NOTTINGHAM
TRENT UNIVERSITY

School of Science and Technology

COURSEWORK ASSESSMENT COMPONENT Part 2 of 2

MODULE CODE	:	SOFT20091
MODULE TITLE	:	Software Design and Implementation
MODULE LEADER	:	**Richard Hibberd**
TUTOR(S)	:	Giovanna Martinez-Arellano Patrick Merrit
COMPONENT	:	2 of 2 (for Element 1)
TITLE	:	Design and Development of a Film Database
LEARNING OUTCOMES ASSESSED	:	All
WEIGHTING	:	50% of the overall module mark (+ 50% from CW1)
DISTRIBUTION DATE	:	14th March 2016
SUBMISSION DATE	:	22th April 2016 (Demo/*viva* in labs weeks of 25-29 April and 2-6 of May 2016)
SUBMISSION METHOD	:	Code submission & cover sheet to module dropbox; printed (paper) report brought to demo/*viva*.
NOTE	:	The usual University penalties apply for late submission and plagiarism. Please consult your student handbook for further details.

I. Assessment Requirements

Given the problem defined in Section II produce a C++ solution and provide a report of between 5 and 10 pages with full code appendices.

The software will be submitted electronically (Visual Studio solution) to the module dropbox by the required submission date, together with a completed summary and ownership form. A physical printed copy of the report should be brought to the demo/viva.

A viva will be performed in the 2 weeks after the submission during which you will be required to demonstrate and explain your code and potentially to make changes to it to demonstrate your ownership of the code.

Section II refers to the assessment problem, Section III refers to the assessment criteria and Section IV refers to the feedback opportunities.

II. Assessment Scenario/Problem

Scenario:
You are a newly recruited analyst programmer, working for TrentCorp, a major IT company. This company won a contract to develop a film project management system for TrekStar Pictures. It has to be designed using object oriented techniques and coded in C++. This system has to manage information about film projects, such as "E.T." and the "Fast and the Furious" series. You can assume that the data will be sufficiently small to be memory-resident during the daily operation of the system. This exercise does not allow for the development of a database application; you must manage the files yourself.
From a second meeting with a TrekStar Pictures representative, it has been clarified that in addition to the project details and associated materials of the original brief, the system should handle the following information and business rules:

- Projects that are currently under production are considered "unreleased" projects (these wouldn't have any associated materials). The system **should not allow** the user to input any theatrical weekly box office information nor the addition of materials. Projects related to films that are currently showing on cinemas are identified as "now playing". The system should allow the user to input theatrical weekly box office information but should not allow at this point the addition of materials. Projects related to films that are no longer at cinemas are identified as "released". Only these type of projects will allow the addition of material information.

- Each project has a crew. This crew consists of all the people involved in the production of the film such as the producer, director, writer, cast (set of actors), editor, production designer, set decorator and costume designer.

Feel free to email the TrentCorp principals (Giovanna Martinez-Arellano, Patrick Merrit and Richard Hibberd) to address and resolve any questions you have.

In addition to the start-up (load data) & shut-down of the system (store data persistently), at least three functionalities must be developed:

Original Brief

*Every time the production of a new film begins, a new project is created. This **project** has all the relevant information regarding the production of the film. Each new project has a title, summary, genre, release date, list of filming locations, language, runtime, keywords and a weekly ticket sale (theatrical weekly box office).*

*TrekStar pictures launches together with the film, a series of **materials** for retail. These comprise single-sided DVDs, double-sided DVDs, combo box sets (containing two or three DVDs), VHS (on old projects) and Blu-rays. All of them contain features such as identification number, film title, format, audio format (Dolby, Dolby digital, MPEG-1, PCM or DTS), run time, language, retail price, subtitles, frame aspect (wide screen). However, each of them has different packaging specifications. A single or double DVD will have a plastic box packaging, while a combo box set would have a cardboard box. The Blu-ray would also have a plastic box as the DVD, but with a different size. A VHS can come in a plastic or cardboard package.*

DVDs and Blu-ray can have multiple language tracks and subtitles in different languages, compared to VHS that can only have one subtitle and one audio track. In addition, DVDs and Blu-rays may contain bonus features (additional material such as short films or director comments of the production). For double-sided DVDs, it is important to know what contents are in one side, and what in the other (chapters, bonus features and languages).

*Model this as an inheritance hierarchy with a **Material** base class and implement a C++ version. Using your classes, code functionality to:*

a. Read the details of a collection of materials related to a project from a text file (e.g. csv file) and create an object for each material.
b. Add new materials to a database, persistently (added to the file!).

Film Project Management System Requirements:

The following terms are identified in the document:

May	- An optional feature of the system
Should	- A desirable statement about the system
Shall	- A recommended statement about the system
Must	- A mandatory statement about the system

A need has been identified to develop a system which **must** represent the current details of projects at TrekStar Pictures.

The system **must** be capable of representing a variety of materials for each project and must differentiate between the different types of materials; the variety need not be exhaustive but will allow representative operations on the database. All materials of a project should be managed in the same container using base class pointers.

The system **must** present the user with the following functionality:

- From a cold start, existing project and material details **must** be loaded from file.
- Project and Material Creation function – this **must** allow the addition of new projects and materials to the database (file) with the following rules:
  - The system **must** not allow to add new materials to projects that are "unreleased" or "now playing".
  - Only for those projects being created with a "now playing" status, the system **should** allow to add a set of weekly box office figures during creation.
  - The system **should** allow the creation of more than one project with the same name.
- Existing Projects and Materials Update/Removal – the system must allow for any changes in the existing projects and materials following the business rules stated previously. The deletion of an existing project should remove all the material information associated with it.
- Catalogue browsing – **must** allow bi-directional sequential browsing of all projects; **shall** allow interactive project and materials search with the following rules:
  - A search by project title will allow the user to see main project detail information as well as summarised information of materials (e.g. "is currently available on DVD and VHS").
  - Provide the user with the option to view the materials details when searching by project title. When displaying double-sided DVDs, the view should show details of the contents per side. The same with combo box sets, the view should display details of every DVD in the combo box (single and double-sided DVDs).
  - A search by actor should return all project titles where that actor was part of the cast.

  The system **may** allow more flexible search with multiple fields. For example, search only for all the double-sided DVDs available of a specific project title.
- Maintenance Mode – a number of Utility functions which can either be integrated or run as command line applications, to:
  - Add/remove projects and materials to the database.
  - Raise daily reports on (**may reuse Logger class**)
    - New projects/materials added
    - Projects with certain amount of total box office earnings

Each submission will be assessed across the 3 strands (S1-S3) below.

S1. Data modelling and serialisation
S2. Object management
S3. Data manipulation, control and view.

S1. Data modelling and serialisation

You should produce a data hierarchy for the materials with appropriate data fields for the system. You do not need to fully represent any particular instance, but you need to be able to differentiate between the core material types identified in the specification. All data (project details, materials and crew) must be capable of being written to disk and retrieved as needed.

S2. Object management

The objects should be managed through a projects & materials 'data base' type structure which can be iterated through and which can be searched. You can use your existing (SeminarB) custom container, implement new ones or use one or more of the standard library collection classes. Each approach has advantages and drawbacks and you may want to discuss these with your lab tutor. Better solutions will use dynamic binding to organise function dispatch, rather than typing the objects and switches.

S3. Data manipulation, control and view

The new system needs to be implemented using an MVC design pattern. You should apply the OO design principles to organise your classes with extensibility and maintainability in mind. Make every effort to de-couple the presentation of your data from the control (main workflow of your system) and the data manipulation. For each of the functionalities that need to be developed, make sure you make a sensible decision as to which class should encapsulate such behaviour (e.g. who should encapsulate the creation of materials, who should encapsulate the search, and so on).

III. Assessment Criteria

The report and code submission will constitute 100% of the assessment mark. Successful completion of the viva is a required part of the coursework and no mark can be assigned without this section, though it is considered to be zero weighted, it can act to increase or decrease the indicated grade based on ownership of the code. You must identify any code that is not entirely yours (downloaded or developed in a group) and indicate your contribution, to allow the correct allocation of credit for the work.

	First	2:1	2:2	3	Fail
	Ex | H | M | L	H | M | L	H | M | L	H | M | L	Marg | M | L | 0
	Two components improved to a higher level, and one to an awesome level.	One component improved to a higher level, other two at good level.	All components to a good level.	All components to a minimal level.	At least one of S1, S2, S3 not implemented.
S1	As 2:2 with some additional characteristics. Typically serialisation to a non-flat file data structure. Possibly with intermediate aggregation of some attributes[1].		As 3rd, however, improvement to hierarchy or data serialisation to demonstrate a greater understanding and design process.	Minimal data hierarchy is implemented allowing differentiation between core classes of material with defined attributes.	Component not present or poorly presented such that functionality is compromised.
			Serialised to a plain text file such as CSV[2].		
S2	Complex containers, with robust handling. Issues of object lifetime and mgt understood and handled; standard containers may use smart pointers		Appropriate containers, appropriately manipulated	Uses a static container or one container per type of material.	Component not present or poorly presented such that functionality is compromised.
S3	Interface is strongly separated from data, Unused fields should not be displayed on the screen	Strong separation of interface, control logic and data model.	Some separation between data model, control and display logic. Use of some kind of factory pattern for creation of materials.	At least one layer separated from the rest of the system.	Component not present or poorly presented such that functionality is compromised.
	Some other design patterns might have been considered/applied.				
Functionality	Fully- featured solution, with evidence of insightful design	Solid implementation with most functionality present	Core functionality, and some search capability.	Basic functionality only	Component not present or poorly presented such that functionality is compromised.
	Typically uses dynamic binding Virtual functions/RTTI		Typically explicitly labels objects with type information		
Code Quality and Clarity	Code is of high quality and consistently follows a structure. Code can be read quickly and accurately.	Code is of good quality and highly functional. Code is well structured and clear with a good commenting style	Code is functional. Code is readable with little effort and SDI coding standards followed in the main.	Code is functional, however with minor errors. Code is readable with some effort.	Code is badly written and / or commented and does not represent a clear and well thought out approach.
	Style Guide compliant				

[1] This is quite possible using CSV as the file type – aggregate the 'flat' data for a better interface to the object constructors/serialisers.

[2] 2:2 solutions tend to use CSV; XML requires more thought & design, so *usually* is a feature of better design. Both can be used well or badly & all points in between.

Report	High quality report giving useful information for an analyst taking the prototype to final production. Good judgement of what to include/exclude	As 2:2 with explanations of advanced functionality and comment on code design and quality.	As 3rd with more depth on structure and Interface design	Report provides justification for basic design decisions, provides class diagrams.	Report missing or insufficient information to understand the reasoning behind design decisions.

Grade-based Assessment Scheme

Each assessment element is awarded a grade in line with the descriptions below. The final grade is determined by how well the criteria have been met overall and not the sum of the individual aspects of the work.

IV. Feedback Opportunities

Formative (Whilst you're working on the coursework)

Support and guidance will be available in the labs, by email or by appointment

Summative (After you've submitted the coursework)

You will receive specific feedback regarding your coursework submission together with, in most cases, your awarded mark/grade initially at the demo.