Your organization needs an Examination system.

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, every Question Object of Question List will be logged to a Same file. each Question List has Different File

(help on TextWrite and TextReader Class).

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)