SPRINT BACKLOG					
Parent PBI ID	**Parent PBI Description**	**Task ID**	**Task Description**	**Owner(s)**	**Task Status**
1	Run an algorithm to determine the results of a Plurality type election	1	Refactor code for plurality elections where possible. Make sure code is efficient and well commented.		Not Done
1	Run an algorithm to determine the results of a Plurality type election	2	Test: The program runs without crashing and computes correct results for an election with 0 ballots.		Not Done
1	Run an algorithm to determine the results of a Plurality type election	3	Test: The program runs without crashing and computes correct results for an election with 100,000 ballots.		Not Done
1	Run an algorithm to determine the results of a Plurality type election	4	Test: The program runs without crashing and computes correct results for an election with 1 candidate		Not Done
1	Run an algorithm to determine the results of a Plurality type election	5	Test: The program runs without crashing and computes correct results for an election with 10 candidates		Not Done
1	Run an algorithm to determine the results of a Plurality type election	6	Test: The program runs without crashing and computes correct results for an election that contains only ballots that do not rank candidates		Not Done
1	Run an algorithm to determine the results of a Plurality type election	7	Test: The program runs without crashing and computes correct results for an election with 10 candidates		Not Done
1	Run an algorithm to determine the results of a Plurality type election	8	Test: The program runs without crashing and computes correct results for an election with a two-way tie		Not Done
1	Run an algorithm to determine the results of a Plurality type election	9	Test: The program runs without crashing and computes correct results for an election with a four-way tie		Not Done
2	Run an algorithm to determine the results of a Plurality type election	10	Refactor code for plurality elections where possible. Make sure code is efficient and well commented.		Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	11	Test: The program runs without crashing and computes correct results for an election with 0 ballots.	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	12	Test: The program runs without crashing and computes correct results for an election with 100,000 ballots.	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	13	Test: The program runs without crashing and computes correct results for an election with 1 candidate	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	14	Test: The program runs without crashing and computes correct results for an election with 10 candidates	Taylor	Not Done

2	Run an algorithm to determine the results of a STV with Droop Quota type election	15	Test: The program runs without crashing and computes correct results for an election that contains only ballots that do not rank candidates	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	16	Test: The program runs without crashing and computes correct results for an election with 10 candidates	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	17	Test: Write unit tests to make sure the candidate class works correctly.	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	18	Add common dependencies for tests.	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	19	Change existing tests to use the common dependencies.	Taylor	Not Done
2	Run an algorithm to determine the results of a STV with Droop Quota type election	20	Create insructions to compile and run the program and tests.	Taylor	Not Done
3	Have the ballot files loaded shuffled	21	Test: Check if the first ballot id does not match with the given while shuffling.	Arun	Done
4	Have the program check if any of the ballots added are invalid	22	Planning and Design: Look up and determine the best approach to preprocess the invalidated ballots and discard them	Amir	In Progress
4	Have the program check if any of the ballots added are invalid	23	Write code to identify invalid ballots for elections and discard them before running the election.	Amir	In Progress
4	Have the program check if any of the ballots added are invalid	24	Document the code to identify invalid ballots. Also change documentation for existing code that was changed (if any) by this.	Amir	In Progress
4	Have the program check if any of the ballots added are invalid	25	Test: Check if the program correctly identifies the invalidated ballots passed in and discards them before running the election.	Arun	In Progress
5	Have the program report the invalidated ballots in a Droop election by saving them to disk in a file.	26	Write code to save the invalidated ballot's information to a file, then to save that file to disk.	Arun	Not Done
5	Have the program report the invalidated ballots in a Droop election by saving them to disk in a file.	27	Document the code to save invalid ballots.	Arun	Not Done
5	Have the program report the invalidated ballots in a Droop election by saving them to disk in a file.	28	Test: Check if the invalidated report file is created and written in the disk with the expected contents.	Arun	Not Done
6	Select multiple ballot files with a GUI.	29	Write code to provide the user with a file chooser that allows them to select only ballot files. The user should be able to select multiple files.		Done
6	Select multiple ballot files with a GUI.	30	Document the code for the file chooser for multiple ballot files.		Done
6	Select multiple ballot files with a GUI.	31	Test: Verify that the system correctly gathers all ballots and candiates for multiple ballot files with 5 candidates and 30 ballots.	Nicholas	In Progress

7	Search for ballot files on the disk with a GUI.	32	Ensure that the file chooser provides search functionality for the user.	Nicholas	Done
7	Search for ballot files on the disk with a GUI.	33	Test: Verify that the explorer opens and can find ballot files successfully.	Nicholas	In Progress
8	Select a directory to load ballot files from using a GUI.	34	Write code to provide the user with a file chooser that allows them to select only directories. The user should be able to select only one directory.		Done
8	Select a directory to load ballot files from using a GUI.	35	Document the code for the file chooser for a single directory.		Done
8	Select a directory to load ballot files from using a GUI.	36	Test: Verify that the system correctly gathers all ballots and candiates for single/multiple ballot file(s) with 5 candidates and 30 ballots. Subdirectories should not be included, only the directory selected.	Nicholas	In Progress
8	Select a directory to load ballot files from using a GUI.	37	Test: Verify that the system correctly gathers all ballots and candiates for single/multiple ballot file(s) with 10 candidates and 50 ballots. Subdirectories should not be included, only the directory selected.	Nicholas	In Progress
9	Create a short report of election results and save it to disk as a file.	38	Create a method to interpret the results of an election and produce the info needed for the short report as a String.	Nicholas	Not Done
9	Create a short report of election results and save it to disk as a file.	39	Write code to save this String to a file as the short report.	Nicholas	Not Done
9	Create a short report of election results and save it to disk as a file.	40	Test: Verify that the report generated contains the correct election results	Nicholas	Not Done
9	Create a short report of election results and save it to disk as a file.	41	Test: Verify that the file is saved in the location specified by the user	Nicholas	Not Done
9	Create a short report of election results and save it to disk as a file.	42	Document the new code for creating the short report.	Nicholas	Not Done