SOFT20091 - Software Design & Implementation 2015/16 cw2 cover sheet

Name		I confirm that the source code and report are my own work; any other contributors are explicitly acknowledged in the documentation, in line with the University Regulations Signed to be signed at the demo
Group		
NTU ID		

Function	Y/N	Choose/add functionality implemented + brief description	Notes / grade (tutor)
Cold start read files & create objects		Complete only the shaded areas where they are appropriate. You should delete any red text – it's there for your guidance. You can extend this table up to a single A4 page	
Browse the data		Browse / Search (single/fixed/multiple fields) / and/or query Delete the ones it can't do. Add detail to clarify the functionality.	
update weekly box office data		Interactive / batch (file of transaction data) You can delete any unused rows of this table Don't change the width of the columns	
'Admin' functionality		add / delete Projects / materials Reports (which) Other	

Underlying solution	Choose/add + brief description Red is suggestion	Notes / grade (tutor)
Files	Name # & format / R/W by app / library?	
Object model	Inheritance used (+ derived class extras) Virtual functions (list them) Collection object	
Containers	Array / map / vector / other (own/std) Element type Collection object	
Search	Describe scope & limits of search	
Other	Such as logging / libraries or frameworks used / GUI	

SOFT20091 - Software Design & Implementation 2015/16 cw2 cover sheet

	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 and data structures.	Some separation between data structures 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				
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.

[1] Thi 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.