|  |  |
| --- | --- |
| **Project Name: Aegis Project 1: Voting System Team#7** | |
| **Test Stage: Unit X System \_** | **Test Date:** 3/27/21 |
| **Test Case ID#:** Helper\_6: extractOPLNames | **Name(s) of Testers:** Michael/Donald/Lucky/Grant |
| **Test Description:**  Given a string in the format of an OPL election, parse it and extract the candidates and parties into a string vector. |  |
| **Automated: yes X no \_\_\_** | **Indicate where you are storing the tests (what file) and the name of the method/functions being used.**  Stored in file:  Method names:   * extractOPLNames() |
| **Results: Pass \_\_X\_\_\_** Fail **\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  The strings containing candidates are set up correctly as well as any other variables. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
|  |  |  |  |  |  |
| 1 | Set up the strings that will be used for the test. |  |  |  |  |
| 2 | Test for random string | std::string line = "[a1][a2][a3]"; | EXPECT\_THAT(extractOPLNames(line), testing::ElementsAre("a1", "a2", "a3")) = true; | EXPECT\_THAT(extractOPLNames(line), testing::ElementsAre("a1", "a2", "a3")) = true; |  |
| 3 | Test for empty string | std::string empty = ""; | EXPECT\_EQ(0, extractIRNames(empty\_line).size()) = true | EXPECT\_EQ(0, extractIRNames(empty\_line).size()) = true |  |
| 4 | Test for random string | std::string weird = "[a1]a2[a3]"; | EXPECT\_THAT(extractOPLNames(weird), testing::ElementsAre("a1", "a3")) = true | EXPECT\_THAT(extractOPLNames(weird), testing::ElementsAre("a1", "a3")) = true |  |

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

The value returned should be a string vector containing the candidate names and parties.