UNIVERSITY OF BOLTON
FACULTY OF ARTS AND MEDIA TECHNOLOGIES
COURSEWORK SUBMISSION FORM

Student/Centre to complete:

SURNAME/FAMILY NAME: **Plumpton** FORENAMES: **Thomas**

BOLTON STUDENT ID: **1500936** EMAIL: **Thomas.Plumpton@hotmail.co.uk**
TP2AMT@Bolton.ac.uk

DATE OF SUBMISSION: ……………………………………………………………………...……..

MODULE NO./TITLE:…**CPU4003 Introduction to Programming**……………………

TUTOR'S NAME: ………………~~A. Razak~~, ~~B. Tighe~~, ***A. Parker***…………………

COURSEWORK TITLE: …… **Clap Hill Library System** ……….

Please state if this is your FIRST submission OR REFERRED/DEFERRED submission OR a REPEAT submission? **FIRST**

……………..………………………………………………………………………………………

Declaration:
I hereby declare that this work is my own work. I understand that if I am suspected of plagiarism or another form of cheating, my work be referred to Academic Registrar and/or the Board of Examiners, which may result in me being expelled from the programme. I understand once I submit this work, it will automatically belong to the University of Bolton.

Academic staff to complete:

Feedback: ……………………………………………………………………………………

……………………………………………………………………………......................

………………………………………………………………………………………………

…………………………………………………………………………………………......

Date Issued …Week 6 (31 October 2014). Hand-In Date:……17:30 16th January 2015..

Other Relevant Date e.g. Demonstration ……………………………………………..

Received: On Time ☐ Late ☐ (within 5 days of published deadline date)

Mark awarded: ………..% Do not apply mark penalty unless the work was submitted late.

Assessors Name: ………………….…… Signature:...

Date:…………………………

Degree Conversions	A: 70-100%	B: 60-69%	C: 50-59%	D: 40-49%	F: 0-39%
HND Conversions	Pass: 40-49%	Merit: 50-66%	Distinction: 67-100%		

Late submission details can be found on the student handbook http://www.bolton.ac.uk/Students

Introduction to Programming

Assignment weighting

This assignment is worth 70% of the overall mark for this module.

Learning Objectives

The following learning objectives are covered by this assignment

	Learning outcome	**Assessment criteria**
1	have understood and learnt the programming language syntax and constructs	can create and/or recognise syntactically correct code for specified aspects of the programming language syntax and constructs
2	write simple code to demonstrate appropriate use of differing aspects of syntax	writes code for a precisely specified outcome
3	design and program a solution to a given problem	designs, implements and tests a program to solve a given problem

When reading this assignment brief DON'T PANIC it has been written so that it provides a challenge for the strongest programming students but also enables other students to demonstrate they can create simple Java programs.
In this assignment you are going to create an application that can be used in a lending library.

Scenario

The Clap Hill Lending Library

The village of Clap Hill has had its own library for 25 years but it is now under threat of closure due to spending cuts. The only hope is to improve the library's efficiency and to do this they have asked you to create a computer program to keep track of its members and books.
Using your knowledge of object orientation you have identified a number of entities that need to be modelled by your software, they are:

Member

The library keeps basic information about each of its members e.g. first name, last name and phone number, also each member is given a unique membership number (this is a positive integer).

Book

The library gives each book a unique identity number (this is a positive integer) which is used to track loans. The library also records a book's author, title and whether it is fiction or non-fiction.

Loan

A loan is a member borrowing a particular book, so if a member comes into the library and borrows three books then this is considered to be three loans. The library records the membership ID number, the book ID number, the date loaned and the date due back (the loan period for all books is 3 weeks)

Library

This entity provides all the functionality required to maintain collections of members, books and loans. The expected functionality is shown on the next page.

Library system user requirements (functionality / methods)

User Requirements Set A	**Max mark**
1. Membership records: a. add a new member b. for a given member ID number print the member's details c. print a list all members 2. Book Records a. add a new book b. for a given book ID number print the book details c. print a list of all books	49%
User Requirements Set B	
3. Membership records: a. return a member with a particular member ID number b. search for and print all members where their surname contains a particular search string c. remove a member with a particular member ID number 4. Book Records: a. return a book with a particular book ID number b. search for and print all books whose title contains a particular search string c. remove a book with a particular book ID number	59%
User Requirements Set C	
5. Loan records (For this set there is no need to record date loaned etc.) a. add a loan for a given member and book ID numbers b. remove a loan for a given member and book ID numbers c. list all loans giving book and member details	69%
User Requirements Set D	
6. Loan records (For this you will need to work with dates) a. print a list of overdue loans giving book and member details.	99%

Some guidelines to creating the software

Step 1
You are obviously going to need some classes to model the member and the book so you should create these first. For each class:

- identify and define the fields required, these can be found in the scenario;
- create accessor and mutator methods for each field;
- create a parameter constructor to simplify the creation of member and book objects;
- create a method that prints the objects details on one line, this will be useful when you want to print a list of members or books.

Step 2
Once you have created these classes you can then create the library class which will be used to maintain collections of members and books.

- Define two fields of type ArrayList, one for the members and one for the books;
- create a constructor that will create the ArrayLists;
- create methods to implement the functionality in 1 and 2 – each user requirement would be a separate method in this class.

Step 3
The final part is implementing the loans functionality. Here I suggest you create a loan class that has at least two attributes which hold member ID and book ID (or member and book object references). To implement the final part using dates I recommend that you investigate the Java library class GregorianCalendar.

What you are required to do

There are three parts to the assignment that <u>must be attempted</u>.

1) Program implementation
   a) You must create a program that provides some, or all of the functionality specified in the ***Library system user requirements*** provided on page 3. The maximum mark available depends on how much of the functionality you can successfully produce.
2) Class diagram
   a) use the Violet UML editor software provided to create a class diagram which accurately describes the implementation.
3) Javadoc documentation
   a) use the documentation tool in BlueJ to produce a set of web pages that accurately describes the program implementation.

This part is optional but students can gain extra credit by creating one or more test classes.

4) Testing
   a) create an additional class or classes for the purpose of testing the library functionality;
   b) this class(es) will automatically create library, member and book objects;
   c) this class will have methods that can be executed from the BlueJ interface to test aspects of the implementation e.g. adding a book, removing a member etc.
   d) Javadoc documentation will be provided for the class(es)

Marking

The mark awarded will be based on the level of functionality provided ***and*** the quality of the program and its documentation. For instance a submission of poor quality may get a lower mark than a program with less functionality but is of higher quality.

Referencing

If you have used a piece of code that you have not written or used ideas from online forums, please ensure that you properly cite the reference from where you obtained the code. This will prevent you being accused of plagiarism. Also please remember that we will be assessing your ability to develop and write the programme.

Assignment submission

Your assignment must be submitted by the date shown on the front page of this assignment brief with the standard submission form available in student services or outside room B1-05.

You must submit

1) A zip folder to the Moodle Submission link clearly identified with your name and student ID number.

In the Zip folder	**Check**
Your class diagram for the library system created with Violet UML	
If you have implemented any additional functionality; a single page Word document describing the extra features.	
Your completed BlueJ project including the Javadoc documentation	

2) With your Zip folder - this assignment brief (in its entirety) signed by you and dated on the front page. Also you must identify the functionality you have implemented on the feedback sheet.

Failure to submit all these **will** result in a significant loss of marks.

Assignment 1 – Marking Scheme and Student Feedback

Student ID number & Name	Tom Plumpton, 1500936.

FUNCTIONALITY PROVIDED					Max Mark
User Requirements Set A	1	a	b	c	49%
	2	a	b	c	
User Requirements Set B	3	a	b	c	59%
	4	a	b	c	
User Requirements Set C	5	a	b	c	69%
User Requirements Set D	6	a			99%

Circle the letters showing what functionality you have implemented.

PROGRAMMING CODE	BS	WA	A	G
Appropriate user defined classes provided				
Classes have appropriate fields				
Classes have appropriate constructors				
Classes have appropriate accessor & mutator methods				
Classes have other methods as required				
Java naming convention used for classes, fields and methods				
Java naming convention used for parameters and variables				
Use of descriptive variable names				
Appropriate validation and error messages				
Overall structure and logic of code				
Overall quality of code				

JAVADOC DOCUMENTATION	BS	WA	A	G
All classes documented				
All public fields and methods documented				
All parameters and return values documented				
Documentation clearly states the purpose of the class/method				
Documentation clearly explains how to use the methods				

CLASS DIAGRAM	BS	WA	A	G
All classes included				
All public fields and methods included				
All private fields and methods included				
Correct relationships between classes				

BS – Below acceptable standard : WA – Weak but acceptable : A – Acceptable : G – Good