Project Name: Project 2: Agile Scrum Team 21

Test stage: System **Test Date:** 04/18/2023
Test Case ID#: CPL_REMAINDER_1 **Name of Tester(s):** Cuong Ha
Test Description: This is a system/integration testing for the remainder round of CPL to make sure the results returned are reasonable. The test will take place under the CLI mode.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command \`make\`. Must have a valid election .csv file to input and it must be in either Project2 or Project2/testing. Must be on the "Election Information" view page with an entered valid CPL election file displayed.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	1. Run the program using \`make run\` 2. Enter "cpl_input_test_10.csv" as the input file name 3. Confirm the file 4. Continue until reaching the final results	N/A	Seats are reasonably allocated to each party according to their ranks	Matched expectations	

Postcondition(s) for Test:
An audit file containing higher level details is created and the results written in it match the results displayed on the screen in the final layout, and should be reasonable.

Test stage: System **Test Date:** 04/18/2023
Test Case ID#: CPL_REMAINDER_1 **Name of Tester(s):** Cuong Ha
Test Description: This is a system/integration testing for the remainder round of CPL to make sure the results returned are reasonable. The test will take place under the CLI mode.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command `make`. Must have a valid election .csv file to input and it must be in either Project2 or Project2/testing. Must be on the "Election Information" view page with an entered valid CPL election file displayed.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	1. Run the program using `make run` 2. Enter "cpl_input_test_11.csv" as the input file name 3. Confirm the file 4. Continue until reaching the final results	N/A	Seats are reasonably allocated to each party according to their ranks	Matched expectations	

Postcondition(s) for Test:
An audit file containing higher level details is created and the results written in it match the results displayed on the screen in the final layout, and should be reasonable.

Test stage: System **Test Date:** 04/18/2023
Test Case ID#: CPL_REMAINDER_1 **Name of Tester(s):** Cuong Ha
Test Description: This is a system/integration testing for the remainder round of CPL to make sure the results returned are reasonable. The test will take place under the CLI mode.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command `make`. Must have a valid election .csv file to input and it must be in either Project2 or Project2/testing. Must be on the "Election Information" view page with an entered valid CPL election file displayed.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	1. Run the program using `make run` 2. Enter "cpl_input_test_12.csv" as the input file name 3. Confirm the file 4. Continue until reaching the final results	N/A	Seats are reasonably allocated to each party according to their ranks	Matched expectations	

Postcondition(s) for Test:
An audit file containing higher level details is created and the results written in it match the results displayed on the screen in the final layout, and should be reasonable.

Test stage: Unit **Test Date:** 04/24/2023
Test Case ID#: date_input_1 **Name of Tester(s):** Brian Bianchi
Test Description: This is a set of unit tests to make sure that the string output by the HandleDateString function has all '/' and '.' replaced by '_'.
Automated: Yes
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: make. Test file is made with the following command: "g++ -Wall -g -pthread -o DateTesting testing/date_input_utest.cc src/controller.cc src/cli.cc src/election.cc src/ir.cc src/cpl.cc src/candidate.cc src/ballot.cc src/party.cc -lgtest_main -lgtest -lpthread " (Everything inside the "", not including them).

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	1. Run the test using ./DateTesting	The first test automatically tried is of string "04_19_2023"	No changes should be made to that string; it should output "04_19_2023"	Matched expectations	
2	Second test is automatically run testing string with periods	The second test automatically tried is of string "04.19.2023"	The function should swap the '.' to '_' and output "04_19_2023"	Matched expectations	
3	Third test is automatically run testing string with slashes	The second test automatically tried is of string "04/19/2023"	The function should swap the '/' to '_' and output "04_19_2023"	Matched expectations	

Postcondition(s) for Test:
A string is output with the altered (or not) input void of periods or slashes.

Test stage: Unit **Test Date:** 04/24/2023
Test Case ID#: MULTI_FILE_HEADERS_IR **Name of Tester(s):** Cuong Ha
Test Description: This is a unit test for the IR that checks whether the modifications to header reading logic actually allows the system to read multiple headers from different input files.
Automated: No
Result: Pass
Preconditions for Test: Must have multiple valid election .csv files in either Project2 or Project2/testing director. If the modifications to this part of code have not been integrated to the original code yet, the original `main.cc` must be commented out first.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	1. Change directory to Project2/testing 2. Run the program using `g++ multi_header_utest_IR.cc ../src/controller.cc; ./a.out` 3. Confirm the result printed to the terminal	N/A	The number of ballots from four IR input files should be showed up like this at the bottom: 1080 6 107100 6904	Matched expectations	

Postcondition(s) for Test:
The vector containing all the numbers of ballots from all input files are readable/accessible by the testers or the system (if integrated) and are ready to be passed into an instance of IR class.

Test stage: Unit **Test Date:** 04/29/2023
Test Case ID#: IR_RESULTS_1 **Name of Tester(s):** Andrew Brevick
Test Description: This is a unit test to make sure that methods to display the change in vote count between rounds are working correctly
Automated: Yes
Result: Pass
Preconditions for Test: We construct a candidate, then add and remove ballots while checking that the ballot counting and change tracking is working as expected

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Construct Candidate	“John”,”R”	0	0	

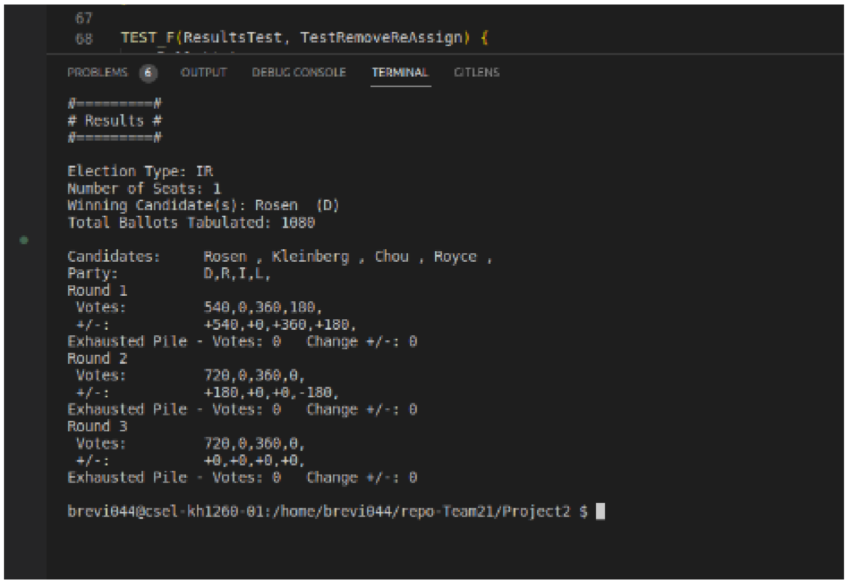
2	Assign Ballots	NA	3	3	
3	Eval Previous Count	NA	0	0	
4	Remove Ballotst	NA	2,2,3	2,2,3	

Postcondition(s) for Test:
Ballot counter and change tracker reflect changes to the ballots assigned to the test candidate.

Test stage: System **Test Date:** 04/29/2023
Test Case ID#: IR_RESULTS_2 **Name of Tester(s):** Andrew Brevick
Test Description: This is a system test to make sure that the updated IR results table is displaying correctly
Automated: No
Result: Pass
Preconditions for Test: Ballot files are created

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Run system nominally	IR_input_test.csv	Check all fields in results table. Including Candidate names, parties. For each round, we expect to see ballot counts and changes with appropriate sign. For this file, there are 1080 ballots, and we can check validity for the first candidate. Ballot count should be 540(+540),720(+180), 720(+0)	All checks match expectations.	
2	Run system with dead ballots	IR_input_test_3.csv	All checks as above, plus additional checks that the “exhasuted pile” aka dead ballots is working as expected. The first dead ballots should appear in the 6th round (+12600)	All checks match expectations.	

Test 1



```
TEST_F(ResultsTest, TestRemoveReAssign) {
```

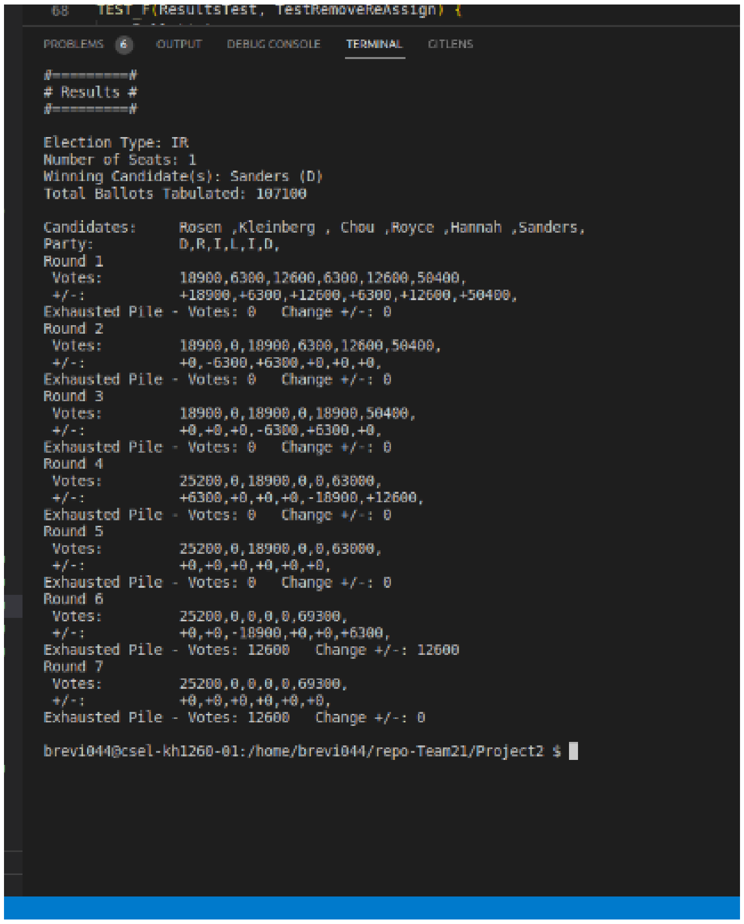
```
#=========#
# Results #
#=========#

Election Type: IR
Number of Seats: 1
Winning Candidate(s): Rosen  (D)
Total Ballots Tabulated: 1080

Candidates:     Rosen , Kleinberg , Chou , Royce ,
Party:          D,R,I,L,
Round 1
 Votes:         540,0,360,180,
 +/-:           +540,+0,+360,+180,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 2
 Votes:         720,0,360,0,
 +/-:           +180,+0,+0,-180,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 3
 Votes:         720,0,360,0,
 +/-:           +0,+0,+0,+0,
Exhausted Pile - Votes: 0   Change +/-: 0

brevi044@csel-kh1260-01:/home/brevi044/repo-Team21/Project2 $ 
```

Test 2



```
TEST_F(ResultsTest, TestRemoveReAssign) {
```

```
#=========#
# Results #
#=========#

Election Type: IR
Number of Seats: 1
Winning Candidate(s): Sanders (D)
Total Ballots Tabulated: 107100

Candidates:     Rosen ,Kleinberg , Chou ,Royce ,Hannah ,Sanders,
Party:          D,R,I,L,I,D,
Round 1
 Votes:         18900,6300,12600,6300,12600,50400,
 +/-:           +18900,+6300,+12600,+6300,+12600,+50400,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 2
 Votes:         18900,0,18900,6300,12600,50400,
 +/-:           +0,-6300,+6300,+0,+0,+0,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 3
 Votes:         18900,0,18900,0,18900,50400,
 +/-:           +0,+0,+0,-6300,+6300,+0,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 4
 Votes:         25200,0,18900,0,0,63000,
 +/-:           +6300,+0,+0,+0,-18900,+12600,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 5
 Votes:         25200,0,18900,0,0,63000,
 +/-:           +0,+0,+0,+0,+0,+0,
Exhausted Pile - Votes: 0   Change +/-: 0
Round 6
 Votes:         25200,0,0,0,0,69300,
 +/-:           +0,+0,-18900,+0,+0,+6300,
Exhausted Pile - Votes: 12600   Change +/-: 12600
Round 7
 Votes:         25200,0,0,0,0,69300,
 +/-:           +0,+0,+0,+0,+0,+0,
Exhausted Pile - Votes: 12600   Change +/-: 0

brevi044@csel-kh1260-01:/home/brevi044/repo-Team21/Project2 $ 
```

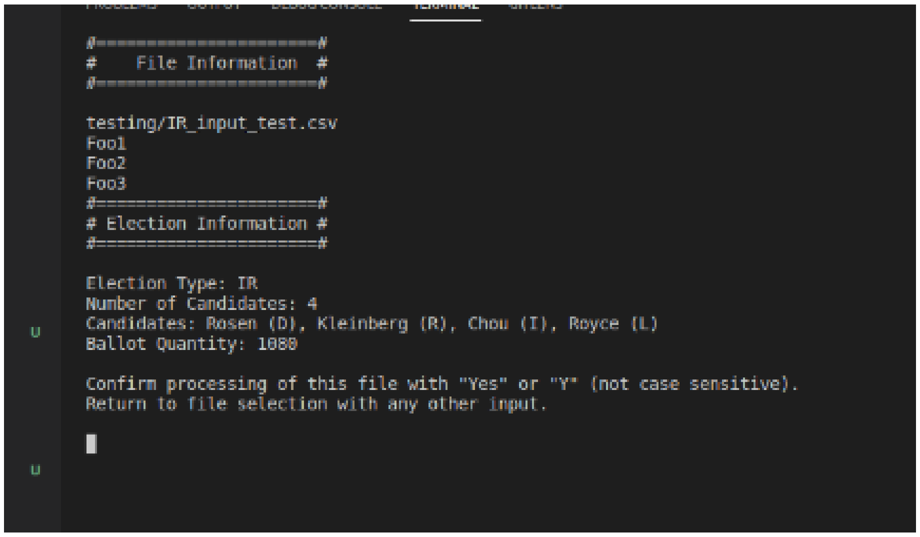
Postcondition(s) for Test:

The program runs nominally, and displays correct results in the expected format.

Test stage: System **Test Date:** 04/29/2023
Test Case ID#: MULTI_INPUT_1 **Name of Tester(s):** Andrew Brevick
Test Description: This is a test that multiple filenames can be input from the command line interface and reported back to the used
Automated: No
Result: Pass
Preconditions for Test: NA

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Enter files	"IR_input_test.csv","foo1","foo2","foo3"	NA	NA	This includes indicating "Y" after each file is input
2	Check that internal file list matches input		"IR_input_test.csv","foo1","foo2","foo3"	"IR_input_test.csv","foo1","foo2","foo3"	

Exact user inputs are as follows:
04-29-2023 [Enter]
IR_input_test.csv [Enter]
Y [Enter]
Foo1 [Enter]
Y [Enter]
Foo2 [Enter]
Y [Enter]
Foo3 [Enter]
N [Enter]



```
#=====================#
#    File Information  #
#=====================#

testing/IR_input_test.csv
Foo1
Foo2
Foo3
#=====================#
# Election Information #
#=====================#

Election Type: IR
Number of Candidates: 4
Candidates: Rosen (D), Kleinberg (R), Chou (I), Royce (L)
Ballot Quantity: 1080

Confirm processing of this file with "Yes" or "Y" (not case sensitive).
Return to file selection with any other input.
```

Postcondition(s) for Test:
The program runs nominally, and displays filenames in the expected format.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: MULTI_INPUT_2 **Name of Tester(s):** Andrew Brevick
Test Description: This is a test that multiple filenames can be input as command arguments
Automated: No
Result: Pass
Preconditions for Test: NA

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Enter files	./Tabulator 04-30-2023 IR_input_test.csv IR_input_test_1.csv	Program executes, skipping file entry	As expected	
2	Check that internal file list matches input		"IR_input_test.csv","IR_input_test_1.csv"	"IR_input_test.csv","IR_input_test_1.csv"	

Postcondition(s) for Test:
The program runs nominally, and displays filenames in the expected format.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_cpl_1 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with a CPL election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid CPL election file is present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing	Matched expectations	

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_cpl_2 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with a CPL election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid CPL election file is present.

Step #	**Test Step Description**	**Test Data**	**Expected Result**	**Actual Result**	**Note**
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "y"	CLI should await an additional election file	Matched expectations	
3	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	Same file to ensure header info constant; Only ballot total changes
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing with a vote total double that of a single load of the file	Matched expectations	

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_cpl_3 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with a CPL election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid CPL election file is present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "y"	CLI should await an additional election file	Matched expectations	
3	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	Same file to ensure header info constant; Only ballot total changes
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing with a vote total double that of a single load of the file	Matched expectations	

5	Enter an answer	Entered "n"	CLI should redisplay date entry	Matched expectations	
6	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
7	Enter a testing file	Entered "cpl_input_test_3.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
8	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing with a vote total of just one copy of the file!	Matched expectations	This is to show that going back and re-reading files still works properly, resetting info entered prior

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_ir_1 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with an IR election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid IR election file is present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "IR_input_test.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing	Matched expectations	

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_ir_2 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with an IR election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid IR election file is present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "IR_input_test.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "y"	CLI should await an additional election file	Matched expectations	
3	Enter a testing file	Entered "IR_input_test.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	Same file to ensure header info constant; Only ballot total changes
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing with a vote total double that of a single load of the file	Matched expectations	

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.

Test stage: System **Test Date:** 04/30/2023
Test Case ID#: multi_header_ir_3 **Name of Tester(s):** Brian Bianchi
Test Description: This is a system test to ensure that multiple header functionality works as intended with an IR election.
Automated: No
Result: Pass
Preconditions for Test: In a shell/command line, navigated to and in Project2 folder, Tabulator built using command: "make" (inside and not including quotes); a valid IR election file is present.

Step #	Test Step Description	Test Data	Expected Result	Actual Result	Note
1	Start the program with the command "make run"	N/A	Brought to a CLI screen and prompted to enter a date	Matched expectations	
2	Enter a testing date	Entered "04-30-2023"	Prompted by the CLI to enter a file name	Matched expectations	
3	Enter a testing file	Entered "IR_input_test.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
4	Enter an answer	Entered "y"	CLI should await an additional election file	Matched expectations	
3	Enter a testing file	Entered "IR_input_test.csv"	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	Same file to ensure header info constant; Only ballot total changes
4	Enter an answer	Entered "n"	CLI should depict accurate header information and await confirmation for processing with a vote total double that of a single load of the file	Matched expectations	

5	Enter an answer	Entered “n”	CLI should redisplay date entry	Matched expectations	
6	Enter a testing date	Entered “04-30-2023”	Prompted by the CLI to enter a file name	Matched expectations	
7	Enter a testing file	Entered “IR_input_test.csv”	Prompted by the CLI to answer whether we have additional files to input or not	Matched expectations	
8	Enter an answer	Entered “n”	CLI should depict accurate header information and await confirmation for processing with a vote total of just one copy of the file!	Matched expectations	This is to show that going back and re-reading files still works properly, resetting info entered prior

Postcondition(s) for Test:
The software is awaiting further interaction to conclude tabulation, but this test is primarily of the ability to enter process multiple file *headers*, as such, the program can be safely ignored or terminated from here. The depiction of the headers here indicates the presence of all headers in program memory for any future utilization of them.