

# Sprint Backlogs Updated

## **1:**

Committed backlog item:

*As the product owner, I want both Plurality and Droop algorithms to be fully implemented and working correctly so that I can run the results of the election through either algorithm.*

### **Acceptance Criteria:**

- Prompt user input: 1 for Plurality, or 2 for Droop
- Validate that user input is an integer, and is either 1 or 2. Any other input will exit the program.
- Inform user of choice and generate either a new Plurality or STV class based on their choice
- Generate success/failure message if the classes cannot be generated/initialized
- Generate error message if error occurs during execution
- Generate error message if the report can not be created or exported.
- Both algorithms fully implemented and working properly

**Effort:** Medium

Completed:

- Fix the issue that when candidate's vote counts are same, plurality algorithm's way of choosing winner is actually not random. Time estimated: 2hr.
  - Test for all functionalities of both algorithms and user's ability to choose algorithms. Time estimated: 1 hr
  - Refactor the program, cancel unneeded functions and information. Time estimated: 1.5 hr
- 

## **2:**

Committed backlog item:

*As a developer, I want to be able to toggle the shuffling of ballots to be on or off so that I can test the functionality of the system and so that users aren't prompted for shuffling.*

**Acceptance Criteria:**

- Ballots are randomly shuffled by default
- The developer can turn on/off the shuffle option through an argument in command line
- The election officials don't need to decide whether the ballots need to be shuffled

**Effort:** Small

Completed:

- Make shuffle the default option for d algorithms. Time estimated: 0.5hr
  - Add one optional argument when user run the program through command line. Time estimated: 0.5hr
  - Add a flag in the program to decide if the shuffle option should be turned off based on if that additional argument got passed in. Time estimated: 1 hr
  - Test if the shuffle really be turned off. Time estimated: 1hr
- 

**3:**

Committed backlog item:

*As a user, I want all election information (seats, ballots, candidates, algorithm) stored in the CSV file so that I don't have to input it manually.*

**Acceptance Criteria:**

- The only (potential, see other PBI on file reading) input should be the file name
- Program is able to read election criteria from CSV file
- Program ensures that all information exists and is valid
  - If not, program informs user and prompts them for missing input
- Program store all input into classes and variables successfully

**Effort:** Small/Medium

Completed:

- Change the way the current csv file is formatted. Time estimated: 1 hr
- Change the way the program read in and parse the contents in the csv file. Time estimated: 2 hr
- Add error checkings for the contents read in from the csv file. Time estimated: 1 hr
- Test that this change won't affect the performance of the algorithms. Time estimated: 1 hr

**4:**

Committed backlog item:

*As a user, I want to either supply a filename as a command line argument or be prompted for the filename if nothing is supplied so that I have two ways to input the file.*

**Acceptance Criteria:**

- User can run the program with an additional argument which is the name of the CSV file
- Validate if the file is existed and opened successfully
- Prompt the user to input filename if no argument supplied, or filename supplied is invalid
- Generate success message when the file is opened successfully
- Read file contents successfully, or generate error message if an error occurs

**Effort:** Small

**Completed:**

- Add one optional argument when user run the program through command line. Time estimated: 0.5hr
  - Change the program structure to enable both way of passing in file name. Time estimated: 1hr
  - Test if both way of passing in file name works correctly. Time estimated: 1hr
- 

**5:**

Committed backlog item:

*As an election official, I want ballots to be discarded if less than half of the candidates are ranked when using droop so that we can ensure the validity of the ballots.*

**Acceptance Criteria:**

- Only run these checks when using Droop Quota, and not Plurality
- Check how many candidates are ranked for each ballot when processing the ballot information
- Discard the ballot if the number of ranking is less than half of the number of candidates

**Effort:** Medium

Completed:

- Check if the ballot has over half of the candidates ranked when read in the ballot, and discard those invalidated ballots, and generate a new ballot array that is used to process the election. Time estimated: 2 hr
  - Add print statements to inform user which ballot or ballots are discarded. Time estimated: 0.5hr
  - Test functionality after discarding invalidated ballots. Time estimated: 2 hr
- 

**6:**

Committed backlog item:

As an election official, I want to have an additional short report containing selection date, type of election, candidates, number of seats, the winners of the election, so that I can give it to the election certification officials.

**Acceptance Criteria:**

- The program created a text file and output the needed information to it
- The file name should be different from the name of the longer report
- Generate failure message if the file can not be created or opened
- Generate error message when error occurs during creating the report

**Effort:** Small/Medium

In process:

- Implement an additional function to generate this short report, it should be within the Algorithm class. Time estimated: 2 hr
- Add functions to get the information this report needed. Time estimated: 1 hr
- Test if the report can be successfully exported and the contents are correct. Time estimated: 1hr