**Additional Features Testing**

**A.**

1. Randomizer generator

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Shuffle Functionality (2)  Sprint Description: SHuffle functionality for STV voting system to randomize the order that ballots come in  Testing Number: Shuffle 1 |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | All inputs will be a string double pointer to a 2D array (all values in array don't matter - only they must all be different so rows can be identified by their location before and after the test is completed) -- this function is done when error handling is already done so only a 2D array can be entered   1. 2D array with 2 rows and 10 columns 2. 2D array with 1 row and 1 columns 3. 2D array with 10 rows and 10 columns 4. 2D array with 100 rows and 1 column 5. 2D array with 100 rows and 15 columns |
| Tests: | 1. Test that the input array has the same rows for an output but the order of the rows is randomized 2. This test is done with all inputs - input columns are noted and then compared to the output 2D array |
| Outputs: | 1. Results in a 2D array with row values being the same - columns flip 50% of the time 2. Results in the same return of a 2D array with one column and row 3. Row values remain the same - order of rows in randomized order 4. Value of all numbers in array change positions randomly 5. Row values remain the same - order of rows in randomized order |
| Passed or Failed: | Passed |
| Date: | 4-20 |

2. Ballot Validity Check

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI - Ballot Validity Check  Sprint Task - Ballot Validity Testing  Testing Number - BV1 |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | 1. .csv files of ballots input into ballot box class for testing    1. Input will be many different types of files with different rows removed for each test    2. Full rows will be removed along with just half the ballots |
| Tests: | 1. Validity test must flag if a row is not valid    1. This includes if no votes are counted    2. This includes where only half the number of candidates are rank |
| Outputs: | 1. Flags the row that has an invalid ballot based off the .csv file |
| Passed or Failed: | Passed |
| Date: | 4-24 |

3. Election Validity

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI - Election Validity Check  Sprint Task - Election Validity Testing  Testing Number - EV1 |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | 1. After Ballot validity test has been run and validate - a 2D array of ballots along with a variable of the number of failures is an input 2. Repeat for many different .csv test files located in /src folder    1. No valid votes    2. Half valid votes    3. Less than half valid votes    4. No invalid votes |
| Tests: | 1. A 2D array and the total number of failures flagged will be passed through the function - either results in an election being run or not will be shown. |
| Outputs: | 1. If more than half the reports are invalid    1. Does not run election and tells user in Audit that not enough ballots have been created 2. If more than half the reports are valid    1. Run election       1. Show the 2D array          1. If file is blank -- will run file as blank file -- bug check when returning to main |
| Passed or Failed: | Passed |
| Date: | 4-25 |

**Voting Function Testing**

**B. These tests contain both Unit testing like above but also System testing of specific voting methods**

1. Ballot Box Votes Added Correctly

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Voting Functions (1)  VF1 -- BallotBox function fixed |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | 1. AddVotes function tested with .csv files - different sizes 2. Run AddVotes with filenames that aren't file addresses 3. Run AddVotes with .csv files that are blank |
| Tests: | Run many different .csv tests to ensure that the ballot box 2D array returned was correctly initialized. |
| Outputs: | 1. 2D array with all initialized ballots (does not use invalidated ballots - must perform tests BV1 and EV1 first) 2. Returns “Error - .csv file not found” and election isn't run 3. Returns blank 2D array |
| Passed or Failed: | Passed - BUG = Need to flag blank .csv file better so that it returns NULL |
| Date: | 4-26 |

1. Proper Test Selected based on number input

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Voting Functions (1)  VF1 -- Proper Selection for RunElection Function |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | 1. Input = 1 2. Input = 2 |
| Tests: | Input is changed in UI so only possible values are 1 and 2. Using Audit section -- if a STV election is chosen, it will display that STV election begins and same for Plurality voting |
| Outputs: | 1. “STV Election selected“ displayed in Audit String 2. “Pl;urality Election selected” displayed in Audit String |
| Passed or Failed: | Passed |
| Date: | 4-26 |

1. STV Voting System runs correctly

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Voting Functions (1)  VF1 -- STV voting fixed |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | .csv files with predetermined winner selected -- audit will note whenever a winner or loser has been found and will show vote allocation after this has happened. Vary the amount of winners to confirm counting and ranking is done correctly.   1. .CSV file with 4 candidates and 100 votes    1. Run with a multitude of different vote counts to ensure ties are counted correctly and ranking is done correctly 2. .CSV with 1 candidate and 100 votes 3. .CSV with 15 candidates and 1 vote |
| Tests: | This is a system test looking at the RunElection function. All inputs for this function are from the GUI and outputs will be in the form of string -- Audit and Results. Checking Audit and Results to confirm proper counting of an election. |
| Outputs: | 1. Will give three different sections within the terminal    1. Audit       1. Talking about how an election was carried out. How winners and losers were handled and when votes were reallocated    2. Results       1. Displays the winners and how many votes they had    3. Invalid Ballots       1. Displays the total number of ballots that are incorrect and what those ballots are and what row the ballot is in    4. Invalid Election       1. If the .csv file is incorrect, will tell the user this and not run the election       2. If too many invalid ballots are in an election (50%), the election will not be run |
| Passed or Failed: | Passed |
| Date: | 4-27 |

1. Coin Toss picks 50%

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Voting Functions (1)  VF1 -- STV voting fixed -- Plurality voting fixed |
| Team Member(s) Responsible: | Adam W. |
| Inputs: | Input two integers into CoinToss Equation |
| Tests: | Testing whether the result of the coin toss function results in random 50/50 chance for each integer |
| Outputs: | 50% the first integer input is output from function  50% the second integer input is output from function |
| Passed or Failed: | Passed - BUG: Randomness might not be totally random. Will result in a random value but not completely random |
| Date: | 4-27 |

1. Plurality Runs correctly

|  |  |
| --- | --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: | PBI: Voting Functions (1)  VF1 -- STV voting fixed -- Plurality voting fixed |
| Team Member(s) Responsible: Adam W. | Adam W. |
| Inputs: | .csv files with predetermined winner selected -- audit will note the rank of the voting and how many votes there are. Vary the amount of winners to confirm counting and ranking is done correctly.   1. .CSV file with 4 candidates and 100-1000 votes    1. Run with a multitude of different vote counts to ensure ties are counted correctly and ranking is done correctly   2. CSV with 1 candidate and 100 votes  3. .CSV with 15 candidates and 1 vote |
| Tests: | This is a system test. Looks at the RunElection function to ensure that plurality counting is done correctly and candidates are ranked based on the count. Audit and Results string used as output for testing but will be implemented with GUI at next stage. |
| Outputs: | 1. Will give three different sections within the terminal    1. Audit       1. Talking about how an election was carried out. How winners and losers were handled and when votes were reallocated    2. Results       1. Displays the winners and how many votes they had    3. Invalid Ballots       1. Displays the total number of ballots that are incorrect and what those ballots are and what row the ballot is in    4. Invalid Election       1. If the .csv file is incorrect, will tell the user this and not run the election       2. If too many invalid ballots are in an election (50%), the election will not be run |
| Passed or Failed: | Passed |
| Date: | 4-27 |