| **Project Name: Project 1: Voting System Team#20** | |
| --- | --- |
| **Test Stage: Unit \_X\_ System \_\_** | **Test Date: 3/25/2023** |
| **Test Case ID#: 1** | **Name(s) of Testers: Nabeel Azam** |
| **Test Description:**  Tests for the Party class |  |
| **Automated: yes\_X\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used: Party.java PartyTest.java** |
| **Results: Pass \_\_\_X\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  Object is created of type party | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | test PartyName | Democrat.getParty() | democrat | democrat | Success |
| 2 | test PartyVotes | Democrat.getVotes() | 4 | 4 | Success |
| 3 | test PartyAddVotes | Democrat.addPartyVotes(4) | 8 | 8 | Success |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

Party object is created and correctly modifiable with its class functions.



| | **Project Name: Project 1: Voting System Team#20** | | | --- | --- | | **Test Stage: Unit \_X\_ System \_\_** | **Test Date: 3/26/2023** | | **Test Case ID#: 5** | **Name(s) of Testers: Mohamed Mohamed** | | **Test Description:**  Tests for the IRElection class |  | | **Automated: yes\_\_X\_ no \_\_\_** | **Indicate Weather the IR algorithms is successfully run for the Given input election file. We are storing the File in IRElection.Java and the Test file in IRElectionTest.java** | | **Results: Pass \_\_X\_\_\_ Fail\_\_\_\_\_\_\_\_** |  | |  |  | | **Preconditions for Test:**  input file must already be made and correct directory must be inputted (e.g: | |  | **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** | | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  | | 1 | Prompt user for input | userPrompter() | file location and “CPL” | file location and “CPL” | Prompt user for input | | 2 | check election type | obj.getElectionType() | CPL | CPL | check election type | | 3 | read file and extract data |  | extract parties, numSeats, and numBallots | extracted parties, numSeats, and numBallots | read file and extract data | |  | Test the winner of the election | Candidates(a)  Candidates(b)  Candidates(c)  setWinner(b) | Expected Result: The Winner of the election is returned as "Jack" | Actual Result: The Winner of the election is returned as "Jack" | The test passes | |  | Test the list of candidates | Candidates( a, b, c ) | The first candidate is returned as "Dave"  The second candidate is returned as "Jack"  The third candidate is returned as "Jim" | The first candidate is returned as "Dave"  The second candidate is returned as "Jack"  The third candidate is returned as "Jim" |  |   **Post condition(s) for Test:**    the winner of the election should be set to the expected candidate, and the list of candidates should be returned in the correct order, respectively.                                                                    **Name: Project 1: Voting System Team#20** | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Stage: Unit \_\_ System \_X\_** | **Test Date: 3/26/2023** |
| **Test Case ID#: 2** | **Name(s) of Testers: Danial Syed** |
| **Test Description: System test for CPL implementation**  Tests for the CPLElection and CPLAudit class |  |
| **Automated: yes \_\_X\_\_\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Voting, CPLElection, CPLAudit** |
| **Results: Pass \_X\_\_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  input file must already be made and correct directory must be inputted (e.g: | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | Prompt user for input | userPrompter() | file location and “CPL” | file location and “CPL” |  |
| 2 | check election type | obj.getElectionType() | CPL | CPL |  |
| 3 | read file and extract data | runCPL.readFile | extract parties, numSeats, and numBallots | extracted parties, numSeats, and numBallots |  |
| 4 | allocate seats | firstSeatWaveAllocation() | seats allocated | seats allocated |  |
| 5 | find party with most seats | main() | setWinner(party) | setWinner(party) |  |
| 6 | Display the winner | displayWinner() | Winner is displayed | Winner is displayed |  |
| 7 | Product election files | productAuditCPL()  produceMediaCPL() | Audit.txt and Media.txt and results written | Audit.txt and Media.txt and results written |  |

**Post condition(s) for Test:**



Audit.txt and Media.txt hold results of the election that are viewable. 

| **Project Name: Project 1: Voting System Team#20** | |
| --- | --- |
| **Test Stage: Unit X\_\_ System \_\_** | **Test Date: 3/25/2023** |
| **Test Case ID#: 3** | **Name(s) of Testers: Nabeel Azam** |
| **Test Description:**  Tests for the Candidate class |  |
| **Automated: yes\_X\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used: Candidate.java CandidateTest.java** |
| **Results: Pass \_\_\_X\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  Candidate object created | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | test CandidateName | Dave.getName() | Dave | Dave | Success |
| 2 | test CandidateParty | Dave.getParty() | democrat | democrat | Success |
| 3 | test CandidateRank | Dave.getRank() | 4 | 4 | Success |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

Candidate object successfully implemented with functions from candidate class. 

| **Project Name: Project 1: Voting System Team#20** | |
| --- | --- |
| **Test Stage: Unit \_X\_ System \_\_** | **Test Date: 3/25/2023** |
| **Test Case ID#: 4** | **Name(s) of Testers: Nabeel Azam** |
| **Test Description:**  Tests for the Ballot class |  |
| **Automated: yes\_\_X\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used: Ballot.java BallotTest.java** |
| **Results: Pass \_\_X\_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  Ballot object created | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | test BallotPreference | test.getPreference() | ("a", "b", "c") | ("a", "b", "c") | Success |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

Ballot object successfully implemented with functions from Ballot class. 

| **Project Name: Project 1: Voting System Team#20** | |
| --- | --- |
| **Test Stage: Unit \_X\_ System \_\_** | **Test Date: 3/26/2023** |
| **Test Case ID#: 6** | **Name(s) of Testers: Danial Syed** |
| **Test Description: Manual tests for creating CPL audit files**  Tests for the Ballot class |  |
| **Automated: yes\_\_\_ no X\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used: CPLAudit.java, CPLAuditTest.java** |
| **Results: Pass \_\_\_\_\_ Fail\_\_\_X\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  CPL algorithm has run correctly | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | produce audit file with manual inputs | testProduceAuditCPL() | "Election Conducted: Closed Party List Election \r\n" + "Election Date: " + new Date() + "\r\n"  + "The total votes in this election was: 647\r\n" + "Party and Seat totals: {Democratic=72, Republican=143, New Wave=0, Reform=144, Green=72, Independent=216}\r\n"  + "Party and Vote totals: {Democratic=2, Republican=4, New Wave=0, Reform=4, Green=2, Independent=6}\r\n" + "The Winner of The Election Was Independent!"; | NullPointerException | Election is being run the same way in main(), not entirely sure how this produces and error but main() does not |
| 2 | produce media file with manual inputs | testProduceMediaCPL() | "Election Conducted: Closed Party List Election \r\n" + "Election Date: " + new Date() + "\r\n"  + "Party and Seat totals: {Democratic=72, Republican=143, New Wave=0, Reform=144, Green=72, Independent=216}\r\n" + "The Winner of The Election Was Independent!"; | NullPointerException | ^^same as above test |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

Audit and media files are now viewable for election results.

| **Project Name: Project 1: Voting System Team#20** | |
| --- | --- |
| **Test Stage: Unit \_X\_ System \_\_** | **Test Date: 3/26/2023** |
| **Test Case ID#: 7** | **Name(s) of Testers: Danial Syed** |
| **Test Description: Manual tests CPLElection algorithm** |  |
| **Automated: yes\_\_\_ no \_\_X\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used: CPLElection.java, CPLElectionTest.java** |
| **Results: Pass \_\_\_\_\_ Fail\_\_\_\_X\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  CPL is the given election type and valid file is inputted | |

| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 1 | return total available seats | testTotalSeats | 20 | NullPointerException |  |
| 2 | return winning party | testWinner | “Independent” | NullPointerException |  |
| 3 | return each party along with the number of votes it received | testPartyVotes | "{Democratic=72, Republican=143, New Wave=0, Reform=144, Green=72, Independent=216}" | NullPointerException |  |
| 4 | return each party along with the number of seats it won | testPartySeats | "{Democratic=2, Republican=4, New Wave=0, Reform=4, Green=2, Independent=6}" | NullPointerException |  |
| 5 | return each party along with the remainder of votes it had after its first allocation | testPartyRemainders | "{Democratic=8, Republican=15, New Wave=0, Reform=16, Green=8, Independent=24}" | NullPointerException |  |
| 6 | return the quota using total votes / numSeats | testQuota | 32 | NullPointerException |  |
| 7 | return the overall total votes in the election | testTotalVotes | 647 | NullPointerException |  |
| 8 | randomly choose either 1 or 2 | testWinnerTieDecider | 1 or 2 | NullPointerException |  |

**Post condition(s) for Test:**

Go on to create an audit and media file which stores results of the election.

## 

**Project Name:**  The project #, name of your system, and the team#

**Test Stage:** Indicate whether it is a unit test or a system test.

**Test Date:**  The date the test was performed.

**Test Case ID#:**  A unique ID is required. Decide on a naming convention and use numbering. Example: Ballot\_Shuffle\_1

**Name(s) of Testers:** List the names of anyone involved in running this test case.

**Test Description:**  Describe briefly the test objective.

**Automated:**  Indicate if the test is completely automated or being checked manually. (If you have methods running the tests and checking results, select “yes”. If you are manually checking results, indicate manual by selecting the “no.”)

**Results:** Indicate if the test passed or failed.

**Step #:** You will be listing the test steps in order. This number is the step number in the process.

**Test Step Description:** Details of the test step.

**Test Data:** What the test data will be for this step. Be clear on what the input data will be. If using a specific file, be clear on the name.

**Expected Result:** What result are you expecting from the program component or system.

**Actual Result:** What result were returned based on the test.

**Postcondition for Test:** What will be true after the test has been run? Has the state of the system changed in any way?

**Notes:** Comments and notesfor you and your team members.