Unit Test Specifications

**Assignment 1**

ITC515 – Professional Programming Practice

Justin McKay - Student ID: 11530610

# IBook Tests

## IBOOK1 – Test\_Can\_Borrow\_Book

**Purpose:**To test that the book can be committed to a loan.

**Pre-conditions:**- The book object must be in the AVAILABLE state

**Post-conditions:**- The book state is changed to the ON\_LOAN state and the book now how has a valid LOAN object associated with the IBook.

**Data required:**- A valid IBook object in the AVAILABLE state.

## IBOOK2 – Test\_Can\_Return\_Book

**Purpose:**To ensure that a book that has been borrowed can be returned.

**Pre-conditions:**- The book must be in the ON\_LOAN state.

**Post-conditions:**- The book will be in the AVAILABLE state if the book is returned not damaged or the book will be in the DAMAGED state if the book is returned damaged.  
- The ILoan object will be removed from the IBook object.

**Data required:**- A valid IBook object that is in the ON\_LOAN state.

## IBOOK3 – Test\_Can\_Set\_Book\_As\_Lost

**Purpose:**To ensure that a book that has been borrowed can be can be flagged as lost.

**Pre-conditions:**- The book must be in the ON\_LOAN state.

**Post-conditions:**- The book will be in the LOST state if the book is unable to be returned.

**Data required:**- A valid IBook object that is in the ON\_LOAN state.

## IBOOK4 – Test\_Can\_Repair\_Book

**Purpose:**To ensure that a book that has been damaged can be set to the AVAILABLE state after repair.

**Pre-conditions:**- The book must be in the DAMAGED state.

**Post-conditions:**- The book will be in the AVAILABLE state.

**Data required:**- A valid IBook object that is in the DAMAGED state.

## IBOOK5 – Test\_Can\_Dispose\_Book

**Purpose:**To ensure that a book can be disposed.

**Pre-conditions:**- The book must be in either the AVAILABLE, DAMAGED or LOST state.

**Post-conditions:**- The book will be in the DISPOSED state.

**Data required:**- A valid IBook object that is in the AVAILABLE, DAMAGED or LOST state.

## IBOOK5 – Test\_Can\_Get\_Correct\_Book\_State

**Purpose:**To ensure that the state of the book returned from the *IBook.getState()* method is the current state of the book.

**Pre-conditions:**- The book must be in either the AVAILABLE, ON\_LOAN, DISPOSED, DAMAGED or LOST state.

**Post-conditions:**- The book will be in the original state.

**Data required:**- A valid IBook object that is in one of the AVAILABLE, ON\_LOAN, DISPOSED, DAMAGED or LOST states.

# ILoan Tests

ILOAN1 - Test\_Loan\_CheckOverDue **Purpose:**   
A check to ensure the *ILoan.checkOverDue()* method will flag the loan as overdue if the current date is greater than the loan due date.

**Pre-conditions:**- A staff member must be logged into the system to check for overdue books.  
- The *ILoan* object must be in either the CURRENT or OVERDUE states.

**Post-conditions:  
-** The *ILoan* objects that are overdue should be in the OVERDUE states.

**Data required:   
-** A collection of at least one loan objects in the CURRENT state, with a due date set.  
- A date > than that of the *ILoan* object’s due date.

ILOAN2 - Test\_Loan\_Object\_Fails\_With\_Invalid\_Constructor\_Data  
**Purpose:** A check to ensure the *Loan : ILoan* object throws the correct exception if any of the constructor parameters are null. Also checks to ensure an exception is thrown if *dueDate* is less than *borrowDate* and ensures that *loanId* is not less than or equal to 0.

**Pre-conditions:**- A borrower must have a valid library account.

**Post-conditions:**- A valid *ILoan* object is created with no exceptions.

**Data required:**- An *IBook* object  
- An *IMember*  
- A borrowDate  
- A *dueDate*  
- A valid *loanId*

ILOAN3 - Test\_Can\_Commit\_Loan  
**Purpose:** A check to ensure that the loan can only be changed to the *CURRENT* state if the loan is in the *PENDING* state.

**Pre-conditions:**  
- The loan must be in the PENDING state

**Post-conditions:**- The loan will be in the CURRENT state

**Data required:**

ILOAN4 - Test\_Can\_Complete\_Loan  
**Purpose:** A test to ensure that the loan can only be changed to the *COMPLETE* state if the loan is in the *CURRENT* state and not in the *OVERDUE* state.

**Pre-conditions:**   
- The loan must be in the CURRENT or OVERDUE state

**Post-conditions:**- The loan will be in the COMPLETE state.

**Data required:**

ILOAN5 - Test\_Loan\_Is\_OverDue  
**Purpose:** A test to check the *ILoan : isOverDue* method returns true if the loan is in the OVERDUE state only.

**Pre-conditions:**   
- The loan must be in the OVERDUE state.

**Post-conditions:**

**Data required:**

ILOAN6 - Test\_Can\_Get\_Borrower  
**Purpose:** A test to ensure that the *ILoan : getBorrower* method returns the correct *IBorrower* object.

**Pre-conditions:**

**Post-conditions:**

**Data required:**- A valid IBorrower object.

ILOAN7 - Test\_Can\_Get\_Book  
**Purpose:** A test to ensure that the *ILoan : getBook* method returns the correct *IBook* object.

**Pre-conditions:**

**Post-conditions:**

**Data required:**- A valid IBorrower object.

# IBookDAO Tests

IBOOKDAO1 – Test\_Can\_Add\_Book\_To\_Collection  
**Purpose:** A test to ensure that an IBook object can be added to the IBookDAO object.

**Pre-conditions:  
-** The author, title and callNumber must be known.

**Post-conditions:**- A new IBOOK object is returned with a valid id.

**Data required:**- The author, title and callNumber values.

IBOOKDAO2 – Test\_Can\_Get\_Book\_By\_Id

**Purpose:** A test to ensure that an IBook can be returned from the IBookDAO object with a specific known Id.

**Pre-conditions:**- The IBookDAO object must contain at least one book in the collection and the ID of that book must be known.

**Post-conditions:**- The IBook will be returned from the method.

**Data required:**- A valid IBook object with a known Id value.

IBOOKDAO3 – Test\_Can\_List\_Books\_In\_IBookDAO\_Collection  
**Purpose:** A test to ensure that the *IBookDAO* object can return a list of the currently stored books.

**Pre-conditions:**- A valid IBookDAO object exists with at least one IBook stored in the object

**Post-conditions:**- A collection of IBook objects will be returned from the IBookDAO class.

**Data required:**- A valid IBookDAO object with at least one IBook object.

IBOOKDAO4 – Test\_Can\_Get\_Books\_By\_Author  
**Purpose:** A test to ensure that the *IBookDAO* object will only return the books in the collection that match the author.

**Pre-conditions:**- A valid IBookDAO object containing multiple IBook objects with differing authors.

**Post-conditions:**- A collection of IBook objects that only match the author.

**Data required:**- A valid IBookDAO object populated with multiple IBook objects with differing author properties.

IBOOKDAO5 – Test\_Can\_Get\_Books\_By\_Title  
**Purpose:** A test to ensure that the *IBookDAO* object will only return the books in the collection that match the book title.

**Pre-conditions:**- A valid IBookDAO object containing multiple IBook objects with differing titles.

**Post-conditions:**- A collection of IBook objects that only match the title specified.

**Data required:**- A valid IBookDAO object populated with multiple IBook objects with differing title properties.

IBOOKDAO6 – Test\_Can\_Get\_Books\_By\_Author\_And\_Title  
**Purpose:** A test to ensure that the *IBookDAO* object will only return the books in the collection that match both the author and the title.

**Pre-conditions:**- A valid IBookDAO object containing multiple IBook objects with differing author and title values.

**Post-conditions:**- A collection of IBook objects that only match the specified author and title.

**Data required:**- A valid IBookDAO object populated with multiple IBook objects with differing author and title properties.

# ILoanDAO Tests

ILOANDAO1 – Test\_Can\_Create\_Pending\_List  
**Purpose:** A test to ensure that an *ILoanDAO* can create a new pending loan list for the *IMember*.

**Pre-conditions:  
-** The *IMember* object must exist.

**Post-conditions:**- A new pending list is created in the *ILoanDAO* collection.

**Data required:**- An *IMember* object.

ILOANDAO2 – Test\_Can\_Create\_Pending\_Loan  
**Purpose:** A test to ensure that an *ILoan* object can be created and to the *ILoanDAO* object based on the *IMember* and *IBook* objects.

**Pre-conditions:  
-** The *ILoanDAO* object must have the pending loan list for the specified member. **-** The *IMember* object must exist.  
- The *IBook* object must exist.

**Post-conditions:**- A new *ILoan* object is returned with a valid id and in the PENDING state.

**Data required:**- The *IBook* and *IMember* objects.  
- A borrow date and a due date.