Project Name: Project 1: Voting System **Team# 09**

Test Stage: Unit ✔ System __ **Test Date:** 21 March 2018

Test Case ID#: CandidateTest_constructor_01 **Name(s) of Testers:** Shalom Nguyen

Test Description: To test the default values in the candidate object constructor and the getters/setters to private data of said object.

The unit tests are in the unittest.cc in the testing folder. Once user runs make, a directory is created in the testing folder. Traverse the directory (/build/bin) and execute the executable (unittest).

Automated: yes ✔ no ___

Results: Pass ✔ Fail_____

Preconditions for Test:

1. default values are set in the constructor
    a. if the default values in the constructor are altered, the unit test must be accommodated.
    b. expected values are fixed constants.
    c. to pass the test, these changes must be accounted for.

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

1	Instantiate the class object	default values of the default constructor in candidate.cc	candidate_name_ == "";
			num_ballots_ == 0;
			isWinner == false;
				candidate_name_ == "";
				num_ballots_ == 0;
				isWinner == false;
					the member data, ballot_list_ is a pointer to a Ballot object and therefore, cannot be actually tested.

2	calling object's setters	candidate_name_ = "Test";
		num_ballots_ = 99;
		isWinner = true;
			candidate_name_ == "Test";
			num_ballots_ == 99;
			isWinner == true;
				candidate_name_ == "Test";
				num_ballots_ == 99;
				isWinner == true;	

Post condition(s) for Test:

In its current state, the object, in this case, candidate, now has been assigned the above values (2). However, after testing, the object is destroyed via destructor and therefore, the system state remains the same.

Project Name: Project 1: Voting System **Team# 09**

Test Stage: Unit ✔ System __ **Test Date:** 21 March 2018

Test Case ID#: CandidateTest_toString_02 **Name(s) of Testers:** Shalom Nguyen

Test Description: To test the toString function.

The unit tests are in the unittest.cc in the testing folder. Once user runs make, a directory is created in the testing folder. Traverse the directory (/build/bin) and execute the executable (unittest).

Automated: yes ✔ no ___

Results: Pass ✔ Fail_____

Preconditions for Test:

1. var candidate_name_ exists

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

1	Instantiate the class object	default values of the default constructor in candidate.cc	candidate_name_ == "";
			num_ballots_ == 0;
			isWinner == false;
				candidate_name_ == "";
				num_ballots_ == 0;
				isWinner == false;
					the member data, ballot_list_ is a pointer to a Ballot object and therefore, cannot be actually tested.

2	Calling object's setCandidate_name	candidate_name_ = "QWERTY";	candidate_name_ == "QWERTY";	candidate_name_ == "QWERTY";	
3	Test toString	candidate_name_	candidate_name_ == "QWERTY";	candidate_name_ == "QWERTY";	toString is a getter(: candidate_name_)
4	Return to default value	candidate_name_ = "";	candidate_name_ == "";	candidate_name_ == "";	

5	Test values are as expected	candidate object	candidate_name_ == "";
			num_ballots_ == 0;
			isWinner == false;
				candidate_name_ == "";
				num_ballots_ == 0;
				isWinner == false;	

Post condition(s) for Test:

After testing, the object is destroyed via destructor and therefore, the system state remains the same.

Project Name: Project 1: Voting System **Team# 09**

Test Stage: Unit ✔ System __ **Test Date:** 21 March 2018

Test Case ID#: CandidateTest_toStringWithVotes_03 **Name(s) of Testers:** Shalom Nguyen

Test Description: To test the toStringWithVotes

The unit tests are in the unittest.cc in the testing folder. Once user runs make, a directory is created in the testing folder. Traverse the directory (/build/bin) and execute the executable (unittest).

Automated: yes ✔ no ___

Results: Pass ✔ Fail_____

Preconditions for Test:

1. Initialization: candidate_name_, num_ballots_, and ballot_list_

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

1	Instantiate the class object	default values of the default constructor in candidate.cc	candidate_name_ == "";
			num_ballots_ == 0;
			isWinner == false;
				candidate_name_ == "";
				num_ballots_ == 0;
				isWinner == false;
					the member data, ballot_list_ is a pointer to a Ballot object and therefore, cannot be actually tested.

2	calling object's setters	candidate_name_ = "default"; num_ballots_ = 1;	candidate_name_ == "default"; num_ballots_ == 1;	candidate_name_ == "default"; num_ballots_ == 1;	

3	test toStringWithVotes	Ballot* someBallot = new Ballot;
		someBallot->setBallot_id(1);
		ballot_list_ = someBallot;
			"default: 1"	"default: 1"	

4	Return to default values	candidate_name_ = ""; num_ballots_ = 0;	candidate_name_ == ""; num_ballots_ == 0;	candidate_name_ == ""; num_ballots_ == 0;	

5	Test values are as expected	candidate object	candidate_name_ == "";
			num_ballots_ == 0;
			isWinner == false;
				candidate_name_ == "";
				num_ballots_ == 0;
				isWinner == false;	

Post condition(s) for Test:

After testing, the object is destroyed via destructor and therefore, the system state remains the same. There may exist a Ballot* to someBallot even after testing.