Test 1: sortCandidateVotesTest
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<OPLParty> OPLParties, std::vector<CandidateOPL> OPLCandidates, and std::vector<Ballot> candidateBallots. Run sortCandidateVotes once the three vectors are filled with data.
Tests: Test to see if the OPLParties size matches with the parties pushed into the vector. Test to see if the candidate in the party with the most votes is in the first index of the vector. Test to see if the candidate with the lower vote count is in the second index of the vector.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 2: largestRemainderSeatAllocationTest_Party
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<Party> parties, std::vector<Seat> seats, std::vector<Ballot> PartyBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data
Tests: Test to see if the size of the seats vector matches with the size of the parties vector. Test to see if the first index of the seats vector has 0 seats given. Test to see if the second index of the seats vector has 1 seat given. Test to see if the third index of the seats vector has 0 seats given. Test to see if the forth index of the seats vector has 1 seat given.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 3: largestRemainderSeatAllocationTest_OPLParty
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<OPLParty> parties, std::vector<Seat> seats, std::vector<Ballot> PartyBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data
Tests: Test to see if the size of the seats vector matches with the size of the parties vector. Test to see if the first index of the seats vector has 1 seat given. Test to see if the second index of the seats vector has 1 seat given. Test to see if the third index of the seats vector has 0 seats given.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 4: largestRemainderSeatAllocationTest_MPOParty
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<OPLParty> parties, std::vector<Seat> seats, std::vector<Ballot> PartyBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data
Tests: Test to see if the size of the seats vector matches with the size of the parties vector. Test to see if the first index of seats has 1 seat given. Test to see if the second index of seats has 1 seat given.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 5: largestRemainderSeatAllocationTest_MVParty
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<CandidateMV> MCandidates, std::vector<Seat> seats, std::vector<Ballot> candidateBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data.
Tests: Test to see if the size of the seats vector matches with the size of the parties vector. Test to see if the first index of seats has 1 seat given. Test to see if the forth index of seats has 1 seat given. Test to see if the fifth index of seats has 1 seat given.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 6: OutputFunction1
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<OPLParty> parties, std::vector<Seat> seats, std::vector<Ballot> PartyBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data. RunOPL afterwards with the file set to “OPL.csv”. Then run output with parties and seats as the input.
Tests: Test to see if the result of running output from election.cpp with the input vectors (parties and seats) returns a 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 7: OutputFunction2
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<Party> parties, std::vector<Seat> seats, std::vector<Ballot> PartyBallots, int totalSeats, and int totalBallots. Run largest_reaminder_seat_allocation once these three vectors are filled with data. RunCPL afterwards with the file set to “CPL.csv”. Then run output with parties and seats as the input.
Tests: Test to see if the result of running output from election.cpp with the input vectors (parties and seats) returns a 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 8: RunMPOSystemTest
Team Member(s) Responsible:Arzab Bhattarai/Sam Bakri
Inputs: std::ifstream inputFile(“MPO.csv”) and string header. Run the runMPO function with the input file.
Tests: Test to see if result equals 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 9: RunMVSystemTest
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::ifstream inputFile(“MV.csv”) and string header. Run the runMV function with the input file.
Tests: Test to see if result equals 0.
Outputs: OK
Passed or Failed: Passed

Date: 4/20/24

Test 10: RunFunction
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<std::string> fileNameVector. Push two files into the vector: OPL.csv and CPL.csv. Run the run function with fileNameVector as the parameter
Tests: Test to see if result equals 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 11: RunCPLSystemTest
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::ifstream inputFile(“CPL.csv”) and string header. Run the runCPL function with the input file as the parameter.
Tests: Test to see if result equals 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 12: RunOPLSystemTest
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::ifstream inputFile(“OPL.csv”) and string header. Run the runOPL function with the input file as the parameter.
Tests: Test to see if result equals 0.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 13: promptForFiles_ValidFilenames
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: std::vector<std::string> expected Answer is set to {“OPL.csv, “CPL.csv”}, int numFiles is set to expectedAnswer.size(), std::stringstream input, and std::vector<std::string> resultAnswer. Run promptForFiles().
Tests: Test to see if numFiles is 2.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 14: FileExists
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: Use “CPL.csv” as a parameter for FileExists()
Tests: Test to see if the expected value is true once fileExists(“CPL.csv”) is called.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24

Test 15: FileDoesNotExist
Team Member(s) Responsible: Arzab Bhattarai/Sam Bakri
Inputs: Use “random.csv” as a parameter for FileExists(). Random.csv is not a file that exists within the directory.
Tests: Test to see if the expected value is false once fileExists(“random.csv”) is called.
Outputs: OK
Passed or Failed: Passed
Date: 4/20/24