Team 4: Project 1 Testing Report

System Testing

1. System Test 1

Test Case ID:	ST_001	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with different input files	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	plurality1.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 10	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 5	System continue asking input for number of voter if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass

7.	System displays winner	NULL	Number of displayed winners should match the input, and candidate with high vote count in csv file should be selected as winner	As Expected	Pass

2. System Test 2

Test Case ID:	ST_002	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with different input files	Dependencies:	User inputs and content from CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System takes a CSV file	droop_quota2.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 6	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 2	System continue asking input for number of voter if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for	As Expected	Pass

			algorithm if the input is valid		
6.	System asks for type of algorithm	Algorithm = 1	System display winners	As Expected	Pass
7.	System displays winner	NULL	Number of displayed winners should match the input, and candidate with high vote count in csv file should be selected	As Expected	Pass

3. System Test 3

Test Case ID:	ST_003	Test Designed by:	Yuhao Li
Test Priority:	Medium	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required when run a droop_quota format file using plurality algorithm.	Dependencies:	User inputs and content from CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System takes a CSV file	droop_quota3.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 6	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number	Winner = 2	System continue	As Expected	Pass

	of winner		asking input for number of voters if the input is valid		
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass
7.	System displays winner	NULL	Number of displayed winners should match the input, and candidate with high vote ranking in csv file should be selected	As Expected	Pass

4. System Test 4

Test Case ID:	ST_004	Test Designed by:	Yuhao Li
Test Priority:	Medium	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required when run a plurality format file using droop quota algorithm.	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	plurality1.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass

3.	System asks for number of candidate	Candidate = 10	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 5	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 1	System display winners	As Expected	Pass
7.	System displays winner	NULL	Number of displayed winners should match the input, and candidate with high vote count in csv file should be selected as winner	As Expected	Pass

5. System Test 5

Test Case ID:	ST_001	Test Designed by:	Yuhao Li
Test Priority:	Low	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with a csv file that is not existing.	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV	notExist.csv	System start asking	As Expected	Pass

	file		user input		
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 10	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 5	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

6. System Test 6

Test Case ID:	ST_006	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with different csv file	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	droop_quota2.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 14	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 3	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 1	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

7. System Test 7

Test Case ID:	ST_007	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with different csv file	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	plurality2.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 5	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 2	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 11	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

8. System Test 8

Test Case ID:	ST_008	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with different csv file	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	plurality3.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 5	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 3	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 13	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

9. System Test 9

Test Case ID:	ST_009	Test Designed by:	Yuhao Li
Test Priority:	High	Test Designed date:	03/20/2018
Executed by:	Yuhao Li	Test execution date:	03/20/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid
Description:	Ensure the entire program works as the specifications required with	Dependencies:	Input CSV file

	different csv file		

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	droop_quota3.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 5	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 2	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 7	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 1	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

10. System Test 10

Test Case ID:	ST_010	Test Designed by:	Xueman Liang
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Xueman Liang	Test execution date:	03/21/2018
Test Title:	Test final result from the system	Pre-conditions:	The csv file is in a correct format, and user inputs are valid

Description:	Ensure the entire program works as the specifications required with different csv file	Dependencies:	Input CSV file

Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.	System take a CSV file	Plurality4.csv	System start asking user input	As Expected	Pass
2.	System asks to choose running model (Testing / No Testing)	Test Model = 1	System continue asking input for number of candidate if the input is valid	As Expected	Pass
3.	System asks for number of candidate	Candidate = 6	System continue asking input for number of winner if the input is valid	As Expected	Pass
4.	System asks for number of winner	Winner = 2	System continue asking input for number of voters if the input is valid	As Expected	Pass
5.	System asks for number of voter	Voter = 10	System continue asking input for algorithm if the input is valid	As Expected	Pass
6.	System asks for type of algorithm	Algorithm = 0	System display winners	As Expected	Pass
7.	System displays winner	NULL	An error message should appear since notExist.csv is not existing.	As Expected	Pass

Unit Testing

1. Unit Test 1

Test Case ID:	UT_001	Test Designed by:	Floyd Chen
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Floyd Chen	Test execution date:	03/21/2018
Test Title:	JUnit Test for Droop Quota Algorithm	Pre-conditions:	Correctly set the database for the DroopQuotaTest class to run
Description:	Use JUnit test to ensure the droop quota algorithm works correctly	Dependencies:	JUnit4, Database class

Step	Test Steps	Expected Result	Actual Result	Status
1.	Use data of the input file	None	None	Pass
2.	Initialize a Database instance	New database instance initialized	As Expected	Pass
3.	Initialize HashMap Votes in Database	db.votes initialized	As Expected	Pass
4.	Initialize the shuffled list in Database	db.shuffled_list initialized	As Expected	Pass
5.	Initialize winners and losers list as empty list	db.winners and db.losers initialized	As Expected	Pass
6.	DroopQuota dq = new DroopQuota(db);	New DroopQuota instance initialized with input parameter “db”	As Expected	Pass
7.	Call the run() method	Write losers and winners in the db	As Expected	Pass
8.	Use assertEquals() to check the winners and losers in the two list match	Winners are “A” and “C”, the loser is “B”	As Expected	Pass

Console:
 	
`====== Election Results Using Droop Quota Algorithm =====`
`WINNER(S):`
` A: 2`
` C: 1`
 	
`LOSER(S):`
` B: 1`

Input:
 	
`A,B,C`
`1,,`
`2,,1`
`,,`
`1,2,`

2. Unit Test 2

Test Case ID:	UT_002	Test Designed by:	Floyd Chen
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Floyd Chen	Test execution date:	03/21/2018
Test Title:	JUnit Test for Plurality Algorithm	Pre-conditions:	Correctly set the database for the PluralityTest class to run
Description:	Use JUnit test to ensure the plurality algorithm works correctly	Dependencies:	JUnit4, Database class

Step	Test Steps	Expected Result	Actual Result	Status
1.	Initialize a Database instance	New database instance initialized	As Expected	Pass
2.	Initialize HashMap Votes in Database	db.votes initialized	As Expected	Pass
3.	Initialize the shuffled list in Database	db.shuffled_list initialized	As Expected	Pass

4.	Initialize winners and losers list as empty list	db.winners and db.losers initialized	As Expected	Pass
5.	Plurality p = new Plurality(db);	New Plurality instance initialized with input parameter “db”	As Expected	Pass
7.	Call the run() method	Write losers and winners in the db	As Expected	Pass
8.	Use assertEquals() to check the first two winners	The first two winners are “F” and “A”, the loser is “B”	As Expected	Pass
9.	Use assertEquals() to check the third winners	The third winner can be one of “B”, “C” and “D”	As Expected	Pass

Console:

```
====== Election Results Using Plurality Algorithm =====
WINNER(S):
 F: 4
 A: 3
 D: 2

LOSER(S):
 B: 2
 C: 2
 E: 0
```

Input:

```
A,B,C,D,E,F
1,,,,,
,,1,,,
,,,,,1
1,,,,,
,,,1,,
,,,,,1
,1,,,,
,,,1,,
,,,,,1
,,,,,1
,,1,,,
,1,,,,
1,,,,,
```

3. Unit Test 3

Test Case ID:	UT_003	Test Designed by:	Floyd Chen
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Floyd Chen	Test execution date:	03/21/2018
Test Title:	JUnit Test for read_file()	Pre-conditions:	1. Valid input path string 2. VotingSystem instance 3. Database instance
Description:	Use JUnit test the read_file() method in VotingSystem class	Dependencies:	JUnit4, Database class

Step	Test Steps	Expected Result	Actual Result	Status
1.	Initialize a Database instance and provide input string path	Database instance initialized	None	Pass
2.	Use setters the “database” and “filename”	Update the two “database” and “filename” private variables in VotingSystem class	As Expected	Pass
3.	Use try and catch to check if the read_file() throw an exception	No exception thrown	As Expected	Pass

4. Unit Tests 4

Test Case ID:	UT_004	Test Designed by:	Floyd Chen
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Floyd Chen	Test execution date:	03/21/2018
Test Title:	JUnit Test for write_output()	Pre-conditions:	4. Valid input path string 5. VotingSystem

			instance 6. Database instance
Description:	Use JUnit test the write_output() method in VotingSystem class	Dependencies:	JUnit4, Database class

Step	Test Steps	Expected Result	Actual Result	Status
1.	Initialize a Database instance and provide input string path	Database instance initialized	None	Pass
2.	Use setters the “database” and “filename”	Update the two “database” and “filename” private variables in VotingSystem class	As Expected	Pass
3.	Use try and catch to check if the write_output() throw an exception	No exception thrown	As Expected	Pass

5. Unit Tests 5

Test Case ID:	UT_005	Test Designed by:	Floyd Chen
Test Priority:	High	Test Designed date:	03/21/2018
Executed by:	Floyd Chen	Test execution date:	03/21/2018
Test Title:	JUnit Test for write_report()	Pre-conditions:	7. Valid input path string 8. VotingSystem instance 9. Database instance
Description:	Use JUnit test the write_report() method in VotingSystem class	Dependencies:	JUnit4, Database class

Step	Test Steps	Expected Result	Actual Result	Status
1.	Initialize a Database instance and provide input string path	Database instance initialized	None	Pass
2.	Use setters the “database” and “filename”	Update the two “database” and “filename” private variables in VotingSystem class	As Expected	Pass
3.	Use try and catch to check if the write_report() throw an exception	No exception thrown	As Expected	Pass