Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit _X_ System __

Test Date: 03/24/23

Test Case ID#: 1

Name(s) of Testers: Shivali Mukherji, Micheal Vang

Test Description: IR Candidate Tests

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

Automated: yes_X__ no ___

Results: Pass _____ Fail X____

Preconditions for Test: test ballots of class Ballot needs to be created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	ConstructorTest	IRCandidate("name")	"name"	"name"	
3	serNumBallotsTest	setNumBallots(5) setNumBallots(1) setNumBallots(3)	5 1 3	5 1 3	
4	setBallotListTest	ballotList = {b1, b2, b3}	{b1, b2, b3}	{b1, b2, b3}	
5	addBallotTest	Ballot(1, {1,2,3}) IRCandidate.addBallot(Ballot) test_candidate.getBallotList() test_candidate.getBallotList().back() test_candidate.getNumBallots()	{b1, b2, b3, b4} .back() = {b4} numBallots = 4	{b1, b2, b3, b4} .back() {b4} numBallots = 4	Setter functions need to account for adding to the variable
		IRCandidate("Canid") B1 = (1, {1,2}) B2 = (1. {2.1}) Canid.addBallot(&B1); Canid.addBallot(&B2); Canid.getNumBallots() Canid.popBallot()	numBallots = 2 popped = {2,1} numBallots = 1	numBallots = 2 popped = {2,1} numBallots = 1	
6	addAndPopBallot				

Post condition(s) for Test: IRCandidate can be created with the ability to track the number of ballots, its mapping, and its name.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 2

Name(s) of Testers: Shivali Mukherji

Test Description: IR Ballot Tests

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test: methods must be validated in order for IRBallot object to be created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	ConstructorTest	cans{c1,c2,c3,c4} s_map {1,2,3,4} map_% {0.25, 0.5, 0.75, 0.1} IRCandidate {cans, s_map, map_%)	{c1,c2,c3,c4} {1,2,3,4} {0.25,0.5,0.75,0.1}	{c1,c2,c3,c4} {1,2,3,4} {0.25,0.5,0.75,0.1}	
3	setGetCandidatesTest	setCandidates(), getCandidates() IRCandidates.size()	numCandidates = 4 IRCandidates = {c1,c2,c3,c4}	candidates vector can be set and returned	
4	setMapPercentageTest	setMapPercentage(sample_map_percentage) ir_test_ballot.getMapPercentage()	sample_map_percentage = {0.25, 0.5, 0.75, 0.1}	sample_map_percentage = {0.25, 0.5, 0.75, 0.1}	
5	setGetNumCandidatesTest	setNumCandidates(4), setNumCandidates(2), setNumCandidates(6), getNumCandidates()	4 2 6	4 2 6	

Post condition(s) for Test: IRBallot object is created and can be used throughout the program. The IRBallot will contain the map percentage for each candidate in a list, the number of ballots for each candidate in a list, and a list of candidates.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 3

Name(s) of Testers: Micheal Vang

Test Description: Ballot Tests

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

Automated: yes X no

Results: Pass X Fail

Preconditions for Test: methods must be validated in order for Ballot object to be created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	ConstructorTest	Ballot(1, {1,2,3,4})	Rank = 1 Mapping = {1,2,3,4}	Rank = 1 Mapping = {1,2,3,4}	
2	setInvalidRankingTest	.setRank(-1) .setRank(0) .setRank(5)	rank = 1 rank = 1 rank = 4	rank = 1 rank = 1 rank = 4	
3	setValidRankingTest	.setRank(1) .setRank(3) .setRank(4)	rank = 1 rank = 3 rank = 4	rank = 1 rank = 3 rank = 4	
4	increaseRankTest	.setRank(3) .increaseRank()	rank = 4	rank = 4	
5	getAndSetIndex	.setRank(3) int index = .getIndex()	rank = 3 index = 2	rank = 3 index = 2	
6	ConstructorWithParameters	.getRank(), .getMapping(), vector<int> mapping{1, 2, 3}	rank = 1 Mapping = {1,2,3}	rank = 1 Mapping = {1,2,3}	
7	GetRank	vector<int> mapping{1, 2, 3}, .getRank()	rank = 1	rank = 1	
8	setGetMapping	std::vector<int> mapping1 = {1, 2, 3}; std::vector<int> mapping2	mapping = {4, 5, 6}	mapping = {4, 5, 6}	

		= {4, 5, 6}; .setMapping(mapping2), .getMapping() Ballot(1, mapping1)			

Post condition(s) for Test: Ballot object is created and can be used throughout the program. The Ballot will containing a ranking for each candidate, as well as the mapping for each candidate.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 4

Name(s) of Testers: Matin Horri

Test Description: AuditFile Tests

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test: methods must be validated in order for AuditFile object to be created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	ProductFileTest	open(),produceFile(), labelFile(“Test_File”), write(“line 1”)	“line 1”	“line 1”	
3	WriteTest_one	open(), write(“line 1”), write(“line 2”)	“line 1” “line 2”	“line 1” “line 2”	
4	DefaultConstructor	audit.getName()	“audit”	“audit”	
5	ConstructorWithArgs	audit(“test”)	“test”	“test”	
6	LabelFile	audit(“test”), labelFile(“new_test)	“new_test”	“new_test”	
7	Open	audit(“test”), open(), close()	“TRUE”	“TRUE”	
8	Close	audit(“test”), open(), close()	“FALSE”	“FALSE”	
9	ProduceFile	audit(“test”), open(),	“Hello, World!”	“Hello, World”	

		write(“Hello, World!”), produceFile(), Close(), file(“test.txt”)			
10	SetAndGetFile	fopent(“test.txt), setFile(f),fclose(f)	“test.txt”	“test.txt”	
11	SetAndGetOutputResult	setOutputResult(test output), getOutputResult()	“test output”	“test output”	
12	SetAndGetFileName	setName(name), setFileName(filename), getName(), getFileName()	“test” “test.txt”	“test” “test.txt”	
13	GetFileStream	getFileStream() .good()	“TRUE”	“TRUE”	

Post condition(s) for Test: Auditfile object is created and can be used throughout the program. The Auditfile is produced after every election. This class allows the program to produce the file, label the file, and write to the file.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 5

Name(s) of Testers: Wenjing Jiang

Test Description: CPLBallotTests

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

Automated: yes X no

Results: Pass X Fail

Preconditions for Test: A file input has been given and CPLProcessing has processed the information

Tests	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	GetPartiesTests	"Democratic" "New Wave" test_cplParties[2]->getName() test_cplParties[0]->getName()	"Democratic" "New Wave"	"Democratic" "New Wave"	
2	getMapAllocatedSeatTests	mapAllocatedSeat = test_cplballot->getMapAllocatedSeat() mapAllocatedSeat[0] & [2]	0, 2,	0, 2	
3	setMapAllocatedSeatTest	int 1	1	1	
4	getMapRemainSeatTest	CPLBallot -> getMapRemainSeat()	mapRemainSeat[0] = 3 mapRemainSeat[2] = 2	3,2	
5	getSeatsTest	seat = CPLBallot->getSeats()	3	3	
6	setSeatsTest	test_cplBallot->setSeats(10); test_cplBallot->getSeats()	10	10	
7	getMapBallotTest	CPLBallot -> getMapBallot() mapBallot[0] mapBallot[2]	mapBallot[0] = 3 mapBallot[2] = 0	3 0	
8	getNumPartiesTest	CPLBallot -> getNumParties()	6	6	
9	getQuotaTest	Ballot -> getQuota()	3	3	

Post condition(s) for Test: CPLBallot object holds the information correctly and can be used in a CPLVoteSystem.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 6

Name(s) of Testers: Wenjing Jiang

Test Description: CPLProcessingTests

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test: A file has been inputed that CPLProcessing can read

Tests	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	outputBallot	CPLProcess -> output() [CPLBallot] typeid(*cplBallot)	Type CPLBallot Object	Type CPLBallotObject	

Post condition(s) for Test: CPLProcessing outputs a CPLBallot that can be used for CPLVoteSystem

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 7

Name(s) of Testers: Wenjing Jiang

Test Description: CPLVoteSystemTests

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

Automated: yes X no

Results: Pass Fail X

Preconditions for Test: methods must be validated in order for AuditFile object to be created

Tests	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	startElectionTest	CPLVoteSystem->StartElection	True	True	
2	conductElectionTest	CPLVoteSystem->ConductElection	True	True	
3	getWinnerTest	ConductElection() getWinner()	“Foster”	“Green”	Both candidate should be winner. Testing needs to be revamped.

Post condition(s) for Test: CPLVoteSystem has successfully conducted its election and a winner is declared.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 8

Name(s) of Testers: Matin Horri

Test Description: SpecialCase Tests

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

Automated: yes X no

Results: Pass X Fail

Preconditions for Test: If there is no clear majority in IR between only two candidates, then popularity will win between the two. Also whenever there is a tie situation that has occurred between 2 things, i.e., a candidate or party, a fair coin will be tossed and determine who's the winner.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	tieBreakerValidGeneraor	t.run()	SUCCEED if tiebreaker result is less than 2	SUCCEED if tiebreaker result is less than 2	
2	tieBreakerRunSize	t.run(3)	SUCCEED if tiebreaker size is 3	SUCCEED if tiebreaker size is 3	
3	popularityCase	p.run(), "Tom", .getName()	winning candidate is "Tom" if p.run is 1	winning candidate is "Tom" if p.run is 1	
4	poptie	setNumBallots(20), p.run()	popularity case returns 100 + setNumBallots(20) if there is a tie	popularity case returns 100 + setNumBallots(20) if there is a tie	

Post condition(s) for Test: SpecialCase is implemented in cases where there was a tie situation that has occurred between 2 things in the CPL , and coin chose the winner, or popularity case for IR and popularity won the election between the two candidate.

Test Stage: Unit X__ System __

Test Date: 03/25/23

Test Case ID#: 9

Name(s) of Testers: Matin Horri

Test Description: Display Tests

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

Automated: yes X no

Results: Pass _X____ Fail_____

Preconditions for Test: methods must be validated in order for Display object to be created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Overwrites	overWrite("test output",overWrite(“new output”), display.prtin()	“new output”	“new output”	
2	EmptyOutput	display.display(""), display.print()	“”	“”	
3	MultipleOverwrites	display.overWrite("first overwrite"), overWrtie (“second overwrite”), print()	“second overwrite”	“second overwrite”	
4	LongOutput	display.overWrite(large_output);	large_output	large_output	
5	EmptyOutput	`display.overWrite(""); display.print()`	SUCCEED if the result is an empty string	SUCCEED	
6	RandomOutput	display.overWrite(random_output), display.print()	random_ouptput	random_output	
7	OutputEquality	Display display1("test output"); Display display2("test output"); display1.print() display2.print()	“test output” “test output”	“test output” “test output”	

8	DefaulConstructor	Display display1(“”)	“ ”	“ ”	
9	InitializationConstructorTest	display(“Matin”)	“Matin”	“Matin”	
10	PrintTest	overwrite(“Matin”), print()	“Matin”	“Matin”	
11	GetOutputTerminalTest	display(“terminal”),getOutput Terminal()	“terminal1”	“terminal1”	
12	SetOutputTerminalTest	display(“Example”), setOutputTerminal(“Matin”), getOutputTerminal()	“Matin”	“Matin”	

Post condition(s) for Test: DisplayCase is implementedm, and the result is shown into the terminal.

Test Stage: Unit ___ System _X_

Test Date: 03/25/23

Test Case ID#: 10

Name(s) of Testers: Michael Vang

Test Description: IR Vote System

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

Automated: yes X no

Results: Pass Fail X

Preconditions for Test: Test ballots must be created before running tests.

Steps	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	startElectionTest	ir_testVote->startElection()	expect true	true	
2	getWinnerTest	ir_testVote->getWinner(), Winner.getName()	expected result is “Rosen (D)”	“Rosen (D)”	
3	simpleGetSetWinner	IRCandidate("Johnny") setWinner(winner) ir_testVote->getWinner().getName(	“Johnny”	“Johnny”	
4	getSetProcessedBallot	IRCandidate* can1 = new IRCandidate("John"); IRCandidate* can2 = new IRCandidate("Doe"); IRCandidate* can3 = new IRCandidate("Sally"); std::vector<IRCandidate*> candidates = {can1, can2, can3}; std::vector<int> sample_map = {1,2,3}; std::vector<double> sample_map_percntage = {0.17, 0.33, 0.5}; new IRBallot(candidates, sample_map,			

		sample_map_percntage) ir_testVote->setProcessedBallot(test) ir_testVote->getProcessedBallot()			

Post condition(s) for Test:
An election has been run for IR and winner has been elected.

Test Stage: Unit ___ System _X_

Test Date: 03/25/23

Test Case ID#: 11

Name(s) of Testers: Michael Vang

Test Description: IR Processing

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

Automated: yes__X__ no ____

Results: Pass Fail X

Preconditions for Test: A file path must be provided for the test to run.

Steps	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	startElectionTest	ir_testVote->startElection()	expect true	true	
2	getWinnerTest	ir_testVote->getWinner(), Winner.getName()	expected result is “Rosen (D)”	“Rosen (D)”	
3					

Post condition(s) for Test:
IRProcessing was able to process the provided file path.

Test Stage: Unit _X_ System ___

Test Date: 03/25/23

Name(s) of Testers: Wenjing Jiang, Micheal Vang

Test Case ID#: 12

Test Description: Party

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

Automated: yes X no

Results: Pass __X___ Fail _____

Preconditions for Test: Party class must be defined.

Steps	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	setNameTest	testParty.setName("Republican"), testParty.getName()	name of party that is set is equal to testParty.getName()	name of party that is set is equal to testParty.getName()	
2	getNameTest	testParty.getName()	testParty.getName() returns name of party (“Democratic”)	testParty.getName() returns name of party (“Democratic”)	
3	getCandidateTest	testParty.getCandidate()[0], testCanditate.getName()	testCanditate.getName() returns “Foster”	testCanditate.getName() returns “Foster”	
4	setCandidateTest	testParty.setCandidate(candidate2), testParty.getCandidate()[1], testCanditate.getName()	testCanditate.setName() returns sets candidate2 and returns candidate2 (“Volz”)	testCanditate.setName() returns sets candidate2 and returns candidate2 (“Volz”)	
5	getMaxSeatTest	testParty.getMaxSeat()	testParty.getMaxSeat() is 0	testParty.getMaxSeat() is 0	
6	setMaxSeatTest	testParty.setMaxSeat(2); testParty.getMaxSeat()	maxSeat is set as 2, and testParty.getMaxSeat() returns 2	maxSeat is set as 2, and testParty.getMaxSeat() returns 2	

Post condition(s) for Test:
Party class can be constructed that correctly contains information.

Test Stage: Unit _X_ System __

Test Date: 03/25/23

Name(s) of Testers: Wenjing Jiang, Micheal Vang

Test Case ID#: 13

Test Description: Candidate

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

Automated: yes X no

Results: Pass X Fail

Preconditions for Test: Candidate class must be defined.

Steps	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	.setName("Mike")	.setName("Mike") .getName() setParty("Demo") .getParty()	name = Mike party = Demo	name = Mike party = Demo	
2	constructer	Candidate("Dolly", "Repo") .getName() .getParty()	name = Dolly party = repo	name = Dolly party = repo	

Post condition(s) for Test:
Candidate class can be constructed and correctly hol information.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System1 **Name(s) of Testers: Micheal Vang**

Test Description: IR standard case (ir.csv)

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

Automated: yes____ no _X_

Results: Pass __X_____ Fail _____

Preconditions for Test: testing/data/IR.csv must exist and Main was be compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main		Welcome screen prompt	Welcome screen prompt	
2	read testing/data/IR.csv		file gets processed and display the steps	file gets processed and display the steps	
3	Winner is given		Chou (I)	Chou (I)	
4	Auditfile produced		“IRAudit.csv” in the same directory	“IRAudit.csv” in same directory	
1.5	Run ./main testing/data/IR.csv	IR.csv	Stats / steps given and audit file produced in directory “IRAudit.csv” Winner is Chou(I)	Stats / steps given and audit file produced in directory “IRAudit.csv” Winner is Chou(I)	

Post condition(s) for Test: IR election successfully conducted with the winner given and audit file produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System2 **Name(s) of Testers: Micheal Vang**

Test Description: IR Eliminate Tie (IREliminateTie.csv)

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

Automated: yes____ no _X_

Results: Pass_____ Fail X__

Preconditions for Test: testing/data/IREliminateTie.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/IREliminateTie.csv	IREliminateTie.csv	Stats / steps given and audit file produced in directory "IRAudit.csv" Tiebreaker is activated for candidate with lowest ballot. Candidate selected is either or. Winner is Rosen(D)	Stats / steps given and audit file produced in directory "IRAudit.csv" Tiebreaker is activated for candidate with lowest ballot. **Candidate selected is only one person through multiple outputs.** **Same Candidate gets eliminated twice**. Winner is Rosen(D)	There's a bug going on with Tiebreaker and how IRVoteSystem is using it.

Post condition(s) for Test: IR election successfully conducted with the case of tiebreaker of 2 people of candidate with the lowest votes. The winner is given and the audit file is produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System3
Test Description: IR Eliminate Tie 3 way (IREliminateTie3Way.csv)

Name(s) of Testers: Micheal Vang

Indicate where are you storing the tests (what file) and the name of the method/functions being used.
/testing/system_test/ir_eliminate_tie_3way

Automated: yes_____ no _X_

Results: Pass_____ Fail___X___

Preconditions for Test: testing/data/IREliminateTie.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/IREliminateTie3way.csv	IREliminateTie3way.csv	Stats / steps given and audit file produced in directory "IRAudit.csv" Tiebreaker is activated for candidate with lowest ballot. Candidate selected is either or out of three. Winner is Rosen(D)	Stats / steps given and audit file produced in directory "IRAudit.csv" Tiebreaker is activated for candidate with lowest ballot. **Candidate selected is only one person through multiple outputs.** **Same Candidate gets eliminated twice**. Winner is Rosen(D)	There's a bug going on with Tiebreaker and how IRVoteSystem is using it.

Post condition(s) for Test: IR election successfully conducted with the case of tiebreaker of 3 candidates with the lowest votes. The winner is given and the audit file is produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System4
Test Description: IR Popularity Case (IRPop.csv)

Name(s) of Testers: Micheal Vang

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

Automated: yes____ no _X_

Results: Pass _X___ Fail_____

Preconditions for Test: testing/data/IRPop.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/IRPop.csv	IRPop.csv	Stats / steps given and audit file produced in directory "IRAudit.csv" Popularity case is indicated for the winner Winner is Rosen(I)	Stats / steps given and audit file produced in directory "IRAudit.csv" Popularity case is indicated for the winner Winner is Rosen(I)	

Post condition(s) for Test: IR election successfully conducted with the case of popularity with the audit file produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__

Test Date: 03/29/23

Test Case ID#: System5
Test Description: IR Popularity Tie Case (IRPopTie.csv)

Name(s) of Testers: Micheal Vang

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

Automated: yes____ no _X_

Results: Pass ___ Fail_X__

Preconditions for Test: testing/data/IRPopTie.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/IRPopTie.csv	IRPopTie.csv	Stats / steps given and audit file produced in directory "IRAudit.csv" Popularity case is indicated for the winner Tiebreaker indicated for winner Winner is Rosen(I) or Chou(I)	Stats / steps given and audit file produced in directory "IRAudit.csv" Popularity case is indicated for the winner Tiebreaker indicated for winner **Winner is only Chou(I)**	There's a bug going on with Tiebreaker and how IRVoteSystem is using it.

Post condition(s) for Test: IR election successfully conducted with the case of popularity tiebreaker with audit file produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System6
Test Description: CPL (CPL.csv)

Name(s) of Testers: Micheal Vang

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

Automated: yes____ no _X_

Results: Pass _X_ Fail___

Preconditions for Test: testing/data/CPLLottery.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/CPL.csv	CPL.csv	Stats / steps given and audit file produced in directory "CPL.csv" Winner is Green, McClure, Peters (Republican, Reform, Independent)	Stats / steps given and audit file produced in directory "CPL.csv" Winner is Green, McClure, Peters (Republican, Reform, Independent)	

Post condition(s) for Test: Election for CPL is successfully conducted with the seats of the winner. An audit file is also produced.

Project Name: Project 1: Voting System **Team#24**

Test Stage: Unit ___ System __X__ **Test Date: 03/29/23**

Test Case ID#: System7
Test Description: CPLLottery (CPLLottery.csv)

Name(s) of Testers: Micheal Vang

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

Automated: yes____ no _X_

Results: Pass ___ Fail_X_

Preconditions for Test: testing/data/CPLLottery.csv must exist and Main compiled.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Run ./main testing/data/CPLLottery.csv	CPLLottery.csv	Stats / steps given and audit file produced in directory "CPL.csv" Winner is Foster(Democratic) The other 2 seats are randomly given to 2 different candidate/party.	Stats / steps given and audit file produced in directory "CPL.csv" Winner is Foster(Democratic) Volz (Republican) Sally (Reform) for all outputs.	Need to implement random seed in CPL Lottery code.

Post condition(s) for Test: Election for CPL is successfully conducted with the case of lottery. An audit file is also produced.