CSCI 5801 Project 1
Vote Counting System Testing Document
Version 1.0
Team 31
11/18/2019
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

not Table of Contents

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date:

Name(s) of Testers:

Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_1

Test Description: Test Main function in ElectionDriver. The test is being stored in the ElectionDriverTest.java file. It uses the Maven Junit suite. The functionality that will be testing is the parsing of the given file from the user

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

Automated: yes_x__ no ___

Results: Pass _____ Fail___x_____

Preconditions for Test:

Election file exists in the correct location with the following data:

```
// OPL SETUP INFO
private static File OPL_FILE = new File("./assets/test_opl.txt");
private static String OPL_FILE_NAME = "./assets/test_opl.txt";
private static Election OPL_ELECTION = null;
private static String OPL_ELECTION_TYPE = "OPL";
private static ArrayList<String> OPL_CANDIDATE_LIST = new ArrayList<>();
private static ArrayList<String> OPL_BALOT_LIST = new ArrayList<>();
private static ArrayList<String> OPL_ELECTION_INFO = new ArrayList<>();

// CPL SETUP INFO
private static String CPL_FILE_NAME = "./assets/test_cpl.txt";
private static String CPL_PARTY_LIST = "";
private static File CPL_FILE = new File(CPL_FILE_NAME);
private static String CPL_ELECTION_TYPE = "CPL";
private static Election CPL_ELECTION = null;
```

```
private static ArrayList<String> CPL_CANDIDATE_LIST = new ArrayList<>();
private static ArrayList<String> CPL_BALOT_LIST = new ArrayList<>();
private static ArrayList<String> CPL_ELECTION_INFO = new ArrayList<>();
```

OPL File:

```
OPL
3
9
6
[Pike,D]
[Foster,D]
[Deutsch,R]
[Borg,R]
[Jones,R]
[Smith,I]
1,,,,,
1,,,,,
,1,,,,
,,,,1,
,,,,,1
,,,1,,
,,,1,,
1,,,,,
,1,,,,
```

CPL File:

```
CPL
4
[D,R,G,I]
7
100
16
```

[Pike,D,1]
[Foster,D,2]
[Floyd,D,3]
[Jones,D,4]
[Mallory,D,5]
[Deutsch,R,1]
[Wong,R,2]
[Walters,R,3]
[Keller,R,4]
[Borg,R,5]
[Jones,G,1]
[Smith,G,2]
[Lewis,G,3]
[Smith,G,4]
[Li,G,5]
[Perez,I,1]
1,,,
1,,,
,1,,
,1,,
,,1,
,,,1
,,1,
1,,,
,1,,
,,1,
,,1,
,1,,
,1,,

No actual results - conflict with file stream causes runtime errors. (Errors do not exist if you run main through the driver, just in the test)

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

1	set a String [] args	String [] args = OPL_FILE_NAME			
2	run ElectionDriver.getElectionInputFile()		OPL_FILE data (see precondition)	OPL_FILE data (see precondition)	
3	run ElectionDriver.getElection()		OPL_Election (see precondition)	OPL_Election (see precondition)	
4	run ElectionDriver.getElectionType()		OPL	OPL	
5	run ElectionDriver.getFileName()		OPL_FILE_NAME (see precondition)	OPL_FILE_NAME (see precondition)	
6	call buildAssets(OPL_FILE, OPL_CANDIDATE_LIST, OPL_BALOT_LIST, OPL_ELECTION_INFO);				
7	call ElectionDriver.getCandidateList()		OPL_CANDIDATE_LIST (see precondition)	OPL_CANDIDATE_LIST (see precondition)	
8	call ElectionDriver.getBallotList()		OPL_BALOT_LIST (see precondition)	OPL_BALOT_LIST (see precondition)	

9	call ElectionDriver.getElectionInformation()		OPL_ELECTION_INFO (see precondition)	OPL_ELECTION_INFO (see precondition)	
10	call ElectionDriver.clearDriver();				

11	set a String [] args	String [] args = CPL_FILE_NAME			
12	run ElectionDriver.getElectionInputFile()		CPL_FILE data (see precondition)	CPL_FILE data (see precondition)	
13	run ElectionDriver.getElection()		CPL_Election (see precondition)	CPL_Election (see precondition)	
14	run ElectionDriver.getElectionType()		CPL		
15	run ElectionDriver.getFileName()		CPL_FILE_NAME (see precondition)	CPL_FILE_NAME (see precondition)	
16	call buildAssets(CPL_FILE, CPL_CANDIDATE_LIST, CPL_BALOT_LIST, CPL_ELECTION_INFO);				

17	call ElectionDriver.getCandidateList()		CPL_CANDIDATE_LIST (see precondition)	CPL_CANDIDATE_LIST (see precondition)	
18	call ElectionDriver.getBallotList()		CPL_BALOT_LIST (see precondition)	CPL_BALOT_LIST (see precondition)	

Post condition(s) for Test:

File is set and read for Input stream

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_2
Test Description:
OPL's PickWinners method is tested. The test is being stored in the OPLTest.java file.

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
Candidate, Ballot, and Party class are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	add Candidates to list	ArrayList<Candidates>			
3	add Parties to list	ArrayList<Parties>			
4	setNumberOfSeaters	void			
5	call PickWinners()	ArrayList<Candidates>	list of [c1,c2,c4,c5]	list of [c1,c2,c4,c5]	

Post condition(s) for Test:
PickWinners pick correct winners excluding handling ties

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_3
Test Description: coinFlipOPL - OPL's tie case test
The test is being stored in the OPLTest.java file..

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
Candidate, Ballot, and Party class and coinFlip method are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	add Candidates to list	ArrayList<Candidates>			
3	add Parties to list	ArrayList<Parties>			
4	setNumberOfSeaters	void			
5	call PickWinners() 100 times	ArrayList<Candidates>		list of [c1,c2,c4,c5]	
6	increase number of times if candidate is contained in winner list	int			

7	compare if each win rate is >= 35	boolean	True	True	

Post condition(s) for Test:

PickWinners method picked correct winners in 2 way tie

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_4
Test Description: RunGeneral - OPL's general test
The test is being stored in the OPLTest.java file.

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
Candidate, Ballot, and Party class and coinFlip method are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	add Candidates to list	ArrayList<Candidates>			
3	add Parties to list	ArrayList<Parties>			
4	setNumberOfSeaters	void			
5	add expected winners to Expected list	ArrayList<Candidates>			
6	call Run()	ArrayList<Candidates>		list of [c1,c2,c4,c5]	
7	store PickWinners( ) in Actual list	ArrayList<Candidates>			
8	check if each candidate in Actual is in	boolean	True	True	

	Expected winner list				
9	check if candidate that is lost is not in the actual list	boolean	false	false	

Post condition(s) for Test:

OPL test works in general without ties

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_5
Test Description: CPL's coin flip test
The test is being stored in the CPLTest.java file.

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

Automated: yes_x__ no ___

Results: Pass _____ Fail__x_____

Preconditions for Test:
Candidate, Ballot, and Party class and coinFlip method are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	add Candidates to list	ArrayList<Candidates>			
3	add Parties to list	ArrayList<Parties>			
4	setNumberOfSeaters	void			
5	call PickWinners() 100 times	ArrayList<Candidates>		list of [c1,c2]	
6	increase number of times if candidate is contained in winner list	int			
7	compare if each win rate is	boolean	True	False	

	>= 35				

Post condition(s) for Test:

PickWinners can handle 3 way ties

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_6
Test Description: CPL - DistributeSeatTest
The test is being stored in the CPLTest.java file.

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

Automated: yes_x__ no ___

Results: Pass _x____ Fail_____

Preconditions for Test:
Candidate, Ballot, and Party class and coinFlip method are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	add Candidates to list	ArrayList<Candidates>			
3	add Parties to list	ArrayList<Parties>			
4	call setPartyVote( ) on each party	void			
5	call DistributeSeats( ) method	void			
6	check if each party receive the expected number of seats by calling	integer	[2,2,1,1,4]	[2,2,1,1,4]	

	getNumberOfSeats method on each party				

Post condition(s) for Test:

DistributeSeats method works without ties

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_7
Test Description: CPL - run
The test is being stored in the CPLTest.java file.

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

Automated: yes_x__ no ___

Results: Pass _x____ Fail_____

Preconditions for Test:
Candidate, Ballot, and Party class and coinFlip method are fully tested

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	create ballots and add ballots to ballot lists	ArrayList<Ballot>			
3	create parties and add parties to partyList	ArrayList<Party>			
4	construct expected winner list	ArrayList<Candidate>			
5	call getPartyVotes method to check if Count method works	int	6,12,2	6,12,2	
6	call getNumberOfSeats	int	3,6,1	3,6,1	

	method to check if DistributeSeat method works				
7	store winners from PickWinners( ) in Actual list	ArrayList<Candidate>			
8	check if each candidates in Actual list exists in Expected list	boolean	True	True	
9	check if candidates are lost not in Actual list	boolean	False	False	

Post condition(s) for Test:

CPL Run method distribute seats and pick winners correctly without tie cases

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_8
Test Description:
IncreaseVoteCount() in Candidate class
Test is in testing/CandidateTest.java,
function void IncreaseVoteCount()

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

Automated: yes_x__ no ___

Results: Pass _x____ Fail_____

Preconditions for Test:
new instance of Candidate with any name and party
score exists in Candidate

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	create a new candidate with any name and party, and no score	New Candidate(name, PartyString)			
2	call candidate.getScore()	int score	0	0	
3	call candidate.IncreaseVoteCount()	int score	void	void	Method increments score and returns voic
4	call candidate.getScore()	int score	1	1	

Post condition(s) for Test:

File is set and read for Input stream

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019

Name(s) of Testers:

Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_9

Test Description:

This tests the functionality of the getName function in the candidate class. This function allows us to look up the name of a given candidate.

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

Automated: yes_x__ no ___

Results: Pass __x___ Fail_____

Preconditions for Test:

name and partyString are correctly defined Strings

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new Candidate	new Candidate("Test", "R")			
2	Assign new Party("R") to Candidate				
3	call Candidate.getName()	candidate.name	"Test"	"Test"	

Post condition(s) for Test:

Candidate has been made and has an accessible name

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_10
Test Description:
This test is used to test the functionality of the getScore function in the candidate class. This allows up to run statistics on the candidates now knowing their score

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

Automated: yes__X_ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
name and partyString are correctly defined Strings
score exists in Candidate

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	create a new candidate with any name and party, and no score	New Candidate(name, PartyString)			
2	Call candidate.assignParty (new Party(partyString));				
3	call candidate.getScore()	candidate.score	0	0	

Post condition(s) for Test:

File is set and read for Input stream

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_11
Test Description:
This tests the functionality of the getParty function in the candidate class. The test is being stored in the CandidateTest.java file.

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
name and partyString are correctly defined Strings

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new Candidate with any name and the partyString "R"	String "R" Candidate candidate			
2	call candidate.assignParty (partyString)				
3	Call candidate.getParty()	candidate party	R	R	
4					

Post condition(s) for Test:
File is set and read for Input stream

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _X_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_12
Test Description:
This is used to test the functionality of the parties addVote method. The test is being stored in the PartyTest.java file.

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

Automated: yes__X_ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:
A new party has been created
The constructor initialize the vote count to 0

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a party object	party object			
2	Assert the vote equals 0	partyVotes	0	0	
3	run the addvote method	partyVotes			
4	Assert the vote equals 1		1	1	

Post condition(s) for Test:
Party vote is incremented to 1 higher than initial value

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019

Name(s) of Testers:

Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_13

Test Description:

Test addCandidate is a test to check the addCandidate method in the Party class. The test is being stored in the PartyTest.java file.

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

Automated: yes__X_ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:

Candidate is added with candidate list of zero

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create instance of Party				
2	Add candidate	candidateList			
3	Check that party's candidate list now equals size of 1	candidateList	1	1	
4					

Post condition(s) for Test:

Candidate exists in candidateList within Party class

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_14
Test Description:
Test getSeats is a test to check the getSeats method in the Party class. This is also tested again with the setSeats test. The test is being stored in the PartyTest.java file.

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

Automated: yes__X_ no ___

Results: Pass ___X__ Fail_____

Preconditions for Test:
A new instance of party class is created with the constructor and it has a seat number initialized

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create a new instance of the Party Class				
2	Run the get seats method	Party.seats	0	0	

Post condition(s) for Test:
Party has seats and seat number is returned

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_15
Test Description:
Test setSeats is a test to check the setSeats method in the Party class. Backup test of getSeats(). The test is being stored in the PartyTest.java file.

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

Automated: yes__X_ no ___

Results: Pass ___X__ Fail_____

Preconditions for Test:
Instance of party is created with a valid seat number

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create instance of party class				
2	Implement party setSeats to 3	party.seats			
3	Assert getseats =3	party.seats	3	3	This also tests Party.getseats()
4					

Post condition(s) for Test:
The seat number is set to the 3 or a specified number.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_16
Test Description:
Test setPartyVotes is a test to check the setPartyVotes method in the Party class. Backup test of getVotes(). The test is being stored in the PartyTest.java file.

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

Automated: yes_X_ no ___

Results: Pass __X__ Fail_____

Preconditions for Test:
Instance of Party class is created with PartyVotes parameter set to 0

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create instance of party				
2	run the Party.addVote() method	party.Votes			
3	Assert that getvotes = 1	party.votes	1	1	also tests getvotes()
4					

Post condition(s) for Test:
Party has votes and votes are set to 1.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_17
Test Description:
Test getName is a test to check the getName method in the Party class. The test is being stored in the PartyTest.java file.

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

Automated: yes_X_ no ___

Results: Pass __X__ Fail_____

Preconditions for Test:
Party class has a member constructed with a party name

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create party class member with name				
2	Assert that name.equal	party.Name	"jerks"	"jerks"	getName Called
3					
4					

Post condition(s) for Test:
Party has a name correctly entered in constructor and it matches test string.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_18
Test Description:
Test getVotes is a test to check the getVotes method in the Party class. This also is an additional test of getVotes(). The test is being stored in the PartyTest.java file.

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

Automated: yes_X__ no ___

Results: Pass ___X__ Fail_____

Preconditions for Test:
Party class has a member of the class with a vote total = 0.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	create a Party instance				
2	Check that getVotes is 0	party.votes	0	0	
3	call addPartyVote method	party.votes			
4	Check that getvote is 1		1	1	

Post condition(s) for Test:
Party exists with vote and vote number is incremented higher.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_19
Test Description:
Testing of the Results Class.The test is being stored in the ResultsTest.java file.

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

Automated: yes__x_ no ___

Results: Pass ___x__ Fail_____

Preconditions for Test:
Test Results is a member of the Results class and it exists with a class member.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create 3 distinct candidates				
2	Create 3 distinct parties	1,2,3			
3	assign each candidate to a party	r,d,l			
4	increase one candidates votes count OPL	candidate 1 + vote			
5	Check results	candidate 1 1 vote	candidate 1 wins	candidate 1 wins	

Post condition(s) for Test:
Candidate c1 has 1 vote and is the winner in results, OPL.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_20
Test Description:
Test Media results class. The test is being stored in the MediaTest.java file.

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

Automated: yes_x__ no ___

Results: Pass ___x__ Fail_____

Preconditions for Test:
Election information is received and read via stream

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create 3 distict Candidates;	c1,c2,c3			
2	Create 3 distinct Parties	r,d,i			
3	Assign each Candidate to a party.	...			
4	create a new OPL election and add vote	c1 + vote	c1 wins , in report	c1 wins, report exists	
5	Create media report and view.		report checked		results are correct

Post condition(s) for Test:
Media results are correct and exist in correct location.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_21
Test Description:
Test the Audit Log. The test is being stored in the AuditTest.java file.

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

Automated: yes___ no ___

Results: Pass _____ Fail_____

Preconditions for Test:
Election file is in correct location and file is read and run.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create 2 distinct instance of Audit class	a1, a2			
2	Assert, check they are null.	a1,a2	a1 = a2=null	a1=a2=null	
3	Set the audits from Audit.getInstance()	a1,a2			Uses Audit.getInstance()
4	Checks a1, and a2 are not null	a1,a2	a1=a2=not null	a1=a2=not null	
5	Check that a1, a2 are equal.	a1,a2	a1 = a2	a1 = a2	

Post condition(s) for Test:

Audit file is created and matches voting history from file. Format is correct.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019

Name(s) of Testers:

Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_22

Test Description:

CPL- getPartyListTest

The test is being stored in the CPLTest.java file.

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

Automated: yes_X__ no ___

Results: Pass __X___ Fail_____

Preconditions for Test:

Candidate, Ballot, and Party class and coinFlip method are fully tested
score exists in Candidate

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1					
2	create parties and add some party to party list	ArrayList<Party>			
3	create candidates and add each candidate to	ArrayList<Candidate>			

	candidate list				
4	store list got from getPartyList method as actual list	ArrayList<Party>			
4	check if added parties exists by calling getPartyList	boolean	true	true	
5	check if not added parties not exist in actual list	boolean	false	false	

Post condition(s) for Test:

CPL party list is created and displays with correct entries.

Project Name: Project 1: Voting System **Team# 31**

Test Stage: Unit _x_ System __

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_23
Test Description:
Test Ballot class - score
The test is being stored in the BallotTest.java file.

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

Automated: yes__x_ no ___

Results: Pass ___x__ Fail_____

Preconditions for Test:
String exists and is equal to “,,1,,,,”

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Instance of Ballot class is created using the string “,,1,,,,”.	Ballot ballot String “,,1,,,,”			
2	Ballot.score() is called	ballot.raw	2	2	

Post condition(s) for Test:
Ballot has returned a score

Project Name: Project 1: Voting System **Team# 31**

Post condition(s) for Test:
CPL party list is created and displays with correct entries.

Test Stage: Unit __ System _X_

Test Date: 11/16/2019
Name(s) of Testers:
Erin Seichter (seich009)
Griffin Higley (higle018)
Dat Le (le000043)
Dan Belair (bela0019)

Test Case ID#: VCS_24
Test Description:
Test System Class
The test is being stored in the SystemTest.java file.

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

Automated: yes__x_ no ___

Results: Pass _____ Fail____X____

Preconditions for Test:
Nothing has been setup. Conditions to Run all other tests have been met

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Notes
1	Create classes for all other test classes				The reason this test fails is due to a bug in the election driver test class.
2	Sequentially call all of the other test classes test methods				

Post condition(s) for Test:
All tests have passed