|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bug# | Description of the bug | Location of Bug | Steps to Recreate Bug | Root Cause |
| 1 | Candidate’s votes cannot holds a limited value. | Candidate.java | Pass in a file with over 9223372036854775807 ballots going to a single candidate. | Limit of long in Java. |
| 2 | Candidate cannot set vote over 9223372036854775807. | Candidate.java | Pass in a file with over 9223372036854775807 ballots going to a single candidate. | Limit of long in Java. |
| 3 | Candidate cannot add votes once vote reaches 9223372036854775807. | Candidate.java | Pass in a file with over 9223372036854775807 ballots going to a single candidate. | Limit of long in Java. |
| 4. | Candidate.setBallots(ArrayList<Ballot> ballots) does not update the number of votes. | Candidate.setBallots(ArrayList<Ballot> ballots) | Not a function accessible to the user. A programmer would need to use Candidate.setBallots(ArrayList<Ballot> ballots) to change the Ballots of a Candidate. | Candidate.setBallots(ArrayList<Ballot> ballots) does not update the vote counter when it is resetting the list. |
| 5. | Candidate.getParty() returns null in IR voting | Candidate.getParty() | Not a function accessible to the user. A programmer would need to use Candidate.getParty() when Candidate does not have a Party. | Candidates are not specialized to deal with different election types that not care about the Party. |
| 6 | Candidate id variable may be never used and is may be never set. | Candidate.id | Not a function accessible to the user. A programmer would need to use a constructor that never sets the Id. Then the ids are not unique. | Constructor does not automatically generate unique id numbers. |
| 7 | Party’s CompareTo does not handle values over 9223372036854775807 values. | Party.compareTo(Party) | Pass in a file with over 9223372036854775807 ballots going to a single Party. | Limit of long in Java. |
| 8 | Party’s addTotalCounts does not handle adding to votes counts over 9223372036854775807 values. | Party.addTotalCounts(long) | Pass in a file with over 9223372036854775807 ballots going to a single Candidate. | Limit of long in Java. |
| 9 | Party’s addSeats does not handle adding to seats counts over 9223372036854775807 values. | Party.addSeats(long) | Pass in a file with over 9223372036854775807 seats going to a single Candidate. | Limit of long in Java. |
| 10 | Does not handle empty party and Candidate names properly. You will get Candidate and Party names of “”. | Candidate | Pass in a file with no value for candidate names or party names ie “[]” or Name “()” | Getters and setters for name and Party in Candidate and Party do not check null or empty strings. |
| 11 | Party.setName(null) does not throw an error. | Party.setName(null) | Function not handle case. | Function not handle case. |
| 12 | OPLVoting.run() and Util.writeAuditFile() does not handle elections where there are more seats than Candidates properly. It shows all the winners, but it doesn’t mention that there are less Candidates than seats | OPLVoting.run() and Util.writeAuditFile() | Pass in any OPL file with less Candidates than seats. Check the audit file. | OPLVoting.run() simply did not handle this case. |
| 13 | OPLVoting.parseInputFile(String filename) and IRVoting.parseInputFile(String filename) do not check if ballot number is consistent with the number of ballots shown or if the number of candidates listed on line is different from number listed, etc | OPLVoting.parseInputFile(String filename) and IRVoting.parseInputFile(String filename) | Pass in an OPL or IR file where the line with number of candidates is not consistent with the number listed on the other line. | Neither functions check for consistency in input files. |
| 14 | System does handle user input of a non-existent file. System will throw an error | VoteCounter | Enter filename of a file that’s not in the directory. | VoteCounter does not check for correct input. |