Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: Ballot_tests.h
Test Description:
Checks to see if the constructors in the Ballot class work correctly

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

Automated: yes_X_ no ___

Results: Pass _X____ Fail________

Preconditions for Test: Ballot.cc and Ballot_tests.h compile successfully

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Test if default constructor for Ballot object works correctly.	Takes no arguments.	The ballot number is 0 from b.get_ballot_no().	The result was as expected. The test case asserts that the ballot number does equal 0 as it passes the test case.	Pass
2	Test if manual constructor for Ballot object works correctly.	The constructor takes in an integer vector represent votes (<1,2>) and another integer for the ballot number (1)	The ballot number is 1 from b.get_ballot_no(). The first index of the vector when calling get_votes() is 1 and the second index is 2.	The result was as expected. The test case asserts that the first index of the vector is 1 and the second index is 2 when calling the get_votes() method.	Pass

Post condition(s) for Test:

All the TS_ASSERTS()'s in Ballot_tests.h show successful results.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: BallotList_tests.h
Test Description:
Checks to see if methods in BallotList class work correctly

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
tests/BallotList_tests.h
Tests these functions in order: ShuffleBallots(), RemoveBallot(), AddBallot(), ListSize(), MakeBallot(), and ReadFile()

Automated: yes_X_ no ___

Results: Pass __X___ Fail________

Preconditions for Test:
Ballot.cc, BallotList.cc, and BallotList_tests.h compile correctly.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Checks if ShuffleBallots will actually change the order of a list of ballots.	Takes no arguments.	The shuffled vector should not match the original vector of ballots.	The result was as expected as the test case passed because the shuffled vector does not match the original vector of ballots.	Pass
2	Checks if RemoveBallot will	Takes in an integer value	The ballot corresponding with ballot number 1 will be removed from the	The result was as expected as the test case passed for the	Pass

	actually remove a single ballot from the vector list of them.	representing the voter number which is the number of ballots to remove from the list.	BallotList vector.	ballot corresponding to number 1 was removed from the BallotList vector.	
3	Checks if AddBallot will add a ballot to the vector list of them all.	Takes the argument type of Ballot that indicates the ballot object that is to be added to the vector list.	Ballots of b0, b1, b2 are to be added into the vector. This should result in the ballot having a size of 3 for the amount of ballots stored in the vector. Once ballot number b3 is added to the vector, thew new list size will be updated to size of 4 and the get_ballot_list()[3].get_ballot_bo() will retrieve the latest ballot added to the vector resulting in the ballot number of 4.	The result was as expected.	Pass
4	Checks if ListSize will actually retrieve the size of the ballot vector list.	Takes no arguments.	When three ballots of b0, b1, and b2 are added to the vector list. The ListSize() should return the value 3.	The result was as expected.	Pass
5	Checks if MakeBallot will create a ballot.	Takes in a string variable that represents a single line from the csv file.	It makes sure that the BallotList object for the MakeBallot method assigns the votes of the ballots correctly to the array index based on its position.	The result was as expected as the test case ran successfully.	Pass
6	Checks if ReadFile will read the csv file to access the file information.	Takes in a string variable that represents the name of the file to read. It also takes in an integer value that represents	Will read the file that is being tested so in this instance “test.csv.” In reading the file, it will properly store in an array the vote value for each correct index based on row and column location of the file votes.	The result was as expected. The ReadFile method opened the “tests/test/csv” file and read in the 3 ballots and assigned their votes accordingly.	Pass

		the number of ballots to read.			

Post condition(s) for Test:

All the TS_ASSERTS()'s in BallotList_tests.h show successful results.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 03/20/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: CandidateList_tests.h
Test Description:
Checks to see if methods in CandidateList class work correctly

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
tests/CandidateList_tests.h
Tests these functions in order:]generic_constructor(), RemoveCandidate(), AddCandidate(), ReturnLoser(), ResturnWinner(), and ReturnWinners()

Automated: yes_X__ no ___

Results: Pass __X___ Fail________

Preconditions for Test:
The Candidate.cpp, Candidate.h, CandidateList.cpp, CandidateList.h compile.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Checks if default constructor creates a	Takes no arguments.	Makes sure size of empty vector of	The result is as the expected result.	Pass

	vector.		candidates size is equal to the candidate_list size vector.		
2	Checks if RemoveCandidate will actually remove a candidate from the candidate_list	Takes in a string variable that represents the name of the candidate to remove from the list.	Makes sure that candidate c with name “A” is removed from the candidate list. It should result in asserting that the candidate that is added has the same name as the one that is removed. It will also assert that the candidate list size is equal to 0 as only one candidate was added to the vector and is currently removed now.	The result was as expected as the test case passed in asserting that the candidate object name is the same as teh removed candidate name and that the the candidate list size is 0 because there are no longer any candidates in the list.	Pass
3	ThereExists	Takes a string variable that stands for the name of the candidate	The method makes sure that the name of the candidate “A” exists.	The result returned as being true as the candidate with the name “A” does exist.	Pass
4	Checks if AddCandidate will add a candidate to the candidate_list.	Takes in a Candidate object that holds the entities associated with that Candidate.	Makes sure that Candidate c with name “A” is added to the candidate list in index 0 of the vector.	The results were as expected. The Candidate c object that had a name of “A” was added to the list as the name of c is equal to the get_candidate_list name at the zeroth index.	Pass
5	Check if ReturnLoser would return the losing candidate’s name.	Takes no arguments.	Makes sure that candidate a with name “A” is the loser by verifying the candidates name with the ReturnLoser().get_name().	The results were as expected.	Pass
6	ReturnCandidate	Takes a string variable that represents the	The method makes sure that the name of the candidate “A” is equal	The results were as expected.	Pass

		name of the candidate to be returned.	to the name of the candidate that the Candidate List contains.		
7	Check if ReturnWinner would return a single winning candidate's name	Takes no arguments.	Makes sure that candidate b is the winner by checking to see that the candidate b's name matches correctly with the CandidateList winning candidate name.	The results were as expected in asserting that the winner is candidate b, whose name is "B" matches the CandidateList's ReturnWinner get_name function result.	Pass
8	Check if ReturnWinners would return many winning candidate's name	Takes in an integer value that shows the number of candidates seats that are available to be elected. This indicates the number of winning candidates to return.	The method makes sure that the Candidate object b name which is "B" is equal to the CandidateList object c's ReturnWinner's candidate name.	The results were as expected.	Pass
9	ListSize	Takes no arguments.	The method makes sure that the two added candidates to the CandidateList are accounted for by asserting that the size of the CandidateList is 2.	The results were expected as the test case successfully asserted that the list size was 2.	Pass

Post condition(s) for Test:

All the TS_ASSERTS()'s in CandidateList_tests.h show successful results

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __ **Test Date: 3/19/18**

Test Case ID#: Candidate_tests.h **Name(s) of Testers:** Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Description:
Checks to see if methods in Candidate class work correctly

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
tests/Candidate_tests.h
Tests these functions in order: generic_constructor() and custom_constructor()

Automated: yes_X_ no ___

Results: Pass __X___ Fail________

Preconditions for Test:
The Candidate.cpp, Ballot.cpp, Ballot.h and Candidate.h compile correctly.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Checks to see if the default, generic constructor is working properly in doing what it is suppose to.	Takes no arguments.	Makes sure the default constructor instantiates empty candidate names and sets the ballot number to zero	The expected result was the same as the actual result.	Pass
2	Checks to see if the custom constructor is working properly in doing what it is suppose to.	Takes no arguments.	Makes sure that the custom constructor properly assigns the name of the candidate based on the declaration given. Likewise, it should properly find the associated ballot number and list size. For candidate “A” being specified the constructor should obtain the candidate’s name of being “A” and	The expected result was the same as the actual result.	Pass

			set the number of ballot to 0 and the list size to 0 as well.		

Post condition(s) for Test:

All the TS_ASSERTS()'s in Candidate_tests.h show successful results.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: Election_tests.h
Test Description:
Checks to see if methods in Election Candidate class work correctly

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
tests/Election_tests.h
Tests these functions in order: generic_constructor(), custom_constructor(), move_ballot(), and read_names().

Automated: yes_X_ no ___

Results: Pass _____ Fail________

Preconditions for Test:
The CandidateList.cpp, CandidateList.h, BallotList.cpp, BallotList.h Candidate.cpp, Ballot.cpp, Ballot.h and Candidate.h compile correctly.

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

1	Test if the generic_constructor() which is the default constructor for Election works properly.	Takes in no arguments.	Makes sure to check if the number of seats, number of candidates, and number of ballots are initialized to 0 in the beginning.	The expected result was the same as the actual result.	Pass
2	Test if the custom_constructor which is the manual constructor for Election object works correctly.	Takes in three parameters of integers which include a variable for the number of seats available in the election, the number of candidates running and the number of ballots that exist.	Makes sure that the constructor correctly takes in values from the user input that defines its the number of seats available in the election, the number of candidates running and the number of ballots that exist.	The actual result holds valid as the number of seats, candidates and ballot values equal the values that were passed by the user.	Pass
3	Test if the Move_Ballot method is working correctly.	It takes in three parameters. The first parameter is an integer that is the ballot number. The next two parameters are of type BallotList that are the source and	It makes sure that ballot is added to the BallotList destination and is removed from the BallotList source.	The expected result was the same as the actual result.	Pass

		destination location for the BallotList object that is to be passed in.			
4	Test if the Read_Name method is working properly.	It takes in a string variable that represent the csv filename.	It makes sure that the candidate names from the “tests/test.csv” file is read and stored in the array of candidate_list for candidate names is stored properly.	The expected result was the same as the actual result.	Pass

Post condition(s) for Test:
All the TS_ASSERTS()’s in Election_tests.h show successful results.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit __ System _X_

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: Plurality_test
Test Description:
Checks the plurality method of this voting system.

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
This test is done manually by compiling and running “./elect.out” and entering Plurality when prompted by the console.

Automated: yes__ no _X_

Results: Pass __X___ Fail________

Preconditions for Test:
All files must compile correctly.

Step	Test Step	Test	Expected	Actual	

#	Description	Data	Result	Result	Notes
1	Check if plurality works with a single seat.	Number of candidates (9), number of seats (1), number of ballots (11), and ex.csv.	Displays the winner is [A]	as expected	Pass
1	Check if plurality works with multiple seats.	Number of candidates (9), number of seats (3), number of ballots (11), and ex.csv.	Displays the winners are [A], [G], E], [I]	as expected	Pass

Post condition(s) for Test:

The console displays the correct winners.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: Plurality_tests.h

Test Description:
Checks to see if the Plurality class methods of the ReturnHighestVoteIndex method is working.

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
tests/Plurality_tests.h
Tests the Plurality function of ReturnHighestVoteIndex().

Automated: yes_X_ no ___

Results: Pass _X____ Fail________

Preconditions for Test: The CandidateList.cpp, CandidateList.h, BallotList.cpp, BallotList.h Candidate.cpp, Ballot.cpp, Ballot.h and Candidate.h, Election.cpp, and Election.h must compile correctly.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Checks to see if the ReturnHighestVoteIndex method is working properly to return the highest votes index in the vector.	Takes in a Ballot object that has the attributes of a ballot.	The expected result is to return the highest vote index for the Ballot object b that has ballot number 1 in the vector v. The highest vote index for b is at 1 because in	The result was as expected.	Pass

			the vector that is the position that the ballot number 1 is located in.		

Post condition(s) for Test:
All the TS_ASSERTS()'s in Plurality_tests.h show successful results.

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit _X_ System __

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: STV_tests.h
Test Description:
Checks the STV methods of this voting system.

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

Automated: yes_X_ no __

Results: Pass __X___ Fail________

Preconditions for Test:
All files must compile correctly.

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

1	Checks if the GetIndex() function in STV works to return the index of the current vote on the ballot.	An integer vector of length 3, containing the integers 1,2,3. The function looks for the index containing 2.	Returns the index 1.	as expected	Pass
2	Checks if the ReturnNameOfVote function in STV works to return the Candidate name for the given vote.	The data in tests/test.csv in objects	For each index in the BallotList, it should return [A],[B],[A] for indeces 0,1,2 respectively.	as expected	Pass
3	Checks if the MoveCandidate function in STV works to move a Candidate from a source CandidateList to a destination CandidateList.	2 CandidateList’s containing 2 and 0 candidates respectively.	Each resulting CandidateList should be of length 1, and the name of the moved Candidate should appear in the originally empty CandidateList.	as expected	Pass
4	Checks if the CalculateDroop function in STV works to calculate the droop quota for the given BallotList.	The election parameters in tests/test.csv. Namely: 4 candidates, 1 seat, and 3 ballots.	The ‘droop’ attribute should contain the integer 2.	as expected	Pass

Post condition(s) for Test:

The testing framework indicates that all tests have passed.

+

Project Name: Project 1: Voting System **Team# 13**

Test Stage: Unit __ System _X_

Test Date: 3/19/18
Name(s) of Testers: Maxwell Dahl, Sanjana Jonnalagadda, Anthony Phan, Ronny Yogiswara

Test Case ID#: STV_test
Test Description:
Checks the STV method of this voting system.

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
This test is done manually by compiling and running “./elect.out” and entering STV when prompted by the console.

Automated: yes__ no _X_

Results: Pass __X___ Fail________

Preconditions for Test:
All files must compile correctly, Shuffling is turned off so we can see consistent results

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Check if STV works with 3 seats.	Number of candidates (9), number of seats (3), number of ballots (11), and ex.csv.	Displays the winner is [A], [D], [H]	as expected	Pass
1	Check if STV	Number of	This should in theory, return 6 winners	Winners are [A], [G], [D],	The droop will be 2

	works with 6 seats.	candidates (9), number of seats (6), number of ballots (11), and ex.csv.		[C], [H] Only 5, instead of 6.	and there are 6 seats, this means that 12 Ballots are required to fill all these seats. However, only 11 ballots are provided. We are uncertain on how to handle this problem and for now assume that displaying only all the winners possible (5) is the best solution.

Post condition(s) for Test:

The console displays the correct winners.