Your organization needs an Examination system.

The Examination Management System is a desktop application designed to automate the exam process for organizations. The system allows users to record and manage questions and exams and answers in xml files, validate it against a predefined schema using XML Schema (XSD), and generate reports through XSLT transformations.

Design a Class to represent the Question Object, Question is constructed from a Body, Marks, and Header and ……..

We want the application to accept different Question Types, True or False, Choose One and Choose All each has a different way off representation.

We need to define a Base Question class and every type as an inherited one.

**Design a class to represent the Question list by inheriting the List<> class**

**Override the Add Method, keep the default behavior for the Add Method and add logic to open a file and Log the Questions in it as xml file, every Question Object of Question List will be logged to a Same file. each Question List has Different File**

We need to define a class for the answers and also the Answer List.

Question Object is associated with an Object of AnswerList

Design a Base class: Exam, exam class describe the common attributes concerning the exam, Time, number of Questions, **Question Answer Dictionary(Which will be used for Exam Correction)** , a Show Exam Functionality that it’s implementations will be differed to the further classes in the hierarchy

Every Exam object is Associated to a Subject Object (implement any desired Subject class members)

We have two types of Exams , Practice Exam and Final Exam , Practice exam shows the right answer after finishing taking the Exam , while the Final Exam Only Shows The Question and Answers .

**You want to consider what type of constraints you need to add to this Generic class**

In the Main declare two objects one of practice exam and one final Exam

We need the end user to select the Exam Type , and upon this choice we will show the Exam .

Implement Iclonable , IComparable , consider overriding ToString , **Equals , GetHashCode** all the constructors use chaining .

**Every exam has a mode : Starting , Queued , Finished**

**When the exam in Starting Mode , Every Student taking this subject should be notified ( implement the desired class hierarchy and implement the required evens and delegate to produce this functionality )**

**Key Features:**

**User Types:**

Students =>login , takes exam , practice or final exam.

Teacher => login , add questions , answers

Admin => add students and teachers

**User Authentication:**

* Users must log in with valid credentials to access the system.
* Different user roles (Admin, Teacher, Student) with corresponding permissions.

**Question Recording:**

* Teachers can add questions and saved to xml file
* Students can answer exam and degrees are added to xml file with student name

**XML Data Storage:**

* question data is stored in XML format.
* Student answers are stored in xml file

**XML Schema Validation:**

* Define and load an XML Schema (XSD) for validating the question data.
* The system ensures that recorded question adheres to the specified schema.

**Attendance Reporting:**

* Generate result reports for individual students
* Reports can be viewed within the application or exported to external formats (PDF, Excel).

**User Interface (UI):**

* Clear and intuitive UI for ease of use.
* Separate interfaces for Admin, Teacher, and Student with role-specific functionalities.

**Error Handling:**

* Implement robust error handling for scenarios such as invalid login credentials, data input errors, etc.
* Provide informative error messages to guide users.

**File Management:**

* Save and load question and exam data from XML files.
* Automatic backup of question and exam data at regular intervals.

**Preferences:**

* Allow customization of preferences such as date formats, language, etc.

**Documentation:**

* User manual explaining system usage and features.
* Include a README file with installation instructions and any prerequisites.

**Technologies Used:**

* C# (Windows Forms)
* XML
* XSD (XML Schema Definition)
* XSLT (XML Style sheet Language)