**Design and Analysis of Data Structures and Algorithms 2014 – 15**

**Assignment Specification**

**Assignment Background and Context**

An online business maintains a large collection of antiques and collectables stored in various repositories around the world. The business sells these items through its online store and also through a variety of online auction houses. The business is facing an operational problem that sometimes an item might be sold more than once to different customers as the information is relayed to all the other online facilities by email and administrators of the various action houses have to search for the sold item and remove it from their lists manually. The ensuing delays might lead to a duplicate sale and thus might compromise the integrity and the reputation of the company as it’ll be exposed to negative criticism and possible law suits. Furthermore the company’s partner action houses have started complaining and have threatened to withdraw from the partnerships if such issues and complaints continue as such situations undermine their reputation too.

To resolve such conflicts the company has tasked you to design and develop appropriate software that utilises suitable data structures and relevant algorithms that would process the data to complete the following tasks.

* New items can be added to a repository and removed from it at any point in time
  + The company would like to keep the placing of items in a repository as transparent as possible to the users, but expects that maximum efficiency in searching and finding an item can be achieved when there is a need to remove it.
  + There might be a need to place the items in a specific order in the repository and this is a requirement that the company is particularly keen on.
  + The same item can be present in many repositories and can form part of many sets of items – e.g. a coin can be part of a set of coins or can be offered elsewhere as an individual coin; likewise a piece of furniture or a piece of clothing.
* Items will need to be deleted from all repositories once they are sold regardless of where they are sold at. If an item is part of a set it should be removed from the set.
* All items are individually identified – even if they are of the same type, e.g. where multiples copies of the same coin exist.
* If an item is removed from a set an item of the same type should be sought through the various repositories to complement the set. You can assume that the physical location of the item is irrelevant in this case.

**Tasks to be completed**

1. Identify and justify the choice of data structures to be used to implement the various data repositories – not more than 500 words.
2. Design the software and provide an efficient algorithm that would support the above requirements. Write relevant pseudo code where appropriate.
3. Implement the system with the following functionality.
   1. Add an item to a repository
   2. Add an item to a set
   3. Link the different instances of the item in the different repositories
   4. Search for an item across different repositories
   5. Remove an item from all repositories
   6. Remove an item from all sets across all repositories
4. Notify all repositories that a specific item has been removed from their repository and /or the item has been removed from a specific set – on screen.
5. Complete appropriate test data sets to thoroughly test your software and use them to test your system.

PLEASE NOTE

* A GUI is not an essential requirement for the work to be submitted and it is not reflected in the mark scheme. Some students though might find it beneficial to have a GUI to demonstrate their work.
* There will be a minimum of THREE different repositories
* There will be a minimum of FIVE sets of items of no less than six items each and at least FOURTY different items in total with at least THREE duplicate items across all repositories.

**Resources provided**

1. Partially complete test data set that would help you explore different conditions that your system should test.

**Deliverables and due date**

**By the due date (26 March 2015) you should submit via Blackboard**

* A zip file, containing the NetBeans project that implements the application. The project should include the test data files that you would have used.
* A word file that contains your choice and justification of data structures and search algorithms used and the completed pseudo code.

**Marking Criteria**

For the submitted work (Code and documentation 65 marks)

0- 21 Little or no attempt at the assignment has been made OR few of the implemented classes compile.

22-26 Some of the implementation compiles and is correct. The student can account in writing for what would be required to complete the implementation

27-35 From the list of tasks given above, 1 – 3 are correct and at least a reasonable attempt at 4 and 5 has been made or can be accounted for. The implementation compiles and runs. The student is aware of what the problems and may be able to begin to suggest ways of correcting them. The submitted documentation is more or less complete.

36-45 A plausible attempt at meeting most of the requirements of the assignment has been made. The implementation compiles and runs and produces, or mostly produces the expected results. The student is able to evaluate the implementation, highlighting what the problems are and suggesting ways of correcting them. The submitted documentation is accurate and more or less complete.

46-55 The requirements of the assignment have been met. The code runs and produces the expected result. In writing, the student is able to evaluate what has been done and discuss what the alternatives/limitations are.

55-65 The requirements of the assignment have been met as above. In writing, the student is able to explain what has been done and discuss what the alternatives/limitations are. The whole is well commented, well presented, well documented. Input is error checked, output is helpful and complete. Overall, the work is of a professional standard.

Demonstration (35 marks)

0 Non-attendance.

0 – 10 The student is able to run the code but unable to explain it, even when prompted. OR, the application is unchanged from its original state.

11 – 17 The student is hesitant about the code but able to explain it with prompting.

18 – 21 The student can explain the code as written but is unable to discuss alternatives or suggest where errors might have occurred

22 - 25 The student can explain the code as written and can discuss alternative solutions or suggest where errors might have occurred

26 - 35 The students can fluently explain the code as written. Alternatives can be discussed. Where there are failures of functionality, the student can suggest why these have occurred and explore solutions.

**Working Arrangements**

This is an individual piece of work. You are expected to use Java to implement your answer. You should expect to spend approximately 24 hours completing this work.

**Plagiarism**

Whilst it is acceptable to discuss your assignment with your peers, this piece of work is intended as an individual assignment. Submissions that show similarities to other work will be subject to investigation according to University regulations