Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X	**Test Date: 3/24/22**
Test Case ID#: IRV_System_1	**Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**
Test Description: Test to see if IRV system can successfully determine the winners for an election with a general case election scenario.	
	Indicate where are you storing the tests (what file) and the name of the method/functions being used. **Input_IRV.csv (run according to ReadMe instructions)**
Automated: yes___ no X	

Results: Pass Fail X

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in “javac -d . src/*.java testing/*.java” And then type in “java p1.Main src/Input_IRV.csv”	Input_IRV.csv			
2	Verify that result.file has the correct election results	resultFile.txt	Rosen and Chou are declared winners of the election	Rosen and Chou are declared winners of the election	
3	Verify that the election process in audit.file	auditFile.txt	The election process can be followed and there are no noticeable errors. Rosen and Chou should be declared winners by	There is a seat redistributed even though there should be no seat redistributions	

			promoting them from loser list.		

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X	**Test Date: 3/25/22**
Test Case ID#: IRV_System_2	**Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**
Test Description: Test to see if IRV system can recognize that a candidate reached droop and declare them a winner	
	Indicate where are you storing the tests (what file) and the name of the method/functions being used.
Automated: yes___ no X	**IRV_Input1.csv (run according to ReadMe instructions)**

Results: Pass Fail X

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in "javac -d . src/*.java testing/*.java" And then type in	Input_IRV1.csv			

	“java p1.Main src/Input_IRV1.csv”				
2	Verify that result.file has the correct election results	resultFile.txt	Rosen and Klienberg are declared winners of the election	Rosen and Klienberg are declared winners of the election	
3	Verify that the election process in audit.file	auditFile.txt	The election process can be followed and there are no noticeable errors. Rosen should be declared a winner by reaching the droop quota and Klienberg should be declared a winner by promoting them from loser list.	Rosen does not receive a seat by droop which is not correct.	

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X	**Test Date: 3/25/22**
Test Case ID#: IRV_System_3	**Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**
Test Description: Test to see if different candidates win depending on election run if when there is a tie	
	Indicate where are you storing the tests (what file) and the name of the method/functions being used. **Input_IRV2.csv (run according to ReadMe instructions)**
Automated: yes___ no X	

Results: Pass X Fail

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in "javac -d . src/*.java testing/*.java" And then type in "java p1.Main src/Input_IRV2.csv"	Input_IRV2.csv			
2	Verify that result.file has the correct election results	resultFile.txt	Rosen is declared winner of the election. Either Chou or Kleinberg is declared second winner based on a coin flip	Rosen and Klienberg are winners.	
3	Verify that the election process in audit.file	auditFile.txt	The election process can be followed and there are no noticeable errors. Rosen should be declared a winner by reaching the droop	Rosen is declared a winner by reaching droop quota and Klienberg is declared a winner by promoting them from the loser list.	Coin flip is not indicated in Audit file. - added to bug list

			quota and either Klienberg or Chou should be declared a winner by promoting them from loser list.		
4	Repeat steps 1				
5	Repeat step 2	resultFile.txt	Chou will be declared a winner instead of Klienberg	Chou is declared a winner instead of Klienberg	

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X **Test Date: 3/25/22**

Test Case ID#: IRV_System_4 **Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**

Test Description: Test to see if system will recognize that candidate reached droop quota through reallocation

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

Input_IRV3.csv (run according to ReadMe instructions)

Automated: yes___ no X

Results: Pass Fail X

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in "javac -d .	Input_IRV3.csv			

	src/*.java testing/*.java” And then type in “java p1.Main src/Input_IRV3.csv”				
2	Verify that result.file has the correct election results	resultFile.txt	Rosen and Chou are winners	Rosen and Chou are winners.	
3	Verify that the election process in audit.file	auditFile.txt	The election process can be followed and there are no noticeable errors.Rosen is declared winner of the election by reaching droop quota through reallocation. Chou is declared the second winner of the election by reaching droop quota.	Rosen is not recognized as reaching droop quota by system after vote reallocation.	
4					
5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X	**Test Date: 3/24/22**
Test Case ID#: OPL_System_1	**Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**
Test Description: Test to see if OPL System works for a general case where seats are allocated both through hitting droop quota and through remainders.	
	Indicate where are you storing the tests (what file) and the name of the method/functions being used. **Input_OPL.csv (run according to ReadMe instructions)**
Automated: yes___ no X	

Results: Pass X Fail

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in “javac -d . src/*.java testing/*.java” And then type in “java p1.Main src/Input_OPL.csv”	Input_OPL.csv			
2	Verify that result.file has the correct election results	resultFile.txt	Seats are given to Pike,Deutsch, Foster. And Party D received 7 votes Party R received 4 votes Party I received 2 votes	Seats are given to Pike,Deutsc, Foster. And Party D received 7 votes Party R received 4 votes Party I received 2 votes	

3	Verify that the election process in audit.file	auditFile.txt	The election process can be followed and there are no noticeable errors. Pike receives a vote through achieving droop quota and Deutsch and Foster receive seats through remainders	The election process can be followed and there are no noticeable errors. Pike receives a vote through achieving droop quota and Deutsch and Foster receive seats through remainders	
4					
5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X **Test Date: 3/26/22**

Test Case ID#: IRV_System_5 **Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**

Test Description: Test to see if shuffle is on by default

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

Input_IRV3.csv (run according to ReadMe instructions)

Automated: yes___ no X

Results: Pass X Fail

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in "javac -d .	Input_IRV3.csv			

	src/*.java testing/*.java” And then type in “java p1.Main src/Input_IRV3.csv”				
2	Verify that the ballot files are in a random order in the audit file	auditFile.txt	Ballots are counted in random orderduring first pass	Ballots are counted in random order during first pass	
3	Repeat step 1				
4	Repeat step 2		Ballots are in different random order than before	Ballots were in different random order than before	
5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X	**Test Date: 3/26/22**
Test Case ID#: IRV_System_6	**Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken**
Test Description: Test to see if shuffle is on by default	
	Indicate where are you storing the tests (what file) and the name of the method/functions being used. **Input_IRV3.csv (run according to ReadMe instructions)**
Automated: yes___ no X	

Results: Pass X Fail

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in “javac -d . src/*.java testing/*.java” And then type in “java p1.Main src/Input_IRV3.csv” e	Input_IRV3.csv			
2	Verify that the ballot files are read in the order in the ballot file	auditFile.txt	Ballots are read in order found in ballot file	Ballots are counted in order in ballot file	
3					
4					

5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X

Test Date: 3/26/22

Test Case ID#: OPL_System_2

Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken

Test Description: Test to see if OPL algorithm can reallocate seats correctly if a party receives more seats than candidates

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

Input_OPL1.csv (run according to ReadMe instructions)

Automated: yes___ no X

Results: Pass Fail X

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in "javac -d . src/*.java testing/*.java" And then type in "java p1.Main src/Input_OPL1.csv"	Input_OPL1.csv			

2	Verify that the election process in auditFile.txt	Verify that the election process in auditFile.txt	The election process can be followed and there are no noticeable errors. 2nd independent seat is resdistirbuted to R party	The system does not recognize that it already awarded the D party a seat by remainder and awards them another one	
3	Repeat step 1 due to tie to see different result				
4	Verify that the election process in auditFile.txt	Verify that the election process in auditFile.txt	The election process can be followed and there are no noticeable errors. 2nd independent seat is resdistirbuted to R party	The system awards 2 seats to the same independent candidate	
5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.

Project Name: Project 1 Voting System

Team #9

Test Stage: Unit __ System X

Test Date: 3/27/22

Test Case ID#: OPL_System_3

Name(s) of Testers: Naviin Vejaya Kumar, John Cullom, Anna Frenz, Kyle Bekken

Test Description: Test to see if OPL algorithm can reallocate seats correctly if a party receives more seats than candidates

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

Input_OPL4.csv (run according to ReadMe instructions)

Automated: yes___ no X

Results: Pass Fail X

Preconditions for Test: Input file must be in the folder as the system's java files so that system can access it.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Navigate to the Project1 folder and type in “javac -d . src/*.java testing/*.java” And then type in “java p1.Main src/Input_OPL4.csv”	Input_OPL4.csv			
2	Verify that the election process in auditFile.txt	auditFile.txt	The Democratic party should be allocated 2 seats by meeting quota and one by remainder, should be able to follow election logic	The Democratic party is awarded 3 seats through the means expectedbut the Democratic party received one extra vote	Added bug to bug list
3	Verify that the election process in resultFile.txt	resultFile.txt	The Democratic party should be allocated 2 seats by meeting quota and one by remainde	The Democratic party is awarded 3 seats through the means expected	

4					
5					

Post condition(s) for Test: New audit and result files are generated and system successfully finishes running.