Appendices G

Inheritence

As outlined in Appendices B within the descriptions of the classes to be developed, I designed a super class called AgreementImpl, which hold generic fields and methods for an account, and I have then created TenancyImpl, LeaseImpl and ContractImpl which all extend AgreementImpl and therefore re-use the code of the AgreementImpl class. Furthermore, by developing sub classes that inherit from a super class, it allows the programmer to make use of polymorphisim, which is outlined below.

```
138     /**
139      *
140      * @param agreement
141      * @param modifiedBy
142      * @throws java.rmi.RemoteException
143      */
144     public void createAgreement(AgreementInterface agreement, ModifiedByInterface modifiedBy) throws RemoteException {
145         if(!this.hasAgreement(agreement.getAgreementRef())) {
146             this.agreements.add(agreement);
147             this.modifiedBy(modifiedBy);
148         }
149     }
```

Fig x – Extract from Database, updateUserAgreements() method

As you can see from Fig x, the Office class has made use of polymorphism by having a List of agreements which can hold any type of agreement, whether it be a tenancy, lease or contract and within the createAgreement() method, you can see that the office class invokes Agreement.getAgreementRef() on any of the Agreement elements, again whether it be a tenancy, lease or contract.