candidate's appendBallot() and getBallots() functions.

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot class needs to be present, a Candidate object has just been instantiated.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new test Ballot, and call appendBallot() to add it to the test Candidate	Input will be a Ballot object	Calling appendBallot() will add the test Ballot to the ArrayList<Ballot> of Candidate which can later be retrieved with getBallots()	getName returned 3	
2	Call getBallots()		Will return an ArrayList<Ballot> containing just the Ballot added in the previous step	getBallots() returned an ArrayList<Ballot> containing just the testBallot	

Post condition(s) for Test:

A Candidate object has been given a Ballot via appendBallot() and calling getBallots() will return all the Ballots it holds

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT9

Name(s) of Testers: Andy Chen

Test Description: Tests Candidate’s getName() function which should return the name it was instantiated with

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot class needs to be present.

Ste p #	Test Step Description	Test Data	Expected Result	Actual Result	Notes

1	Instantiate a new Candidate passing it String “test” and then call getName()	Input is String “test”, no parameter for getName() call	Result of calling getName() should be “test”	getName returned “test”	

Post condition(s) for Test:

A Candidate object has been instantiated with the ‘name’ field initialized to “test”, and calling getName() will properly return “test” in this case, or the contents of the ‘name’ field in others.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT10

Name(s) of Testers: Andy Chen, Sai Tallapragada

Test Description: Tests Party’s constructor, uses getPartyName()

Automated: yes:X no ___

Results: Pass:X   Fail________

Preconditions for Test:

The Ballot, and Candidate class needs to be present.

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Instantiate a new Party by passing it a String for the name	Input is String “D”	Result of calling getPartyName() should return “D”	getPartyName() returns “D”	

Post condition(s) for Test:

A Party has been created with its ‘partyName’ field set to “D”

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT11

Name(s) of Testers: Andy Chen, Sai Tallapragada

Test Description: Tests Party's setVotes() and getVotes() functions

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot, and Candidate class needs to be present, a Party has been instantiated

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Use setVotes() to set their votes to 5	Input Int 5 for setVotes()	Result of calling setVotes() should set the 'votes' field to 5 and calling getVotes() should return 5	getVotes() returns 5	
2	Calling getVotes() on the Party object		getVotes() should return whatever setVotes() set, being 5 in this case	getVotes() returns 5	

Post condition(s) for Test:

The test Party now has its 'votes' field set to 5, and calling getVotes() will properly return the number in 'votes', in this case being 5.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit **Test Date: 3/14**

Test Case ID#: UT12

Name(s) of Testers: Andy Chen, Sai Tallapragada

Test Description: Tests Party's setSeats() and getSeats()

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot, and Candidate class needs to be present, a Party has been instantiated

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes

1	Call setSeats() and with parameter 5	Parameter is int, 5	Calling getSeats() should return the same number set by setSeats(), 5	getSeats() returns 5	

Post condition(s) for Test:

The test Party now has its ‘numSeats’ field set to 5, and calling setSeats() will properly return the number in ‘numSeats’, in this case being 5.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT13

Name(s) of Testers: Andy Chen, Sai Tallapragada

Test Description: Tests Party’s addCandidate() and setTotalVotes(), uses getVotes()

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot, and Candidate class needs to be present, a Party has been instantiated, test Candidates have been created, each having Ballots

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Call addCandidate() on 2 Candidates, 1 having 2 Ballots, 1 having 1 Ballot, to add them to the Party, call setTotalVotes() to set 'votes' to the total Ballots of the 2 Candidates combined	2 Candidates, 1 having 1 Ballot and 1 having 2 Ballots	Result of calling getVotes() should be the combined total of the 2 Candidate's Ballots which should be calculated by setTotalVotes(), being 3.	getVotes() returns 3	

Post condition(s) for Test:

The test Party now has its Candidates ArrayList (‘candidates’) containing 2 Candidate objects, its ‘votes’ field set to 3, and addCandidate(), setTotalVotes(), are confirmed to work.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT14

Name(s) of Testers: Andy Chen, Sai Tallapragada

Test Description: Tests Party’s getTopCandidates(), which also uses addCandidate()

Automated: yes:X no ___

Results: Pass:X Fail________

Preconditions for Test:

The Ballot, and Candidate class needs to be present, a Party has been instantiated, 2 test Candidates have been created, one having more than the other

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Call addCandidate() on 2 Candidates, 1 having 2 Ballots, 1 having 1 Ballot, to add them to the Party, call getTopCandidates() with parameter 1 to get 1 Candidate with the most votes.	Parameter 1 being the number of Candidates to be returned, 2 test Candidates 1 having 2 ballots, and 1 having just 1 ballot	Result of calling getTopCandidates() with parameter 1 should be a size 1 array containing the Candidate with 2 Ballots	getTopCandidates returns an array containing the Candidate with 2 Ballots	

Post condition(s) for Test:

The test Party now has its Candidates ArrayList (‘candidates’) containing 2 Candidate objects, the getTopCandidates() function is confirmed to work properly

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT15

Name(s) of Testers: Sree Pemma, Sai Tallapragada

Test Description: Tests OPL Election’s constructor.

Automated: yes:X no ___

Results: Pass:_ Fail:X

Preconditions for Test:

The test file that needs to be passed into the Buffered Reader needs to be created as well as the Buffered Reader that reads it in.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new test file for the test and a new BufferedReader object that reads in the test file.	Buffered Reader object takes in a test file that is the same as the example on the IR and OPL election description document.			
2	Instantiate new OPL Election object and pass in BufferedReader	Input buffered reader that reads in test file into the OPL_Election object (constructor for that class).			
3	asserting whether the candidates and parties global array lists are not null as well as if the seats are 0.		Result of calling the constructor of OPL_Election should be a new candidate array list and parties array list being created. The number of seats for the election should be set to 0.	We are not sure as the test is not finding the file due to the reasons of either the path not being stated properly or for some other reason the file isn't being found. We get a FIleNotFoundException.	

Post condition(s) for Test:

If the test worked, a new ArrayList for candidates and parties would be created and these arrays would be confirmed to not be null and the number of seats would be confirmed to equal 0.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT16

Name(s) of Testers: Sree Pemma, Sai Tallapragada

Test Description: Tests OPL Election’s Run Election function.

Automated: yes:X no ___

Results: Pass:_ Fail:X

Preconditions for Test:

The test file that needs to be passed into the Buffered Reader needs to be created as well as the Buffered Reader that reads it in.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new test file for the test and a new BufferedReader object that reads in the test file.	Buffered Reader object takes in a test file that is the same as the example on the IR and OPL election description document.			
2	Instantiate new OPL Election object and pass in BufferedReader	Input buffered reader that reads in test file into the OPL_Election object (constructor for that class).			
3	Asserting different things that the run election function does such as the number of candidates and seats and total votes being set properly baked on the test file.		All should be true.	Not sure as it doesn't run..	
4	Asserting whether the expected string matches the actual string that is output		The expected result is "`Party, Seats Won, Candidates:\n`	We do not know what the actual output is as the test file is not found. We receive a	

	from the run election function which declares who won.		`" + "D , 2 , Pike , Foster\n" +`
			`"R , 1 , Borg\n" + "I , 0\n".`
				FileNotFoundError and this could be due to the path not being specified properly or because of some other reason.	

Post condition(s) for Test:

If the test worked, we would know that the run election function runs properly with the right values being set to the right variables as well as that the correct winner is found.

Poject Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit

Test Date: 3/14

Test Case ID#: UT17

Name(s) of Testers: Sree Pemma, Sai Tallapragada

Test Description: Tests Election's assign ballot function.

Automated: yes:X no ___

Results: Pass:X Fail:_

Preconditions for Test:

The test file that needs to be passed into the Buffered Reader needs to be created as well as the Buffered Reader that reads it in.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new test file for the test and a new BufferedReader object that reads in the test file.	Buffered Reader object takes in a test file that is the same as the example on the IR and OPL election description document.			
2	Instantiate new OPL Election object and pass in BufferedReader. Also instantiate a	Input buffered reader that reads in test file into the OPL_Election object			

	ballot and candidate object to pass into the assign ballot function.	(constructor for that class).			
3	Call assign ballot function.	Pass in the ballot and candidate objects that we created earlier as params.			
4	Asserting whether the new ballot has been assigned to the candidate by retrieving the last ballot in the candidate's ballot array.		The assert statement should be true.	The result is that the test passes, meaning that the assert statement is successful.	

Post condition(s) for Test:

The test is successful meaning that the function works properly by assigning a ballot correctly to a candidate.

Project Name: Project 1: IR/OPL Voting System Team#24

Test Stage: Unit	**Test Date: 3/14**
Test Case ID#: UT18	**Name(s) of Testers: Sree Pemma, Sai Tallapragada**

Test Description: Tests OPL Election's constructor.

Automated: yes:X no ___

Results: Pass:_ Fail:X

Preconditions for Test:

The test file that needs to be passed into the Buffered Reader needs to be created as well as the Buffered Reader that reads it in.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new test file for the test and a new	Buffered Reader object takes in a test file that is the			

	BufferedReader object that reads in the test file.	same as the example on the IR and OPL election description document.			
2	Instantiate new IR Election object and pass in BufferedReader.	Input buffered reader that reads in test file into the IR_Election object (constructor for that class).			
3	Asserting whether the number of candidates and the number of eliminated candidates agave the proper values when calling this function.		The assert statement should be true.	The result is that there is a FileNotFoundException as the test file is not being found either because the path is not right or for some other reason.	

Post condition(s) for Test:

The test is not successful but if it was, the post condition would be that the IR Election run election function works properly and correctly identifies the winner and everything else that occurs in this function.