Software Requirements Specification

for

Voting System

Version 1.0

Prepared by Bethany Freeman, Derrick Dischinger, and Rock Zgutowicz

University of Minnesota - Twin Cities

February 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Bethany Freeman	2/6/2024	Updated information pertaining to section 2.3	0.4
Rock Zgutowicz	2/7/2024	Updated information pertaining to to section 1, 2.1, 2.3, 2.4, 2.7	0.4
Rock Zgutowicz	2/9/2024	Updated information pertaining to to section 2.2, 2.5	0.4
Bethany Freeman	2/9/2024	Updated information pertaining to section 2.1, 2.2, 2.4, Appendix A, Section 4, 5.2	0.5
Derrick Dischinger	2/10/2024	Updated information pertaining to Section 1.5, 2.6, 3.2. 3.3, Appendix A, Section 5	0.6
Rock Zgutowicz	2/10/2024	Updated information pertaining to section 5.2	0.6
Bethany Freeman	2/10/2024	Updated information pertaining to section 1.1, 2.1, Appendix A, Section 4	0.7
Derrick Dischinger	2/11/2024	Updated information pertaining to Section 5.4, 3.1, 3.4	0.8
Rock Zgutowicz	2/11/2024	Updated information pertaining to section 2.4, 5.1, 6	0.9
Bethany Freeman	2/12/2024	Updated information pertaining to Appendix B, Updated formating	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of Voting System version 1.0 to determine the results of closed party listing and open party listing elections. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software, testers, and potential developers.

1.1 Document Conventions

This document was created based on the IEEE template for System Requirement Specification Documents.

1.2 Intended Audience and Reading Suggestions

- Typical Users, such as election officials who wish to use this Voting System for generating audit files from election ballots (Open Party List, Closed Party List).
- Programmers who are interested in working on the project by further developing it or fixing existing bugs.
- Testers who wish to ensure proper functionality of the voting system to verify fairness of the results.

1.3 Product Scope

Voting System is a tool that election officials can use to generate audit files containing the results of open party list and closed party list elections efficiently and with accuracy. With this, the generated audit files will contain the result from either an open party listing or closed party listing election. It can also be used to share election results with others, such as media personnel.

1.4 References

Voting System's GitHub page:

https://github.umn.edu/umn-csci-5801-01-S24/repo-Team2

IEEE Template for System Requirement Specification Documents:

https://canvas.umn.edu/courses/413086/files/41290146?module_item_id=11735324

2. Overall Description

2.1 Product Perspective

The Voting System was developed as new software for election officials to efficiently and accurately produce audit files containing information regarding the results of open party list and closed party list elections. The software only supports csv files and does not support write-in ballots.

2.2 Product Functions

File:

- Verify file name through interaction with the command line: Tester runs program through the command line and gives a file to run through the program, which will determine whether or not the file has been found.
- Verify file name through interaction with a text based interface: Election officials will run a program through an interface and give a file to run through the program, which will generate election results.
- Extract information from verified file: After files have been verified, the system will parse through the given file and extract the information into relevant data structures.

Closed Party Listing Election Results:

- Run CPL algorithm: System will run algorithm for CPL election on parsed information from the given file, will generate election results for a CPL voting system.

Open Party Listing Election Results:

- Run OPL algorithm: System will run algorithm for OPL election on parsed information from the given file, will generate election results for an OPL voting system.

Tie Break:

- Tie breaking mechanism: The tie breaking mechanism is used to fairly obtain a result of who has won a tie. This will be usable for both parties and candidates.

Audit File Creation:

- Creating of the audit file of a CPL election: Creates an audit file. This will be a text file, containing the formatted results from the CPL election.
- Creating of the audit file of an OPL election: Creates an audit file. This will be a text file, containing the formatted results from the OPL election.

Display Results:

- Display of results for the tester in an OPL election: Allows the user to instantly see the results of the OPL election through the command line; can refer to the audit file for more information.
- Display of results for the tester in a CPL election: Allows the tester to instantly see the results of the CPL election through the command line; can refer to the audit file for more information.
- Display of results for the election official in an OPL election: Allows the user to instantly see the results of the OPL election through a text based interface; can refer to the audit file for more information.
- Display of results for the election official in a CPL election: Allows the user to instantly see the results of the CPL election through a text based interface; can refer to the audit file for more information.

2.3 User Classes and Characteristics

- Testers: Those interested in testing the various methods and classes of the project before completion.
- Election Officials: Those who want to use the program to generate election results for an open party list or closed party list election.

- Programmers: Those who are interested in working on the project by further developing it or fixing existing bugs.

2.4 Operating Environment

- Ubuntu 22.04.3 LTS

2.5 Design and Implementation Constraints

The Voting System will be developed in Java. The Voting System will only receive one file. The Voting System will support both closed party listing and open party listing elections. The file structure cannot be changed outside of the program. The program must be able to process 100,000 ballots in under 4 minutes. The election file will be in the same directory as the program. The Voting System must be able to run on CSE lab machines.

2.6 User Documentation

A README.md file will be provided along with the program to give instructions as well as to give an overview of the program. All questions regarding usage will be answered in the README.

2.7 Assumptions and Dependencies

Voting System was developed in Java and therefore requires Java to be installed on the user's system. The latest stable version of Voting System requires Java version 19 or higher. This applies to Windows and Linux users. On Mac OS X, Java is bundled with the application.

3. External Interface Requirements

3.1 User Interfaces

The main user Interface for the program will be a text based interface, the system will prompt the user to enter a filename and press the enter key. Once the system has finished processing, the file will display results to the user through the same interface. These results will differ in appearance depending on the type of election the user has run. See Ex. 2 and Ex. 5 for reference in Appendix B.

3.2 Hardware Interfaces

The Voting System has been developed to be able to run on an up to date Ubuntu system.

3.3 Software Interfaces

Java must be installed on the system, see 2.7 for more information. The command line must also be accessible.

3.4 Communications Interfaces

There will be no communication between our system and outside systems. The only information transferred is through the CSV file input and the users receiving the output.

4. System Features

4.1 Read in CSV File through the Command Line

Name	**Verify File Name Through Interaction with Command Line**
ID	VF_001
Dependencies	None
Description	Tester runs the program through the command line and gives a file to run through the program, which will determine whether or not the file has been found.
Actors	Tester
Organizational Benefits	Allow for testing through the command line
Frequency of Use	Testers will use this functionality during the testing phase
Triggers	Tester running program through command line
Precondition(s)	Program is runnable through command line

Postcondition(s)	Verification of whether file name passed in is a valid file
Main	

Step	**Action**
1.	Tester starts voting program through the command line
2.	Program prompts tester for a filename to run the program on
3.	Program verifies that the file exists, gives a message to the command line stating that the file has been found and will next run the extraction of information.

Exceptions	

Step	**Action**
3.	Program not able to find the file, runs into *fileNotFoundException*, will give a message out to the command line letting tester know, prompt for file again

Comments	N/A

4.2 Read in CSV File through the Interface

Name	**Verify File Name Through Interaction with an Interface**
ID	VF_002
Dependencies	None
Description	Election Official runs program through an interface and gives a file to run through the program, which will generate election results
Actors	Election Official
Organizational Benefits	Allows for easy usage by the Election Official
Frequency of Use	Whenever an election official needs voting results for a CPL or OPL election
Triggers	Election official run the program through an executable
Precondition(s)	Election official is running a file from a ballot program to get voting results
Postcondition(s)	Verification of whether file name passed in is a valid file

Main	<table><tr><th>Step</th><th>Action</th></tr><tr><td>1.</td><td>Election Official starts program through an executable</td></tr><tr><td>2.</td><td>Interface will prompt the Election Official for a file to run, and give a place for the Election Official to input that file name</td></tr><tr><td>3.</td><td>Program verifies that the file exists and will next run the extraction of information.</td></tr></table>
Exceptions	<table><tr><th>Step</th><th>Action</th></tr><tr><td>3.</td><td>Program not able to find the file, runs into *fileNotFoundException*, will give a message out to the interface letting the Election Official know, will prompt for a filename again</td></tr></table>
Comments	N/A

4.3 Extract Information from a Valid CSV File

Name	**Extract Information From Verified File**
ID	EI_001
Dependencies	● VF_001: *Verify File Name through interaction with command line* ● VF_002: *Verify file name through interaction with an interface*
Description	After files have been verified, the system will parse through the given file and extract the information into relevant data structures
Actors	System
Organizational Benefits	Parses all information before handing that information off to either the CPL election or the OPL election
Frequency of Use	Whenever the program is run and the file is successfully verified
Triggers	File successfully verified
Precondition(s)	A file was given that exists
Postcondition(s)	All data from given file are successfully extracted and placed into relevant

	data structures
Main	**Step** / **Action** 1. System successfully verified existence of file 2. System will parse given file checking, placing information into data structures. 3. System will then transfer information into proper voting system for result calculation, this will be either CPL or OPL
Exceptions	**Step** / **Action** 2. System does not find OPL or CPL as the first line within the given file, and will send a message to the user that the file contains invalid information. Prompt for new file. 2. System does not find a digit on the first, second, or third line of the given file. Send a message to the user that the file contains invalid information. Prompt for new file. 2. System does not find enough party members and/or candidates to correspond to the third digit found in the given file. Send a message to the user that the file contains invalid information. Prompt for new file. 2. System does not find enough ballots to correspond to the second digit found in the given file. Send a message to the user that the file contains invalid information. Prompt for new file. 2. System does not reach the end of the given file after parsing through all ballots specified by the second digit found in the given file. Send a message to the user that the file contains invalid information. Prompt for new file.
Comments	N/A

4.4 Compute Election Results in the Case of a CPL Election

Name	**Run CPL Algorithm**

ID	VA_001
Dependencies	● EI_001: *Extract Information from verified file*
Description	System will run algorithm for CPL election on parsed information from the given file, will generate election results for an CPL voting system
Actors	System
Organizational Benefits	Easier to produce accurate election results for a CPL election
Frequency of Use	Whenever the first line in a given file corresponds to "CPL"
Triggers	EI_001 has successfully extracted information from the given file and the first line was "CPL"
Precondition(s)	The file was successfully verified and the information was successfully parsed
Postcondition(s)	Results for a CPL election will have been found and saved to send to produce the Audit File
Main	

Step	**Action**
1.	System parses through ballot data structure to determine the number of votes per party
2.	System will do the first round of allocation based on the largest remainder. That remainder will be subtracted from the parties overall votes, this will continue until no parties have more votes then the remainder.
3.	System will save this information
4.	System will start a second allocation based on the largest amount of remaining votes. The remainder will be subtracted from the parties overall votes. This will continue until no more seats are available.
5.	The system will then save the results to a data structure and send to be produced into an audit file

Alternate Course	

Step	**Action**
1.	System parses through ballot data structure to determine the number of votes per party

	2.	System will do the first round of allocation based on the largest remainder. That remainder will be subtracted from the parties overall votes, this will continue until no parties have more votes then the remainder.
	3.	System will save this information
	4.	System will start a second allocation based on the largest amount of remaining votes. The remainder will be subtracted from the parties overall votes. Should two or more parties have a tie in the amount of votes they have remaining and both are in a position in which they could get a seat, the tie breaking mechanism will be called upon and one party will be returned as the party to get the seat. The remainder will be subtracted from the party that won the seat's overall votes. No changes will be made to the votes of any losing parties. This will continue until no more seats are available.
	5.	The system will then save the results to a data structure and send to be produced into an audit file
Exceptions	N/A	
Comments	N/A	

4.5 Compute Election Results in the Case of an OPL Election

Name	**Run OPL Algorithm**
ID	VA_002
Dependencies	● EI_001: *Extract Information from verified file*
Description	System will run algorithm for OPL election on parsed information from the given file, will generation election results for an OPL voting system
Actors	System
Organizational Benefits	Easier to produce accurate election results for a OPL election
Frequency of Use	Whenever the first line in a given file corresponds to "OPL"
Triggers	EI_001 has successfully extracted information from the given file and the first line was "OPL"

Precondition(s)	The file was successfully verified and the information was successfully parsed
Postcondition(s)	Results for a OPL election will have been found and saved to send to produce the Audit File

Main

Step	**Action**
1.	System parses through ballot data structure to determine the number of votes per party and per candidate
2.	System will do the first round of allocation based on the largest remainder. That remainder will be subtracted from the parties overall votes, this will continue until no parties have more votes then the remainder.
3.	System will save this information
4.	System will start a second allocation based on the largest amount of remaining votes. The remainder will be subtracted from the parties overall votes. This will continue until no more seats are available.
5.	System will then rank candidates within each party separately based on the number of ballots within each candidate's name.
6.	The System will then save the results into a data structure and send over to be produced into an audit file

Alternate Course

Step	**Action**
1.	System parses through ballot data structure to determine the number of votes per party and per candidate
2.	System will do the first round of allocation based on the largest remainder. That remainder will be subtracted from the parties overall votes, this will continue until no parties have more votes then the remainder.
3.	System will save this information
4.	System will start a second allocation based on the largest amount of remaining votes. The remainder will be subtracted from the parties overall votes. Should two or more parties have a tie in the amount of votes they have remaining and both are in a position in which they could get a seat, the tie breaking

	mechanism will be called upon and one party will be returned as the party to get the seat. The remainder will be subtracted from the party that won the seat's overall votes. No changes will be made to the votes of any losing parties. This will continue until no more seats are available.
5.	System will then rank candidates within each party separately based on the number of ballots within each candidate's name. Should two or more candidates have a tie in the amount of votes received, the tie breaking mechanism will be called upon and the order of candidates will be returned. No changes to the amount of votes will be made. The final ranking will be based off of the tie break.
6.	The System will then save the results into a data structure and send over to be produced into an audit file

Exceptions	N/A
Comments	N/A

4.6 Compute the Winner of a Tie

Name	**Tie Breaking Mechanism**
ID	TB_001
Dependencies	None
Description	The Tie Breaking mechanism is used to fairly obtain a result of who has won a tie. Is usable for both parties and candidates.
Actors	System
Organizational Benefits	Allows for fair breaking of ties during both OPL and CPL elections
Frequency of Use	Whenever a tie is found during either a CPL or OPL election
Triggers	Two or more parties have the same amount of votes remaining at any point during the second allocation within both OPL and CPL. Two or more candidates have the same amount of votes during ranking of candidates within an OPL election.
Precondition(s)	A tie has been found between two or more parties or candidates during either an OPL or CPL election.

Postcondition(s)	Rankings in order of first to last of tied parties or candidates
Main	

Step	**Action**
1.	A tie has been found within either an OPL or CPL election, all parties that are currently tied at the same amount of votes will be given to the Tie Breaking Mechanism.
2.	A random number, between 1 and 10, will be generated 1000 times, the 1001th time will be the number chosen to be compared against.
3.	Each candidate or party will pull a random number, between 1 and 10, this will be generated 1000 times, the 1001th time will be given to the candidate or party currently generating.
4.	Based on the numbers randomly generated, the candidate or party closest to the random number generated for the comparison will be the winner, if more than two are being compared, the second closet will get second, and so on
5.	Return the listing of the candidates in order of first to last back to the proper election.

Exceptions	

Step	**Action**
4.	If two or more candidates or parties are the same amount away from the random comparison number, the tie breaking mechanism will be run again on those candidates or parties specifically.

Comments	N/A

4.7 Create an Audit File for a CPL Election

Name	**Creation of the Audit File of a CPL Election**
ID	AF_001
Dependencies	● VA_001: *Run CPL algorithm*
Description	Creates an audit file, this will be a text file, containing the formatted results

	from the CPL election
Actors	System
Organizational Benefits	Allows for ease of sharing election results with others and an easy to read output for Election Officials
Frequency of Use	Whenever an election finishes results
Triggers	Whenever an election has ended and results have been obtained
Precondition(s)	A CPL election has run to completion successfully
Postcondition(s)	A file with the formatted election results
Main	

Step	**Action**
1.	Results from a CPL election have been obtained
2.	The system creates a new file with the naming convention CPL_Election_Results_systemTime.txt
3.	Information will be written to the file in this order - CPL Election - The number of parties within the election - The number of ballots that were cast - The number of seats that were available - A formatted table of the Parties and what candidates were in that party, this will be in the same order as what was originally entered - A formatted table containing, in this order, A list of parties up for election, the number of votes each party got, what the seats per party were after the first allocation of seats, how many votes were remaining per party before the second allocation of seats, what the seats per party were from the second allocation of seats, the final seat totals per party, and lastly the percent of votes each party got compared to the percent of seats the party got - Lastly, a formatted table showing the winning parties and an in order list of their seat winners
4.	File will be saved

Exceptions	

Step	**Action**

	2. The system attempts to create a new file but the file could not be created, the system will throw *java.io.IOException*. Will attempt to create another new file with a different name.
Comments	systemTime within the file naming convention refers to the fact that the time of system will be added at the end of the filename

4.8 Create an Audit File for an OPL Election

Name	**Creation of the Audit File of a OPL Election**
ID	AF_001
Dependencies	● VA_002: *Run OPL algorithm*
Description	Creates an audit file, this will be a text file, containing the formatted results from the election
Actors	System
Organizational Benefits	Allows for ease of sharing election results with others and an easy to read output for Election Officials
Frequency of Use	Whenever an election finishes results
Triggers	Whenever an election has ended and results have been obtained
Precondition(s)	An OPL election has run to completion successfully
Postcondition(s)	A file with the formatted election results
Main	

Step	**Action**
1.	Results from a OPL election have been obtained
2.	The system creates a new file with the naming convention OPL_Election_Results_systemTime.txt
3.	Information will be written to the file in this order - OPL Election - The number of parties within the election - The number of ballots that were cast - The number of seats that were available - A formatted table of the Parties and what candidates were in that party, this will be in the order by vote that

	was determined. - A formatted table containing, in this order, A list of parties up for election, the number of votes each party got, what the seats per party were after the first allocation of seats, how many votes were remaining per party before the second allocation of seats, what the seats per party were from the second allocation of seats, the final seat totals per party, and lastly the percent of votes each party got compared to the percent of seats the party got - Lastly, a formatted table showing the winning parties and an in order list of their seat winners, each seat winner will have the number of votes displayed as well
4.	File will be saved

Exceptions	

Step	**Action**
2.	The system attempts to create a new file but the file could not be created, the system will throw *java.io.IOException*. Will attempt to create another new file with a different name.

Comments	systemTime within the file naming convention refers to the fact that the time of system will be added at the end of the filename

4.9 Display the Results to the Command Line for an OPL Election

Name	**Display of Results for the Tester in an OPL Election**
ID	DR_001
Dependencies	● VA_002: *Run OPL algorithm*
Description	A display of the OPL election results to the tester through the command line
Actors	System
Organizational Benefits	Allows the user to instantly see the results of the OPL election; can go to the audit file for more information
Frequency of Use	Whenever an OPL election finishes

Triggers	An election has finished
Precondition(s)	The results have been obtained from a successful OPL election
Postcondition(s)	A formatted display of the results from the election directly to the command line
Main	**Step** / **Action** 1. An election has finished 2. The system will display a formatted table showing the winning parties and an in order list of their seat winners, each seat winner will have the number of votes displayed as well, to the command line 3. The system will output a message to the command line that says what the audit files name is and where it is saved
Exceptions	N/A
Comments	N/A

4.10 Display the Results to the Command Line for a CPL Election

Name	**Display of Results for the Tester in an CPL election**
ID	DR_002
Dependencies	● VA_001: *Run CPL algorithm*
Description	A display of the CPL election results to the tester through the command line
Actors	System
Organizational Benefits	Allows the user to instantly see the results of the CPL election, can go to the audit file for more information
Frequency of Use	Whenever an CPL election finishes
Triggers	An election has finished
Precondition(s)	The results have been obtained from a successful CPL election
Postcondition(s)	A formatted display of the results from the election directly to the

	command line
Main	**Step** / **Action** 1. An election has finished 2. The system will display a formatted table showing the winning parties and an in order list of their seat winners, to the command line 3. The system will output a message to the command line that says what the audit files name is and where it is saved
Exceptions	N/A
Comments	N/A

4.11 Display the Results to the Interface for an OPL Election

Name	**Display of Results for the Election Official in an OPL election**
ID	DR_003
Dependencies	● VA_002: *Run OPL algorithm*
Description	A display of the OPL election results to the user through a text based interface
Actors	System
Organizational Benefits	Allows the user to instantly see the results through a text based interface, can go to the audit file for more information
Frequency of Use	Whenever an OPL election finishes
Triggers	An election has finished
Precondition(s)	The results have been obtained from a successful OPL election
Postcondition(s)	A formatted display of the results from the election directly to the interface
Main	**Step** / **Action**

1.	An election has finished
2.	The system will display a formatted table showing the winning parties and an in order list of their seat winners, each seat winner will have the number of votes displayed as well, to the interface
3.	The system will output a message to the interface that says what the audit files name is and where it is saved

Exceptions	N/A
Comments	N/A

4.12 Display the Results to the Interface for a CPL Election

Name	**Display of Results for the Election Official in an CPL election**
ID	DR_004
Dependencies	● VA_002: *Run CPL algorithm*
Description	A display of the CPL election results to the user through a text based interface
Actors	System
Organizational Benefits	Allows the user to instantly see the results through a text based interface, can go to the audit file for more information
Frequency of Use	Whenever an CPL election finishes
Triggers	An election has finished
Precondition(s)	The results have been obtained from a successful CPL election
Postcondition(s)	A formatted display of the results from the election directly to the interface
Main	

Step	**Action**
1.	An election has finished
2.	The system will display a formatted table showing the winning parties and an in order list of their seat winners, to the interface

	3. The system will output a message to the interface that says what the audit files name is and where it is saved
Exceptions	N/A
Comments	N/A

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The product must be able to process every 100,000 ballots in under 4 minutes, as to ensure election results are produced in a timely manner.

5.2 Safety Requirements

Any CSV file given to the program must have no record of changes after the time they have been generated from the ballot. For any one of these files, the file permissions must be read only. To enforce this, all file reading will be done by the system and will take place past user interference.

5.3 Security Requirements

There are no security requirements regarding user authentication or authority. The security requirements regarding the election information itself must all be handled prior to the usage of this product.

5.4 Software Quality Attributes

The software will prioritize maintainability and extensibility through the implementation of modular design patterns. This will make adding new voting methods easy and efficient, promoting adaptability and ease of maintenance.

5.5 Business Rules

Election officials and designated testers have the authority to pass a file into the system and receive the results. Media personnel are only allowed to view election results after they have been passed through the system and placed inside the audit file.

6. Other Requirements

Appendix A: Glossary

Closed Party Election (CLP): Closed Party Listing is a method of voting where voters vote for a party rather than individual people. Each party gives a list of candidates in a given order. Seats are allocated based on the percentages of the vote won, candidates highest on the list receive the seats allocated to their party..

Open Party Election (OLP): Open Party Listing is a method of voting that has voters vote for a specific candidate rather than a party itself. Ballots consist of unordered lists of candidates for each respective party. Votes towards a candidate also count as a vote towards the candidate's respective party. Seats are then assigned based on the popularity of parties and the popularity of the candidates within those parties.

Comma-Separated Values (CSV) file: a CSV file is a text file format that uses commas to separate values, typically seen as an extension to an Excel document

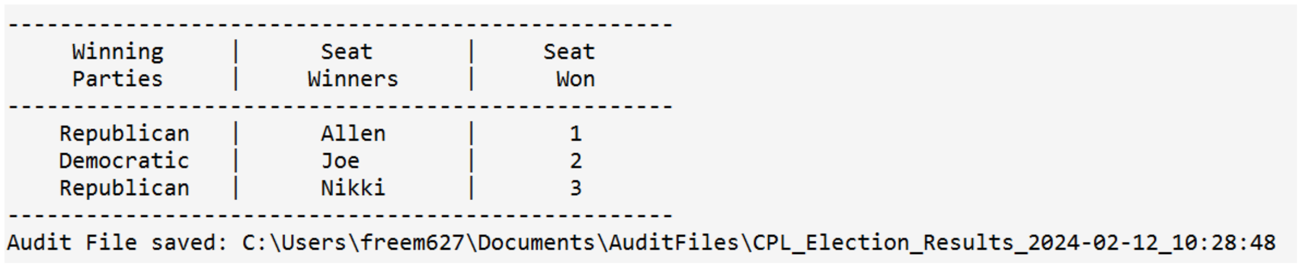
Appendix B: Analysis Models

Closed Party Election (CLP) Documentation Examples:

Example Input of Ballots for CLP Election (Ex.1):

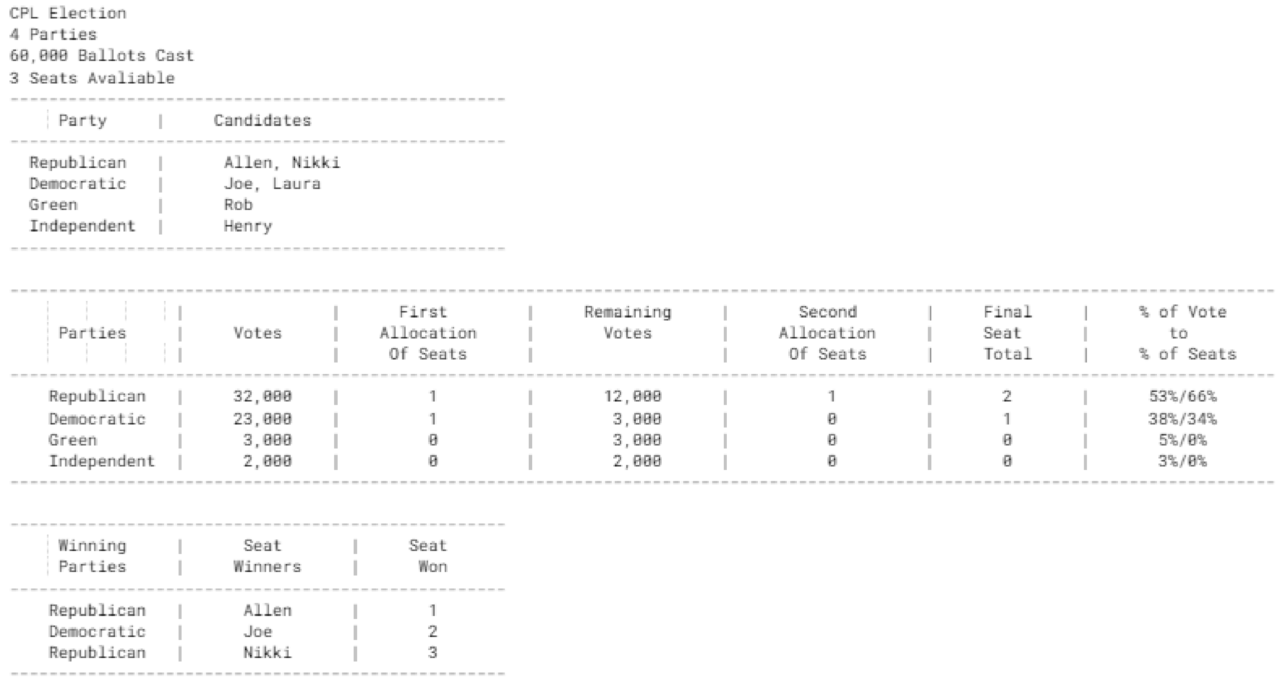
```
CPL
3
9
6
Democratic, Joe, Sally, Ahmed
Republican, Allen, Nikki, Taihui
New Wave, Sarah
Reform, Xinyue, Nikita
Green, Bethany
Independent, Mike
1,,,,,
1,,,,,
,1,,,,
,,,,1,
,,,,,1
,,,1,,
,,,1,,
1,,,,,
,1,,,,
```

Example Output to User for CLP Election (Ex. 2):



```
---------------------------------------------------
     Winning      |       Seat        |      Seat
     Parties      |      Winners      |       Won
---------------------------------------------------
    Republican    |       Allen       |        1
    Democratic    |       Joe         |        2
    Republican    |       Nikki       |        3
---------------------------------------------------
Audit File saved: C:\Users\freem627\Documents\AuditFiles\CPL_Election_Results_2024-02-12_10:28:48
```

Example Output to Audit File for CLP Election:



```
CPL Election
4 Parties
60,000 Ballots Cast
3 Seats Avaliable
---------------------------------------------------
    | Party      |     Candidates
---------------------------------------------------
  Republican  |      Allen, Nikki
  Democratic  |      Joe, Laura
  Green       |      Rob
  Independent |      Henry
---------------------------------------------------

-----------------------------------------------------------------------------------------------------------------------------------
    |            |                |      First       |    Remaining     |     Second      |    Final    |    % of Vote
    | Parties    |      Votes     |    Allocation    |      Votes       |   Allocation    |    Seat     |       to
    |            |                |     Of Seats     |                  |    Of Seats     |    Total    |    % of Seats
-----------------------------------------------------------------------------------------------------------------------------------
    Republican  |     32,000     |        1         |      12,000      |        1        |      2      |      53%/66%
    Democratic  |     23,000     |        1         |       3,000      |        0        |      1      |      38%/34%
    Green       |      3,000     |        0         |       3,000      |        0        |      0      |       5%/0%
    Independent |      2,000     |        0         |       2,000      |        0        |      0      |       3%/0%
-----------------------------------------------------------------------------------------------------------------------------------

---------------------------------------------------
    | Winning    |      Seat      |      Seat
    | Parties    |     Winners    |       Won
---------------------------------------------------
    Republican  |      Allen     |        1
    Democratic  |      Joe       |        2
    Republican  |      Nikki     |        3
---------------------------------------------------
```

Open Party Election (OLP) Documentation Examples:

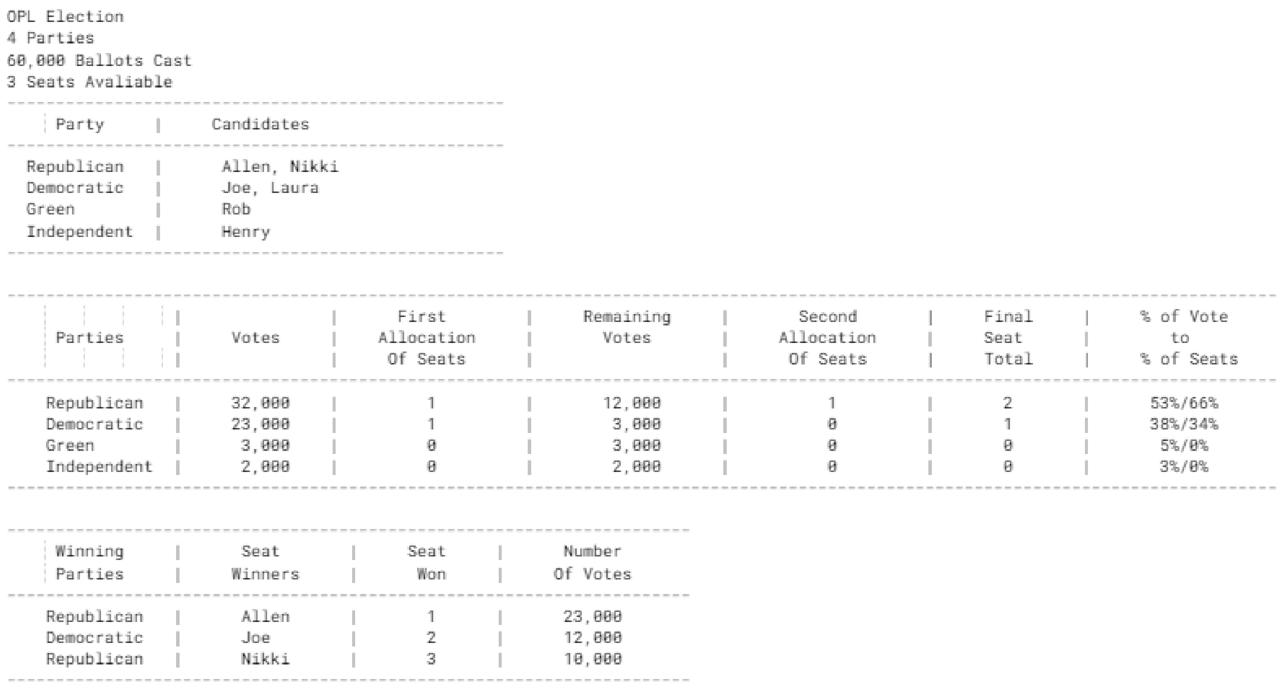
Example Input of Ballots for CLP Election (Ex.4):

```
OPL
2
9
6
Democrat, Pike
Democrat, Lucy
Democrat, Beiye
Republican, Etta
Republican, Alawa
Independent1, Sasha
1,,,,,
1,,,,,
,1,,,,
,,,,1,
,,,,,1
,,,1,,
,,,1,,
,,,,1,
,,,,,1
```

Example Output to User for OLP Election (Ex. 5):

```
----------------------------------------------------------------------
     Winning     |       Seat       |     Seat      |        Number
     Parties     |      Winners     |      Won      |       Of Votes
----------------------------------------------------------------------
    Republican   |       Allen      |       1       |        23,000
    Democratic   |       Joe        |       2       |        12,000
    Republican   |       Nikki      |       3       |        10,000
----------------------------------------------------------------------
Audit File saved: C:\Users\freem627\Documents\AuditFiles\OPL_Election_Results_2024-02-12_10:38:47
```

Example Output to Audit File for OLP Election (Ex. 6):



```
OPL Election
4 Parties
60,000 Ballots Cast
3 Seats Avaliable
---------------------------------------------------
    Party       |     Candidates
---------------------------------------------------
  Republican    |      Allen, Nikki
  Democratic    |      Joe, Laura
  Green         |      Rob
  Independent   |      Henry
---------------------------------------------------

-----------------------------------------------------------------------------------------------------------------------------------
                 |                |      First      |     Remaining     |      Second      |     Final      |     % of Vote
    Parties      |     Votes      |   Allocation    |       Votes       |    Allocation    |     Seat       |        to
                 |                |    Of Seats     |                   |     Of Seats     |     Total      |    % of Seats
-----------------------------------------------------------------------------------------------------------------------------------
    Republican   |     32,000     |        1        |      12,000       |        1         |       2        |     53%/66%
    Democratic   |     23,000     |        1        |       3,000       |        0         |       1        |     38%/34%
    Green        |      3,000     |        0        |       3,000       |        0         |       0        |      5%/0%
    Independent  |      2,000     |        0        |       2,000       |        0         |       0        |      3%/0%
-----------------------------------------------------------------------------------------------------------------------------------

----------------------------------------------------------------------
    Winning      |      Seat      |     Seat      |     Number
    Parties      |     Winners    |      Won      |    Of Votes
----------------------------------------------------------------------
    Republican   |     Allen      |       1       |     23,000
    Democratic   |     Joe        |       2       |     12,000
    Republican   |     Nikki      |       3       |     10,000
----------------------------------------------------------------------
```