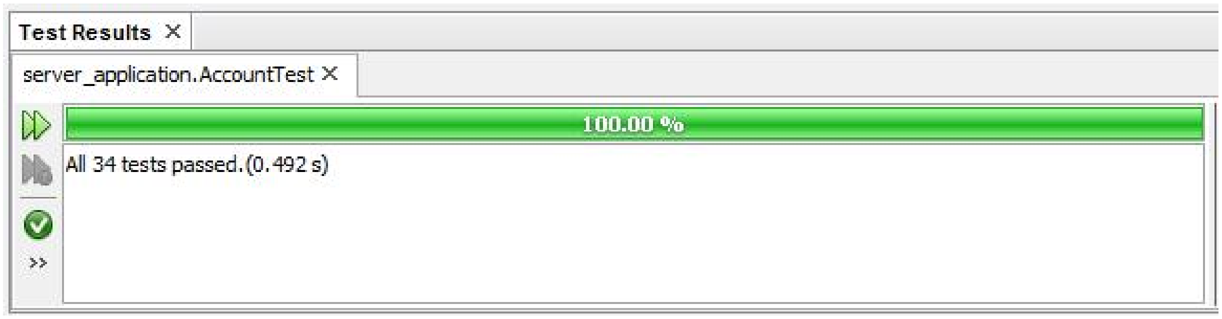
Appendices P

Unit Testing

1. Account Test

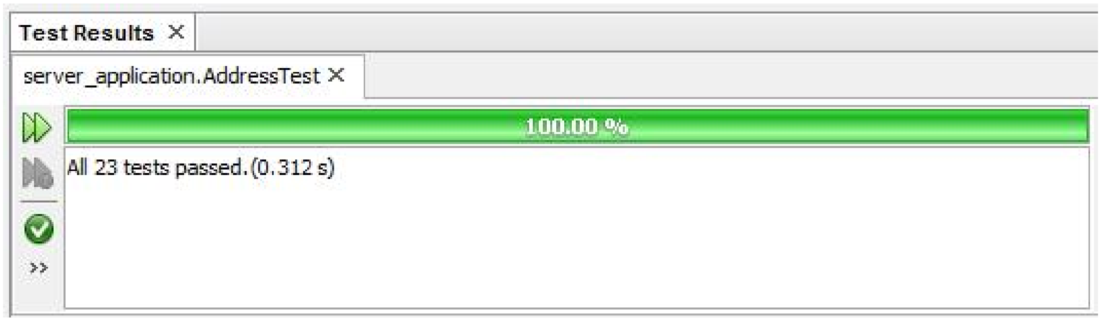


```
All 34 tests passed.(0.492 s)
```

Fig. 1 – Account Test Results

As you can see from Fig. 1, all 34 of the tests within the Account Test Class passed, which shows that all of the methods within the Account Class are working as they should, updating the state of the account object, as it should, and recording all of the modifications as it should. This means that a client of the Account class is able to create transactions, notes, and amend its variables and return the state of the object whilst managing the balance of the account when transactions are created or deleted, and keeping a record of any modifications made.

2. Address Test

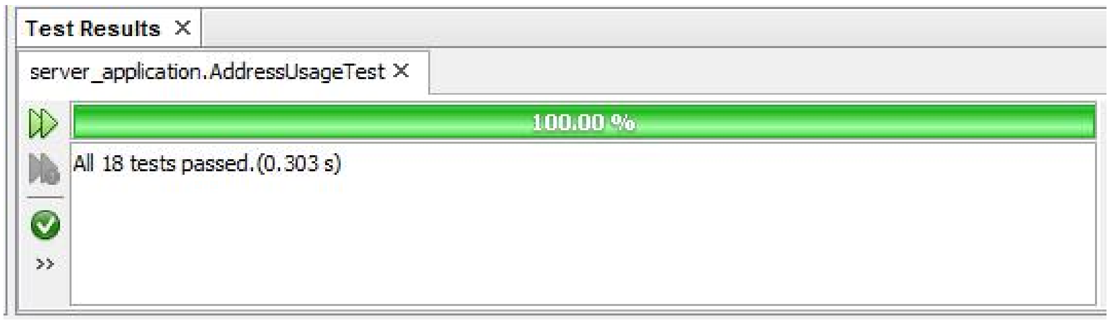


```
All 23 tests passed.(0.312 s)
```

Fig. 2 – Address Test Results

As you can see from Fig. 3, all 23 of the tests within the Address Test Class passed, which shows that all of the methods within the Address Class are working as they should, updating the state of an address object, as it should, and recording all of the modifications as it should. This means that a client of the Address class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

3. **AddressUsage Test**

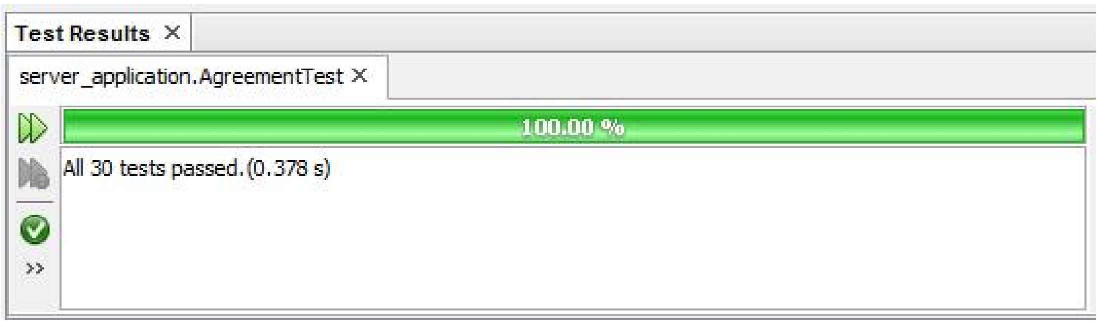


```
All 18 tests passed.(0.303 s)
```

Fig. 3 – AddressUsage Test Results

As you can see from Fig. 3, all 18 of the tests within the AddressUsage Test Class passed, which shows that all of the methods within the AddressUsage Class are working as they should, updating the state of an address usage object, as it should, and recording all of the modifications as it should. This means that a client of the AddressUsage class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

4. **Agreement Test**

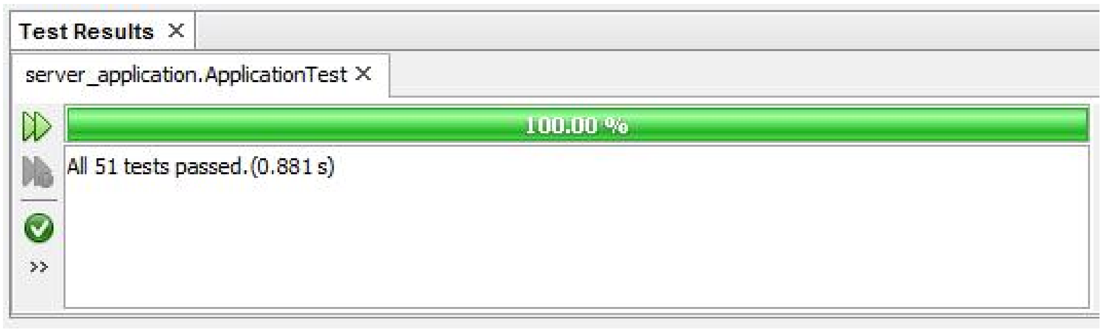


```
All 30 tests passed.(0.378 s)
```

Fig. 4 – Agreement Test Results

As you can see from Fig. 4, all 30 of the tests within the Agreement Test Class passed, which shows that all of the methods within the Agreement Class are working as they should, updating the state of an agreement object, as it should, and recording all of the modifications as it should. This means that a client of the Agreement class is able to create documents and notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

5. Application Test


Test Results ×

server_application.ApplicationTest ×

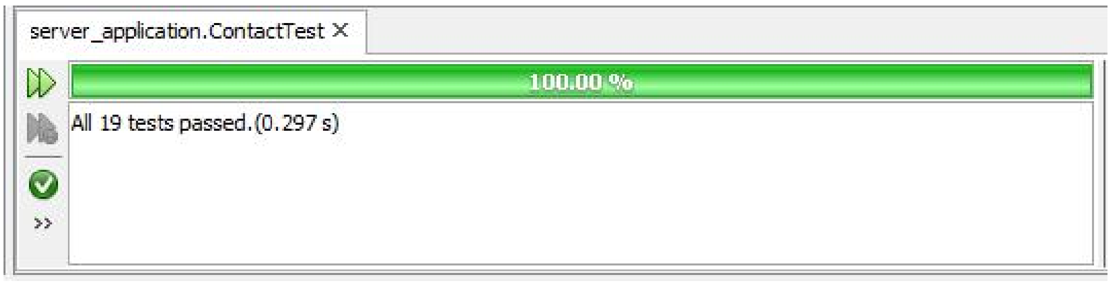
100.00 %

```
All 51 tests passed.(0.881 s)
```

Fig. 5 – Application Test Results

As you can see from Fig. 5, all 51 of the tests within the Application Test Class passed, which shows that all of the methods within the Application Class are working as they should, updating the state of an application object, as it should, and recording all of the modifications as it should. This means that a client of the Agreement class is able to create and delete documents, notes, involved parties, and properties interested in, amend its variables and return the state of the object whilst keeping a record of any modifications made.

6. Contact Test


server_application.ContactTest ×

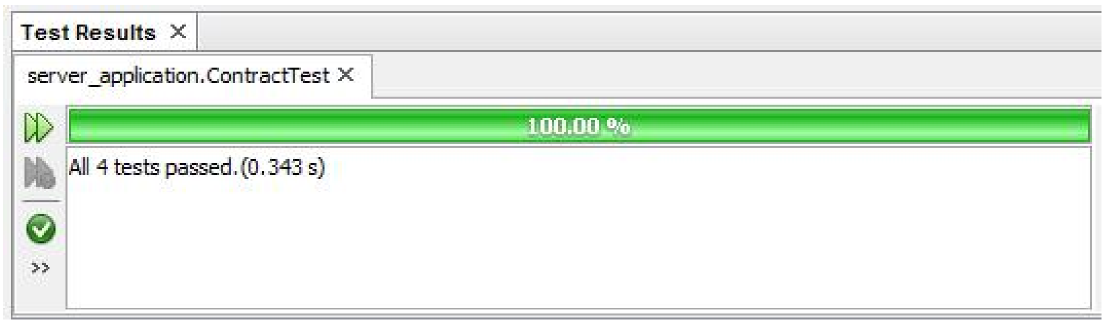
100.00 %

```
All 19 tests passed.(0.297 s)
```

Fig. 6 – Contact Test Results

As you can see from Fig. 6, all 19 of the tests within the Contact Test Class passed, which shows that all of the methods within the Contact Class are working as they should, updating the state of a contact object, as it should, and recording all of the modifications as it should. This means that a client of the Contact class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

7. Contract Test

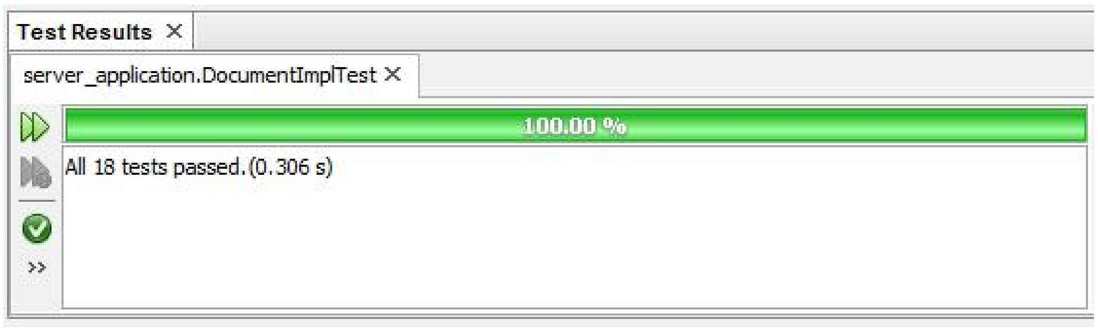


```
All 4 tests passed.(0.343 s)
```

Fig. 7 – Contract Test Results

As you can see from Fig. 7, all 4 of the tests within the Contract Test Class passed, which shows that all of the methods within the Contract Class are working as they should, updating the state of a contract object, as it should, and recording all of the modifications as it should. This means that a client of the Contract class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

8. Document Test

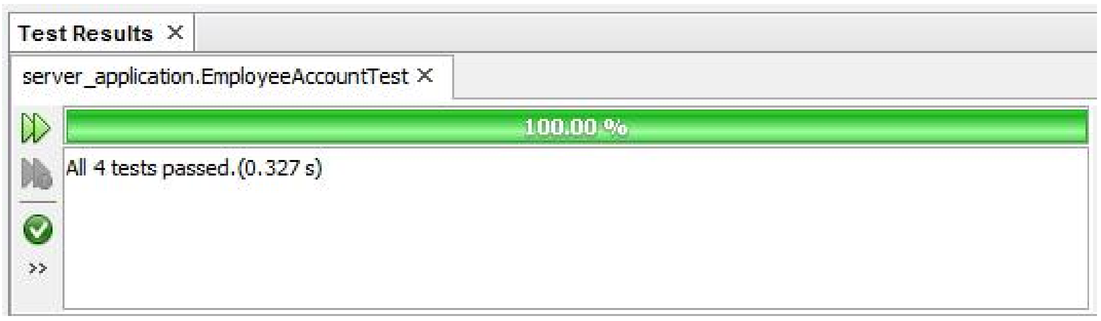


```
All 18 tests passed.(0.306 s)
```

Fig. 8 – Document Test Results

As you can see from Fig. 8, all 18 of the tests within the Document Test Class passed, which shows that all of the methods within the Document Class are working as they should, updating the state of a document object, as it should, and recording all of the modifications as it should. This means that a client of the Document class is able to create a new version of the file stored within the document class, amend its variables and return the state of the object whilst keeping a record of any modifications made.

9. EmployeeAccount Test

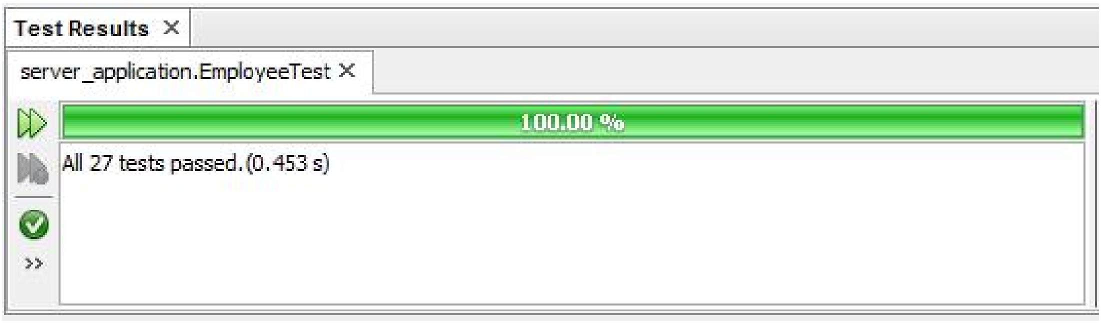


```
All 4 tests passed.(0.327 s)
```

Fig. 9 – EmployeeAccount Test Results

As you can see from Fig. 9, all 4 of the tests within the EmployeeAccount Test Class passed, which shows that all of the methods within the EmployeeAccount Class are working as they should, updating the state of an employee account object, as it should, and recording all of the modifications as it should. This means that a client of the EmployeeAccount class is able to create and delete transactions, documents, notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

10. Employee Test

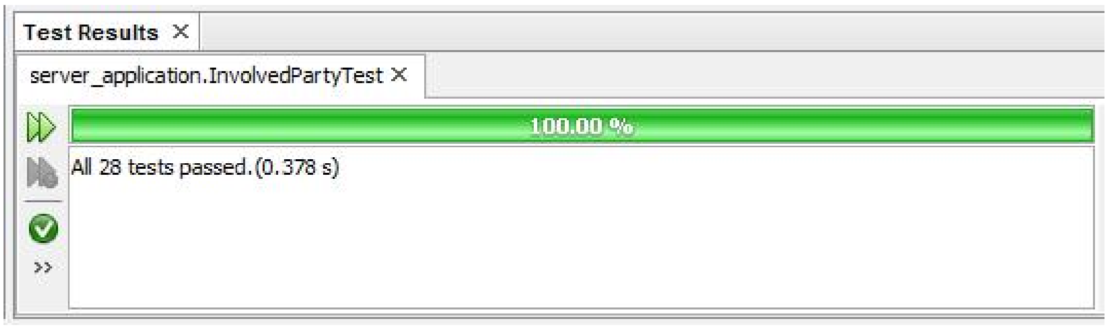


```
All 27 tests passed.(0.453 s)
```

Fig. 10 – Employee Test Results

As you can see from Fig. 10, all 27 of the tests within the Employee Test Class passed, which shows that all of the methods within the Employee Class are working as they should, updating the state of an employee object, as it should, and recording all of the modifications as it should. This means that a client of the Employee class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

11. InvolvedParty Test

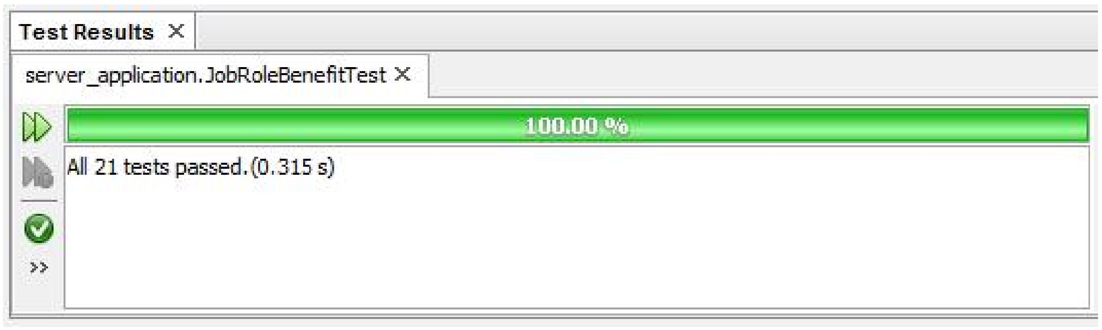


```
All 28 tests passed.(0.378 s)
```

Fig. 11 – InvolvedParty Test Results

As you can see from Fig. 11, all 28 of the tests within the InvolvedParty Test Class passed, which shows that all of the methods within the InvolvedParty Class are working as they should, updating the state of the involved party object, as it should, and recording all of the modifications as it should. This means that a client of the InvolvedParty class is able to create and delete notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

12. JobRoleBenefit

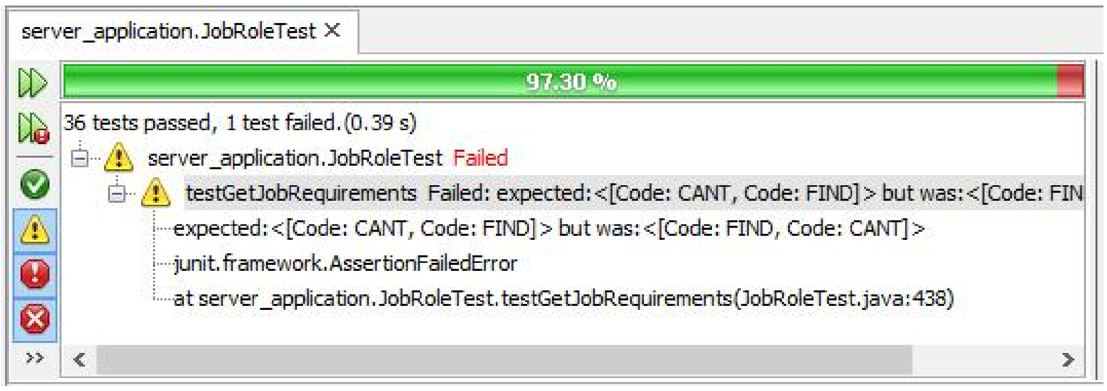


```
All 21 tests passed.(0.315 s)
```

Fig. 12 – JobRoleBenefit Test Results

As you can see from Fig. 12, all 21 of the tests within the JobRoleBenefit Test Class passed, which shows that all of the methods within the JobRoleBenefit Class are working as they should, updating the state of the job role benefit object, as it should, and recording all of the modifications as it should. This means that a client of the JobRoleBenefit class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

13. JobRole Test



```
36 tests passed, 1 test failed.(0.39 s)
server_application.JobRoleTest Failed
  testGetJobRequirements Failed: expected:<[Code: CANT, Code: FIND]> but was:<[Code: FIN
    expected:<[Code: CANT, Code: FIND]> but was:<[Code: FIND, Code: CANT]>
    junit.framework.AssertionFailedError
    at server_application.JobRoleTest.testGetJobRequirements(JobRoleTest.java:438)
```

Fig. 13 – JobRole Test Results

As you can see from Fig. 13, 36 of 37 tests within the JobRole Test Class passed, which shows that nearly all of the methods within the JobRole Class are working as they should, updating the state of the job role object, as it should, and recording all of the modifications as it should. This means that a client of the JobRole class is able to create and delete notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

The test that failed is the getJobRequirements() method, and I believe this has failed because I am trying to match the values of a HashMap<JobRequirements> as a list, with a list created within the test class, however because the HashMap<JobRequirements> key is a String, and a HashMap does not keep the elements in the same order, although I add the JobRequirements to both the HashMap and the test list in the same order, when the HashMap<JobRequirements> values is returned, these are not in the same order and therefore fails, however if you check the size of the list of values returned they are identical to the test list within the test class, so I believe that this method actually works as it should. The Stack Trace for the error is below:

```
expected:<[Code: CANT, Code: FIND]> but was:<[Code: FIND, Code: CANT]>

junit.framework.AssertionFailedError

        at server_application.JobRoleTest.testGetJobRequirements(JobRoleTest.java:438)
```

14. Landlord Test

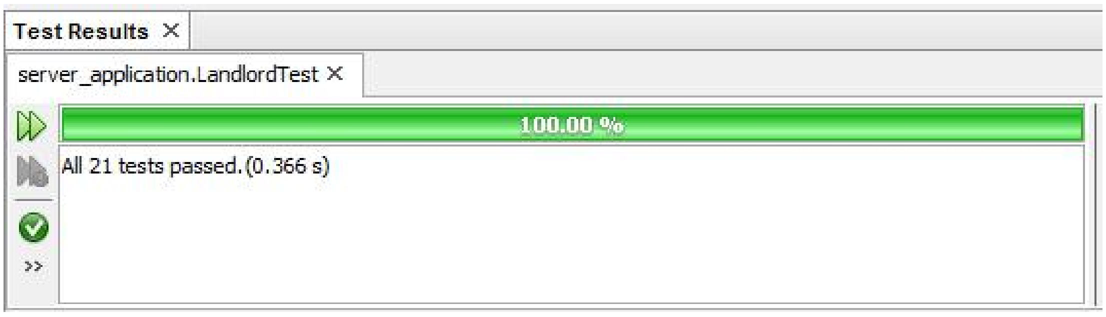
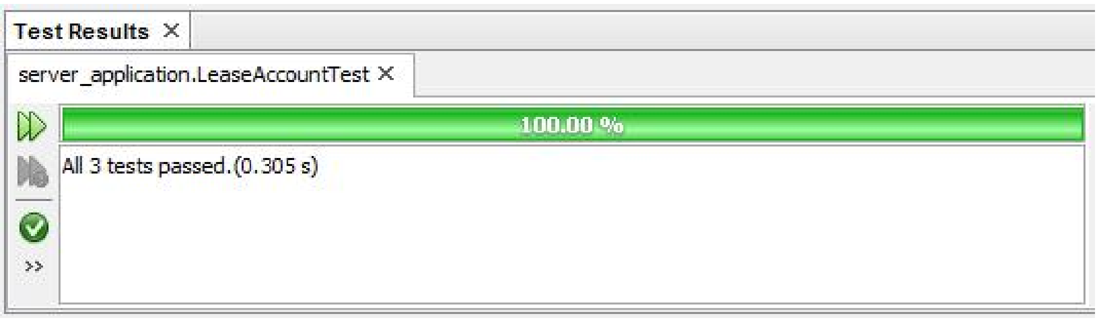

Test Results ✕
server_application.LandlordTest ✕
100.00 %
All 21 tests passed.(0.366 s)

Fig. 14 – Landlord Test Results

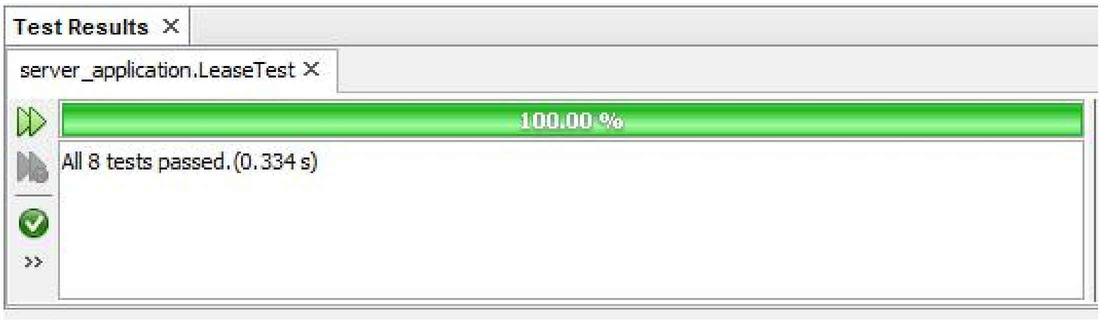
As you can see from Fig. 14, all 21 of the tests within the Landlord Test Class passed, which shows that all of the methods within the Landlord Class are working as they should, updating the state of the landlord object, as it should, and recording all of the modifications as it should. This means that a client of the Landord class is able to create and delete notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

15. LeaseAccount Test


Test Results ✕
server_application.LeaseAccountTest ✕
100.00 %
All 3 tests passed.(0.305 s)

As you can see from Fig. 15, all 3 of the tests within the LeaseAccount Test Class passed, which shows that all of the methods within the LeaseAccount Class are working as they should, updating the state of the lease account object, as it should, and recording all of the modifications as it should. This means that a client of the LeaseAccount class is able to create and delete transactions, notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

16. Lease Test

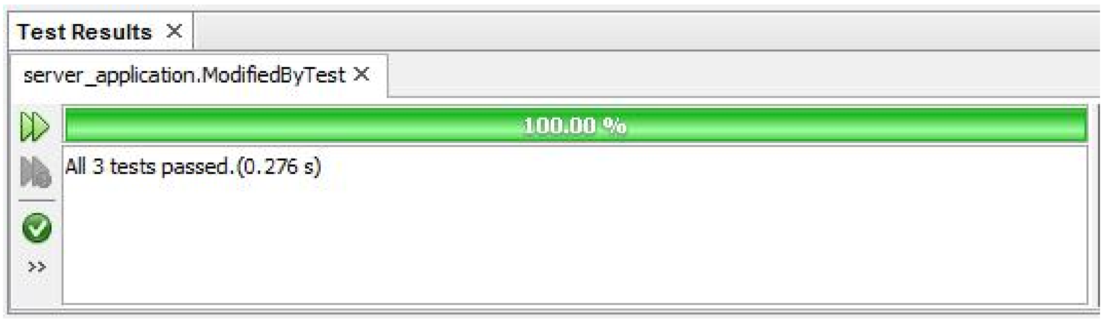


All 8 tests passed.(0.334 s)

Fig. 16 – Lease Test Results

As you can see from Fig. 16, all 8 of the tests within the Lease Test Class passed, which shows that all of the methods within the Lease Class are working as they should, updating the state of the lease object, as it should, and recording all of the modifications as it should. This means that a client of the Lease class is able to create and delete notes, documents, landlords, amend its variables and return the state of the object whilst keeping a record of any modifications made.

17. ModifiedBy Test

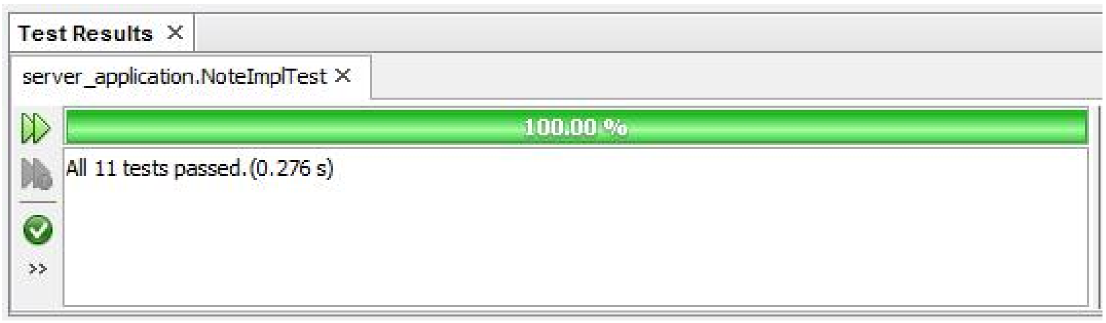


All 3 tests passed.(0.276 s)

Fig. 17 – ModifiedBy Test Results

As you can see from Fig. 17, all 3 of the tests within the ModifiedBy Test Class passed, which shows that all of the methods within the ModfiiedBy Class are working as they should, updating the state of the modified by object, as it should. This means that a client of the ModifiedBy class is able to amend its variables and return the state of the object.

18. NoteImpl Test

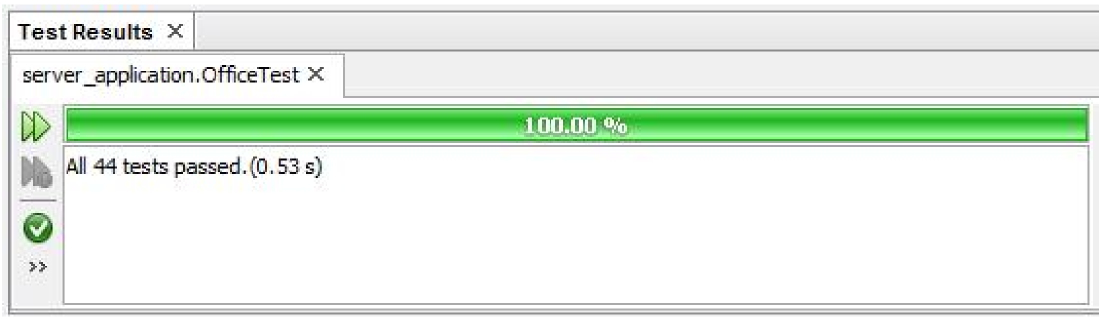


```
All 11 tests passed.(0.276 s)
```

Fig. 18 – Note Test Results

As you can see from Fig. 18, all 11 of the tests within the Note Test Class passed, which shows that all of the methods within the Note Class are working as they should, updating the state of the note object, as it should, and recording all of the modifications as it should. This means that a client of the Object class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

19. Office Test

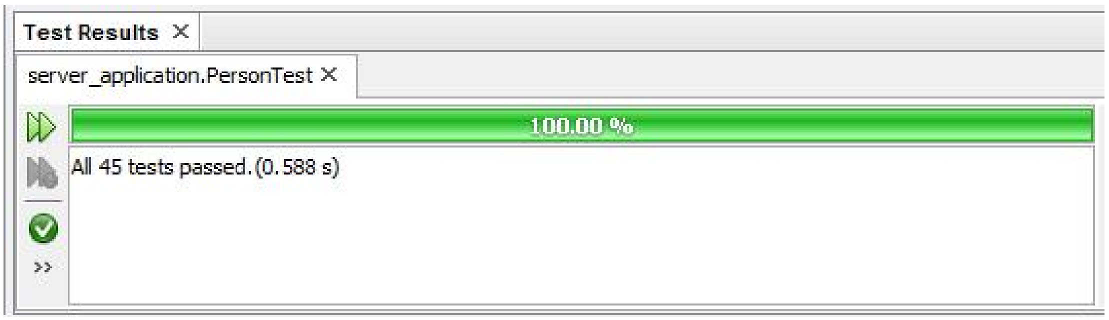


```
All 44 tests passed.(0.53 s)
```

Fig. 19 – Office Test Results

As you can see from Fig. 19, all 44 of the tests within the Office Test Class passed, which shows that all of the methods within the Office Class are working as they should, updating the state of the office object, as it should, and recording all of the modifications as it should. This means that a client of the Office class is able to create and delete notes, contacts, amend its variables and return the state of the object whilst keeping a record of any modifications made.

20. Person Test

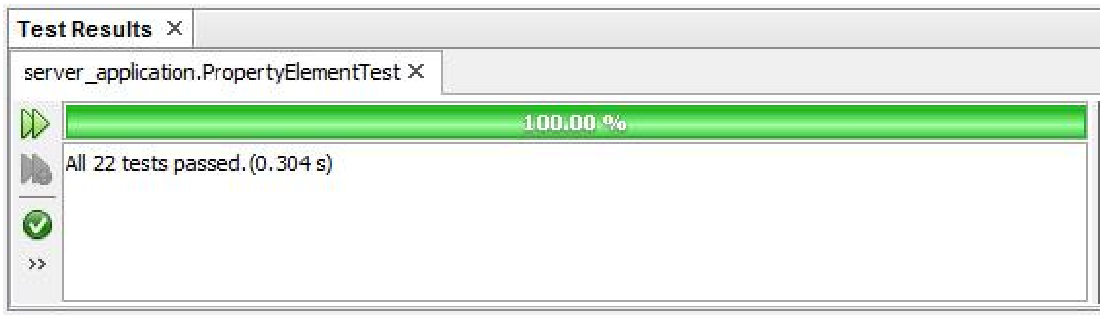


```
All 45 tests passed.(0.588 s)
```

Fig. 20 – Person Test Results

As you can see from Fig. 20, all 45 of the tests within the Person Test Class passed, which shows that all of the methods within the Person Class are working as they should, updating the state of the person object, as it should, and recording all of the modifications as it should. This means that a client of the Person class is able to create and delete notes, documents, contacts, amend its variables and return the state of the object whilst keeping a record of any modifications made.

21. PropertyElement Test

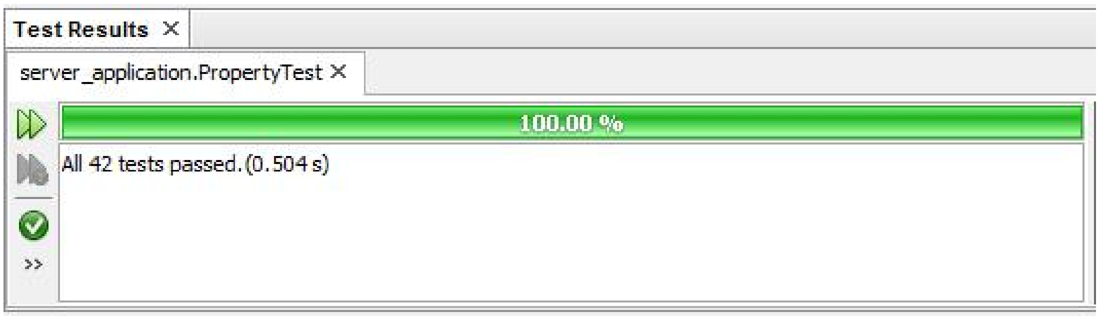


```
All 22 tests passed.(0.304 s)
```

Fig.21 – PropertyElement Test Results

As you can see from Fig. 21, all 22 of the tests within the PropertyElement Test Class passed, which shows that all of the methods within the PropertyElement Class are working as they should, updating the state of the property element object, as it should, and recording all of the modifications as it should. This means that a client of the PropertyElement class is able to amend its variables and return the state of the object whilst keeping a record of any modifications made.

22. Property Test

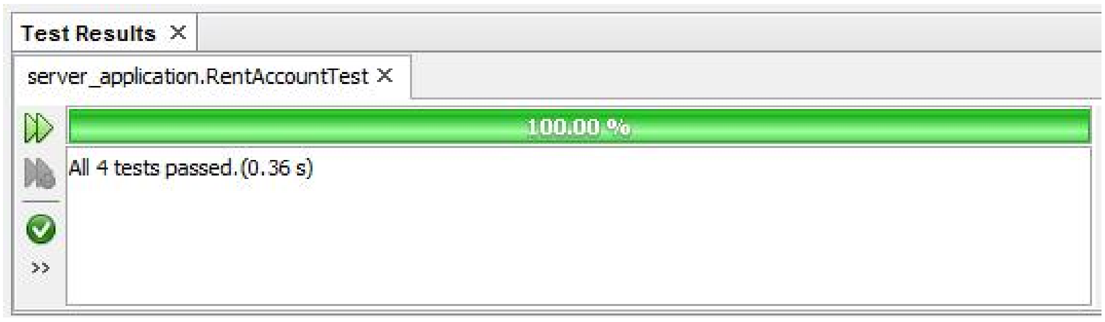


```
All 42 tests passed.(0.504 s)
```

Fig. 22 – Property Test Results

As you can see from Fig. 22, all 42 of the tests within the Property Test Class passed, which shows that all of the methods within the Property Class are working as they should, updating the state of the property object, as it should, and recording all of the modifications as it should. This means that a client of the Property class is able to create documents, notes, assign Landlords, amend its variables and return the state of the object whilst keeping a record of any modifications made.

23. RentAccount Test

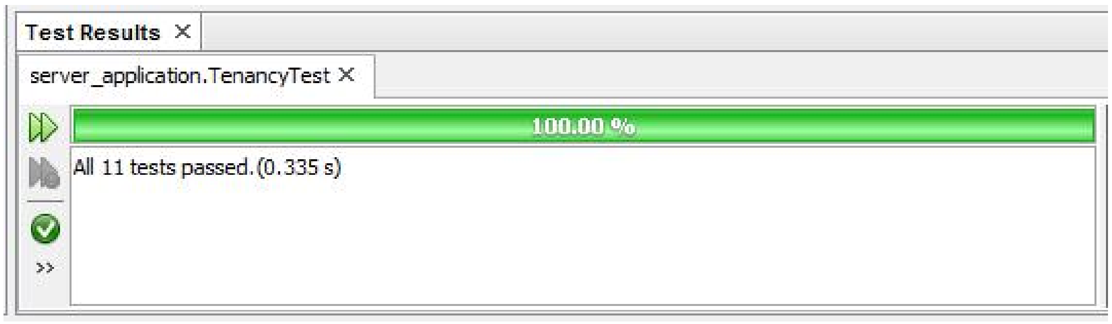


```
All 4 tests passed.(0.36 s)
```

Fig. 23 – Rent Account Test Results

As you can see from Fig. 23, all 4 of the tests within the RentAccount Test Class passed, which shows that all of the methods within the RentAccount Class are working as they should, updating the state of the rent account object, as it should, and recording all of the modifications as it should. This means that a client of the RentAccount class is able to create transactions, notes, and amend its variables and return the state of the object whilst managing the balance of the account when transactions are created or deleted, and keeping a record of any modifications made.

24. Tenancy Test

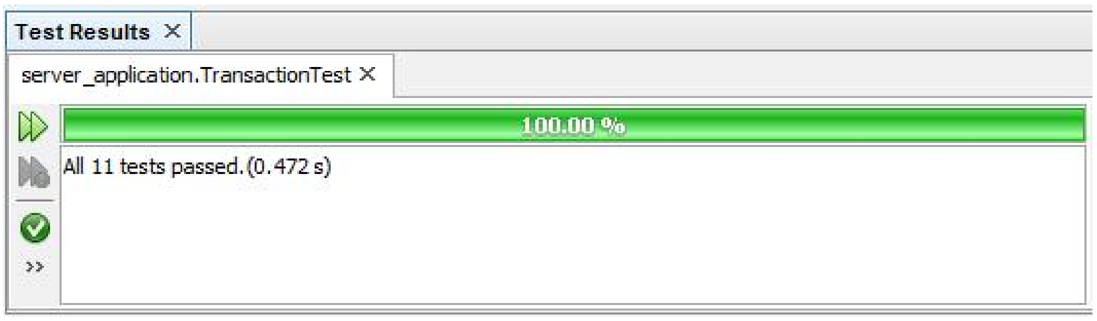


```
All 11 tests passed.(0.335 s)
```

Fig. 24 – Tenancy Test Results

As you can see from Fig. 24, all 11 of the tests within the Tenancy Test Class passed, which shows that all of the methods within the Tenancy Class are working as they should, updating the state of the tenancy object, as it should, and recording all of the modifications as it should. This means that a client of the Tenancy class is able to create and delete notes, amend its variables and return the state of the object whilst keeping a record of any modifications made.

25. Transaction Test

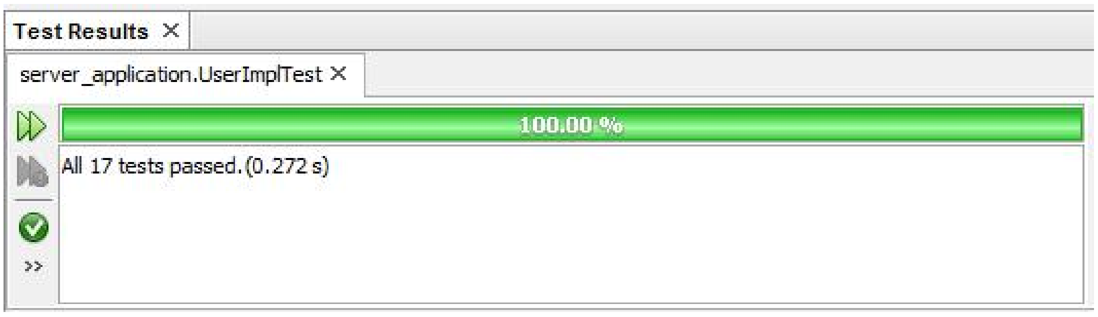


```
All 11 tests passed.(0.472 s)
```

Fig. 25 – Transaction Test Results

As you can see from Fig. 25, all 34 of the tests within the Account Test Class passed, which shows that all of the methods within the Account Class are working as they should, updating the state of the account object, as it should, and recording all of the modifications as it should. This means that a client of the Account class is able to create transactions, notes, and amend its variables and return the state of the object whilst managing the balance of the account when transactions are created or deleted, and keeping a record of any modifications made.

26. User Test



```
All 17 tests passed.(0.272 s)
```

Fig. 26 – User Test Results

As you can see from Fig. 26, all 17 of the tests within the User Test Class passed, which shows that all of the methods within the User Class are working as they should, updating the state of the user object, as it should. This means that a client of the User class is able to update the user permissions, amend the password for the user, amend its variables and return the state of the object.

NB: All of the source code for the unit tests is located in Appendices V along with all of the other source code for this system.