|  |  |
| --- | --- |
| Project Name: Project 1: Voting System Team# 25 | |
| Test Stage: Unit \_1\_ System \_\_ | Test Date: Nov 18, 2019 |
| Test Case ID#: CPL\_001 | Name(s) of Testers: YongFeng Ji |
| Test Description: To see if the CPL class could calculate the correct number of votes for each parties or not. |  |
| Automated: yes\_\_1\_ no \_\_\_ | Indicate where are you storing the tests (what file) and the name of the method/functions being used. |
| Results: Pass \_\_\_\_\_ Fail\_\_\_\_\_\_\_\_ |  |
|  |  |
| Preconditions for Test:  CPL code should be properly implemented, function like Increment, get\_parties, display, create\_text\_file | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step  # | Test Step  Description | Test  Data | Expected  Result | Actual  Result | Notes |
| 1 | get\_parties(), it will create a party vector, test will determine whether the vector have the correct party, and its candidates or not. And the correct number of parties. | CSV file: Include all parties and its candidate information |  |  |  |
| 2 | Increment(), it will calculate the number of votes for each parties. We test it automated with code by call increment function and see if the selected party got correct votes or not. | CSV file: Include all parties and its candidate information, and party votes |  |  |  |
| 3 | cretae\_text\_file(), it function will generate a text file which include all the parties and its candidates, and the votes each parties get. To test this file, we call the function, and check if the text file is correct or not. | CSV file: Include all parties and its candidate information, and party votes |  |  |  |
| 4 | Display(), this function will print the winning party and its candidates on command line, to check this function, we call it, and manually check it with the CSV file to see if this is correct or not. | CSV file: Include all parties and its candidate information, and party votes |  |  |  |
|  |  |  |  |  |  |

Post condition(s) for Test: a text file of CPL will be generated, and each parties vote will be calculated

Project Name: The project #, name of your system, and the team#

Test Stage: Indicate whether it is a unit test or a system test.

Test Date: The date the test was performed.

Test Case ID#: A unique ID is required. Decide on a naming convention and use numbering. Example: Ballot\_Shuffle\_1

Name(s) of Testers: List the names of anyone involved in running this test case.

Test Description: Describe briefly the test objective.

Automated: Indicate if the test is completely automated or being checked manually. (If you have methods running the tests and checking results, select “yes”. If you are manually checking results, indicate manual by selecting the “no.”)

Results: Indicate if the test passed or failed.

Step #: You will be listing the test steps in order. This number is the step number in the process.

Test Step Description: Details of the test step.

Test Data: What the test data will be for this step. Be clear on what the input data will be. If using a specific file, be clear on the name.

Expected Result:  What result are you expecting from the program component or system.

Actual Result:  What result were returned based on the test.

Post condition for Test: What will be true after the test has been run? Has the state of the system changed in any way?

Notes: Comments and notes for you and your team members.