Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __

Test Date: 3-13-2021

Test Case ID#: Ballot_Getters_Setters

Name(s) of Testers: Hoai Bui

Test Description: This test will test the getters and setters for currDis and id data members.

The tests are stored in the ballot_unittest.cc file.
The methods used are SetCurrDis, SetId, GetCurrDis, GetId

Automated: yes_✓__ no ___

Results: Pass __✓___ Fail_________

Preconditions for Test:

- **ballot.h and ballot.cpp must compile**
- **Parameters for SetCurrDis and SetId are integers.**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the GetCurrDis method	none	0	0	
2	Testing the SetCurrDis method	int 1	1	1	
3	Testing the GetId method	none	1	1	
4	Testing the SetId method	99	99	99	

Post condition(s) for Test:

- id and currDis data members are set to the desired values
- Desired values are returned for GetId() and SetId()

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Ballot_Add_Candidate **Name(s) of Testers: Hoai Bui**

Test Description: This test will test that the AddCandidate method properly adds a candidate string to the candidates vector

The test is stored in the ballot_unittest.cc file.
The methods used are AddCandidate and Print

Automated: yes__✓_ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **ballot.h and ballot.cpp must compile**
- **Parameter for AddCandidate must be a string**

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Testing the AddCandidate method	“Trump”	“Trump\n”	“Trump\n”	

Post condition(s) for Test:

- A candidate is added to the candidates vector

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Ballot_Print **Name(s) of Testers: Hoai Bui**

Test Description: This test will test whether or not the Print method prints out the elements of the candidates vector in the correct order

The test is stored in the ballot_unittest.cc file.
The methods used are AddCandidate and Print

Automated: yes ✓ no

Results: Pass ___✓__ Fail ________

Preconditions for Test:

- **ballot.h and ballot.cpp must compile**
- **Parameter for AddCandidate must be a string**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the Print method	“Trump” “Biden” “Harris”	“Trump\nBiden\nHarris\n”	“Trump\nBiden\nHarris\n”	

Post condition(s) for Test:

- The candidates vector is populated with “Trump”, “Biden”, and “Harris”
- The elements of the candidates vector are printed out in the correct order

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Candidate_Getters_Setters **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the getters and setters for the Candidate class work properly.

The tests are stored in the candidate_unittests.cc file. The methods used are GetName, GetParty, SetName, SetParty, GetBallotListSize

Automated: yes __✓_ no ___

Results: Pass ✓ Fail

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **Parameters for SetName and SetParty must strings**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the GetName method	“Bui	“Bui	“Bui”	
2	Testing the GetParty method	“R”	“R”	“R”	
3	Testing the SetName method	“Tran”	“Tran”	“Tran”	
4	Testing the SetParty method	“D”	“D”	“D”	
5	Testing the GetBallotListSize method	none	1	1	

Post condition(s) for Test:

- A candidate object’s name and party are set to desired values
- Desired values for party, name, ballot list size are returned

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __

Test Date: 3-13-2021

Test Case ID#: Candidate_Add_Ballot

Name(s) of Testers: Hoai Bui

Test Description: This test will determine if the AddBallot method properly adds a ballot to the ballots vector for a candidate

The tests are stored in the candidate_unittests.cc file.
The methods used are AddBallot and GetBallotListSize

Automated: yes__✓_ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **Parameter must be a pointer to a ballot object**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Added a ballot to the ballots vector and returned the vector's sze	*newBallot	1	1	

Post condition(s) for Test:

- A ballot is added to a candidate object's ballots vector

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Candidate_Add_Ballot **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the RemoveBallot method properly removes a ballot to the ballots vector for a candidate

The tests are stored in the candidate_unittests.cc file. The methods used are AddBallot, RemoveBallot and GetBallotListSize

Automated: yes __✓__ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Adding a ballot to the ballots vector	*newBallot	none	none	
2	Removed the ballot the from the vector and returned the vector's size	none	0	0	

Post condition(s) for Test:

- A ballot is added to a candidate object's ballots vector

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Driver_Get_File_Name **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the GetFileName method

The test is stored in the driver_unittests.cc file.
The method used is GetFileName

Automated: yes__✓_ no ___

Results: Pass ___✓__ Fail_________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **driver.h and driver.cpp must compile**
- **election.h and election.cpp must compile**
- **A driver object must exist**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the GetFileName method	none	“ballots”	“ballots”	

Post condition(s) for Test:

- A candidate object’s name and party are set to desired values
- Desired values for party, name, ballot list size are returned

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Driver_Read_IR_Arguments **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the read methods work properly for an instant runoff election.

The tests are stored in the driver_unittests.cc file. The methods used are ReadInElectionType, ReadInNumCandidates, ReadInNumberofBallots, ReadInBallots

Automated: yes __✓_ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **driver.h and driver.cpp must compile**
- **election.h and election.cpp must compile**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Driver class is constructed	ir.csv	none	none	
2	Testing ReadInElectionType	none	"Compute IR election.\n"	"Compute IR election.\n"	
3	Testing ReadInNumCandidates	none	"Number of candidates: 4\n"	"Number of candidates: 4\n"	
4	Testing ReadInCandidates	none	0	0	
5	Testing ReadInNumberOfBallots	none	"Number of ballots: 9\n"	"Number of ballots: 9\n"	
6	Testing ReadInBallots	none	0	0	

Post condition(s) for Test:

- All data is read from the CSV file for an instant runoff election.

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Driver_Read_OPL_Arguments **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the read methods work properly for an OPL election.

The tests are stored in the driver_unittests.cc file. The methods used are ReadInElectionType, ReadInNumCandidates, ReadInNumberofBallots, ReadInBallots, ReadInNumberOfSeats

Automated: yes__✓_ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **driver.h and driver.cpp must compile**
- **election.h and election.cpp must compile**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Driver class is constructed	opl.csv	none	none	
2	Testing ReadInElectionType	opl.csv	"Compute OPL election.\n"	"Compute OPL election.\n"	
3	Testing ReadInNumCandidates	opl.csv	"Number of candidates: 6\n"	"Number of candidates: 6\n"	
4	Testing ReadInCandidates	opl.csv	0	0	
5	Testing for ReadInNumberOfSeats	opl.csv	“Number of seats: 3\n”	“Number of seats: 3\n”	
6	Testing ReadInNumberOfBallots	opl.csv	"Number of ballots: 9\n"	"Number of ballots: 9\n"	
7	Testing ReadInBallots	opl.csv	0	0	

Post condition(s) for Test:

- All data is read from the CSV file for an instant OPL election.

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit _✓_ System __ **Test Date: 3-13-2021**

Test Case ID#: Driver_Get_OPL_Vote **Name(s) of Testers: Hoai Bui**

Test Description: This test will determine if the GetOPLVote method works properly.

The test is stored in the driver_unittests.cc file. The method used is GetOPLVote.

Automated: yes ✓ no

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **candidate.h and candidate.cpp must compile**
- **ballot.h and ballot.cpp must compile**
- **driver.h and driver.cpp must compile**
- **election.h and election.cpp must compile**
- **A driver object must be created**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Driver class is constructed	opl.csv	none	none	
2	Testing ReadInElectionType	opl.csv	"Compute OPL election.\n"	"Compute OPL election.\n"	
3	Testing ReadInNumCandidates	opl.csv	"Number of candidates: 6\n"	"Number of candidates: 6\n"	
4	Testing ReadInCandidates	opl.csv	0	0	
5	Testing for ReadInNumberOfSeats	opl.csv	“Number of seats: 3\n”	“Number of seats: 3\n”	
6	Testing ReadInNumberOfBallots	opl.csv	"Number of ballots: 9\n"	"Number of ballots: 9\n"	
7	Testing ReadInBallots	opl.csv	0	0	

Post condition(s) for Test:

- All data is read from the CSV file for an instant OPL election.

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Getter_Setters

Name(s) of Testers: Emma Spindler

Test Description: This test will test all the getter and setters numberOfCandidates, numberOfBallots, numberOfSeats, and quota.

The tests are stored in the election_unittest.cc file. The methods used are GetNumberOfSeats, SetNumberOfSeats, GetNumberOfBallots, SetNumberOfBallots, GetNumberOfCandidates, SetNumberOfCandidates, GetQuota, SetQuota

Automated: yes_✓__ no ___

Results: Pass _✓____ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
- **Parameters for numberOfCandidates, numberOfBallots, numberOfSeats, and quota are integers.**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the GetNumberOfSeats method	none	3	3	
2	Testing the SetNumberOfSeats method	3	3	3	
3	Testing the GetNumberOfBallots method	none	9 6	9 6	
4	Testing the SetNumberOfBallots method	9 6	9 6	9 6	
5	Testing the GetNumberOfCandidates method	none	6	6	
6	Testing the SetNumberOfCandidates method	6	6	6	
7	Testing the GetQuota method	none	3	3	
8	Testing the SetQuota method	3	3	3	

Post condition(s) for Test:

- numberOfCandidates, numberOfBallots, numberOfSeats, and quota data members are set to the desired values

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Get_Set_ElectionType

Name(s) of Testers: Emma Spindler

Test Description: This test will test all the get and set of the electionType

The tests are stored in the election_unittest.cc file.
The methods used are GetElectionType and SetElectionType

Automated: yes_✓__ no ___

Results: Pass ___✓__ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
- **parameters for SetElectionType are a string**

Step # 1	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
2	Testing the GetElectionType method	none	OPL IR	OPL IR	
3	Testing the SetElectionType method	“OPL” “IR”	OPL IR	OPL IR	

Post condition(s) for Test:

None

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_VotesForParty

Name(s) of Testers: Emma Spindler

Test Description: This test will test the get and set votes for party/s

The tests are stored in the election_unittest.cc file. The methods used are GetVotesForParty and SetVotesForParties

Automated: yes_✓__ no ___

Results: Pass _✓____ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
- **Parameters for GetVotesForParty are strings.**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the GetVotesForParty method	‘D’	0	0	
2	Testing the SetVotesForParties method	none	0	0	

Post condition(s) for Test:

- The votes for each party are returned/set

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Increaser

Name(s) of Testers: Emma Spindler

Test Description: This test will test the incremental functions that use numberOfCandidates and numberOfBallots.

The tests are stored in the election_unittest.cc file.
The methods used are IncreaseNumberOfCandidates, and IncreaseNumberOfBallots

Automated: yes_✓__ no ___

Results: Pass ✓ Fail

Preconditions for Test:

- **election.h and election.cpp must compile**
- **numberOfCandidates and numberOfBallots are integers.**

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Notes**
1	Testing the IncreaseNumberOfCandidates method	none	6	7	
2	Testing the IncreaseNumberOFBallots method	none	10	10	

Post condition(s) for Test:

- none

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Add_Remove_Candidates **Name(s) of Testers: Emma Spindler**

Test Description: This test will test that the number of candidates gets modified.

The tests are stored in the election_unittest.cc file.
The methods used are AddCandidate and RemoveCandidate

Automated: yes ✓ no

Results: Pass _✓____ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
-

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the AddCandidate method	Candidate	1 1	1 1	
2	Testing the RemoveCandidate method	0	0	0	

Post condition(s) for Test:

- The number of candidates are set after removing candidate.

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Add_Party

Name(s) of Testers: Emma Spindler

Test Description: This test will add a party by name given.

The tests are stored in the election_unittest.cc file.
The methods used are AddParty.

Automated: yes ✓ no

Results: Pass _✓____ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
- **Must give a string name of the party.**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the AddParty method	string "Test"	0	0	

Post condition(s) for Test:

- none

Project Name: Project 1: Voting System **Team# 11**

Test Date: 3-13-2021

Test Stage: Unit _✓_ System __

Test Case ID#: Election_Majority **Name(s) of Testers: Emma Spindler**

Test Description: This test will test that if theres a majority vote or not.

The tests are stored in the election_unittest.cc file.
The methods used are CheckForMajority

Automated: yes_✓__ no ___

Results: Pass _✓____ Fail________

Preconditions for Test:

- **election.h and election.cpp must compile**
- **Must be IR election**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Testing the CheckForMajority method	none	0	0	

Post condition(s) for Test:

- none

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit __ System _✓_ **Test Date: 3-14-2021**

Test Case ID#: IR_100000_Candidates **Name(s) of Testers: Hoai Bui, Eric Palmer, Ryan Mower**

Test Description: This test will run an instant runoff election with 100000 candidates and check if the system run it in under 8 minutes

The test is stored in ir100000.csv, audit_ir_100000.txt, media_ir_100000.txt, IR_execution_runtime.png

Automated: yes___ no __✓_

Results: Pass __✓___ Fail________

Preconditions for Test:

- **The voting system must compile**
- **There must be an appropriate CSV file to run**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	The voting system is run with a CSV file of 100000 ballots	ir100000.csv	Voting System executes successfully and media and audit reports are produced	Voting System executes successfully and media and audit reports are produced	IR_execution_runtime.png shows that the system ran in under eight minutes and details of how the election went are in audit_ir_100000.txt

Post condition(s) for Test:

- A media report is generated
- An audit report is generated
- Results are displayed on terminal

Project Name: Project 1: Voting System **Team#11**

Test Stage: Unit __ System _✓_

Test Date: 3-14-2021

Test Case ID#: OPL_100000_Candidates

Test Description: This test will run an OPL election with 100000 candidates and check if the system run it in under 8 minutes

Name(s) of Testers: Hoai Bui, Eric Palmer, Ryan Mower

The test is stored in opl100000.csv, audit_opl_100000.txt, media_opl_100000.txt, OPL_execution_runtime.png

Automated: yes___ no __✓_

Results: Pass __✓___ Fail________

Preconditions for Test:

- **The voting system must compile**
- **There must be an appropriate CSV file to run**

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	The voting system is run with a CSV file of 100000 ballots	opl100000.csv	Voting System executes successfully and media and audit reports are produced	Voting System executes successfully and media and audit reports are produced	OPL_execution_runtime.png shows that the system ran in under eight minutes and details of how the election went are in audit_opl_100000.txt

Post condition(s) for Test:

- A media report is generated
- An audit report is generated
- Results are displayed on terminal