|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team# 25** | |
| **Test Stage: Unit √\_\_ System \_\_** | **Test Date: Dec 14** |
| **Test Case ID#: 001** | **Name(s) of Testers: Jing Wu** |
| **Test Description:**  **The Processor( ) class has one public function Processor::process( ) and other private functions like Processor::parse\_OPL\_line( ), Processor::parse\_CPL\_line( ) and Processor::get\_one\_index( ). Since the only pubic function is Processor::process( ), the ProcessorTests.cc file contains the three tests for Processor::process( ) function. The other functions are tested manually.** |  |
| **Automated: yes\_√\_ 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: The input files exist, OPL and CPL class work well.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test Processor::process() | Input1 = “testdata/OPLScenario1.csv” | Output1 = OPLScenario1Expected.txt | Scenario: A tie with enough seats remaining.  OPL  3 seats  9 ballots cast  6 candidates  Pike,D 3 votes  Foster,D 2 votes  Deutsch,R 0 votes  Borg,R 2 votes  Jones,R 1 votes  Smith,I 1 votes  Winner [Pike,D] -- First assigned  Winner [Foster,D] -- These two are assigned next  Winner [Borg,R]  No tie-breakers needed since the amount of candidates  in a tie are equal to the amount of seats available. | Expected is identical to Actual result. This test is implemented by ProcessorTests.cc |
| 2 | Test Processor::process() | Input2 = “testdata/CPLScenario1.csv” | Output2 = CPLScenario1Expected.txt | There are 26 votes, so the threshhold to award a seat is 26/7,  which evaluates to about 4 (should always ceiling it). The 'R'  party should get two seats, then the 'G' party should get two  seats, and the 'D' party should get one seat. There are two seats  remaining and the 'D', 'R', and 'I' parties have the same number  for the remainder, which is 2. A coin flip is done and then the  winner gets a seat. Another coin flip is done between the remaining  two to assign the last seat. Seats should be assigned by respective  ordering of the party. | Expected is identical to Actual result. This test is implemented by ProcessorTests.cc |
| 3 | Test Processor::process() | Input3 = “badfile.abc” | Output3 = “Invalid file” | “Invalid file” | Expected is identical to Actual result. This test is implemented by manually check. |
| 4 | Test Processor::parse\_OPL\_line() | Input = lines of OPL files | Output = 0 | Output = 0 | This function generated audit file and return 0 as success. This test is implemented by manually check. |
| 5 | Test Processor::parse\_CPL\_line() | Input = lines of CPL files | Output = 0 | Output = 0 | This function generated audit file and return 0 as success. This test is implemented by manually check. |
| 6 | Test Processor::get\_one\_index() | Input = lines of files | Output = 6 | Output = 6 | The output is the number of the ballots for candidates. This test is implemented by manually check. |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

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 notesfor you and your team members.