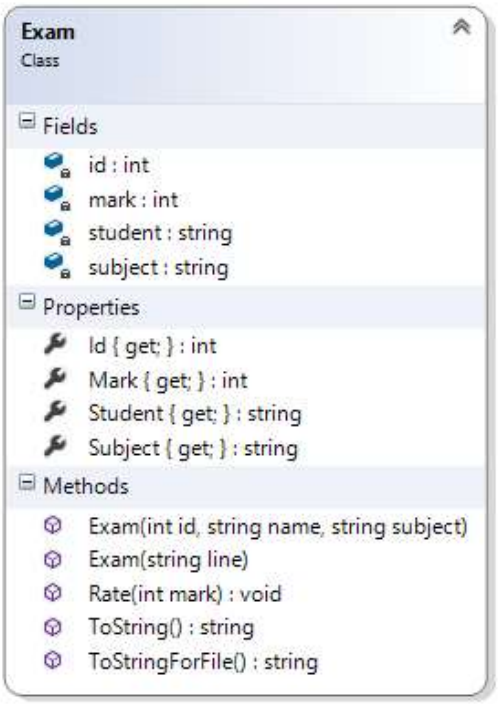
16_MaturaCatalog

Implementiere ein C# Anwendung, mit der die Maturprüfungen erfasst und verwaltet werden können.

Die Daten der Prüfungen befinden sich in der Datei exams.txt. Jede Prüfung wird in einer Zeile durch eine eindeutige Prüfungsnummer (Id), einem Namen des geprüften Schülers (student), die Gegenstandsbezeichnung (subject) und die Prüfungsnote (mark) gespeichert.

Klasse Exam:



Diese Klasse implementiert eine Prüfung.

Felder:

- id – die eindeutige Id einer Prüfung
- student – der Name des geprüften Schülers
- subject – der Gegenstand
- mark – die Note als int (1-5, 0 = keine Note)

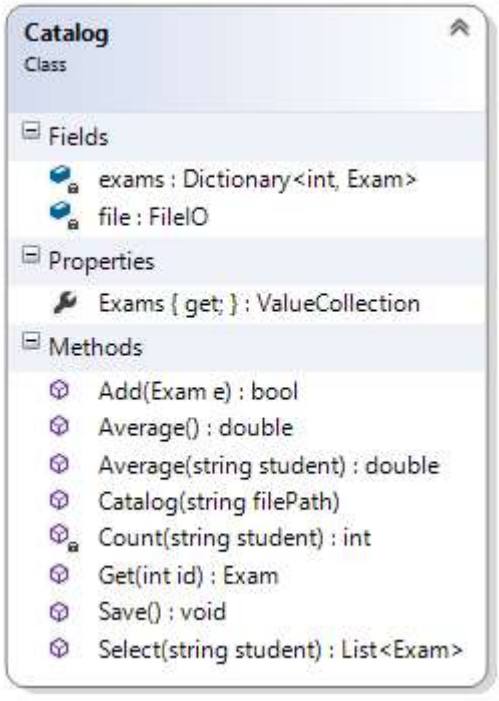
Properties:

- alle Felder sollen für die Anzeige in der GUI mit Hilfe von Properties lesbar gemacht werden. Benenne die Properties wie im Klassendiagramm.

Methoden:

- Der Konstruktor initialisiert alle Felder bis auf die Note mit den Parameterwerten. Die Note wird zuerst auf 0 gesetzt und kann erst später mit Rate() erfasst werden. Sollte die Id keine positive Zahl sein oder name bzw. subject Leerstrings sein, so muss eine ArgumentException geworfen werden.
- ToString – Stringdarstellung für die Anzeige in der ListBox (siehe Screenshot der GUI).
- ToStringForFileIO – liefert einen String zum Speichern als Zeile in der Datei, wo alle Feldwerte durch Strichpunkte voneinander getrennt werden.
- Rate() – hiermit wird die Note zur Prüfung eingetragen. Die Werte 1-5 sind erlaubt. Unerlaubte Werte führen zu einer ArgumentException.

Klasse Catalog



Diese Klasse verwaltet die Exam-Objekte.

Felder:

- exams – ist ein Dictionary, das alle Exam-Objekte speichert, Key ist die Id der Prüfung.
- file – ist das FileIO-Objekt zum Speichern der Exam-Objekte in der Datei exams.txt im Projektordner.

Properties:

- Exams – gibt die ValueCollection mit allen Exams zurück für die Anzeige in der GUI.

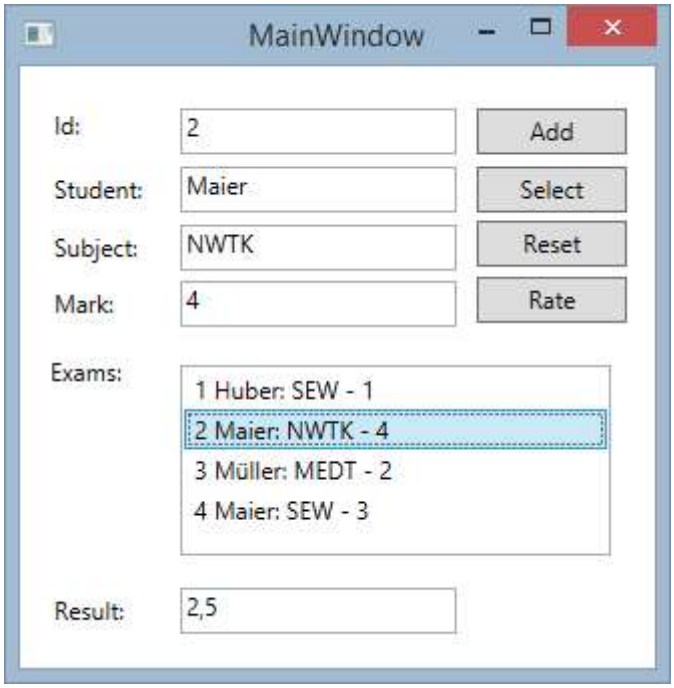
Methoden:

- Der Konstruktor initialisiert alle Felder und bekommt den Dateipfad für FileIO.
- Add – fügt eine neue Prüfung hinzu und liefert true, wenn es noch keine zweite Prüfung für den gleichen Schüler mit diesem Gegenstand gibt. Weiters dürfen pro Schüler maximal 3 unterschiedliche Prüfungen vorhanden sein.
- Average – liefert die Durchschnittsnote aller Prüfungen eines Schülers.

- Count – liefert die Anzahl der erfassten Prüfungen eines Schülers. Wird nur von der Klasse Catalog benötigt.
- Get – liefert zu einer id die Prüfung
- Save – speichert alle Prüfungen in die Datei.
- Select – liefert alle Prüfungen eines bestimmten Schülers.

GUI

Sobald das Programm startet wird, werden auch alle Prüfungen aus der Datei exams.txt geladen und angezeigt.



Der Benutzer kann eine neue Prüfung mit einer eindeutigen Id und Schülernamen aber nur ohne Note erfassen.

Wählt der Benutzer eine Prüfung in der Listbox aus, so werden die Daten der Prüfung in den Eingabefeldern angezeigt.

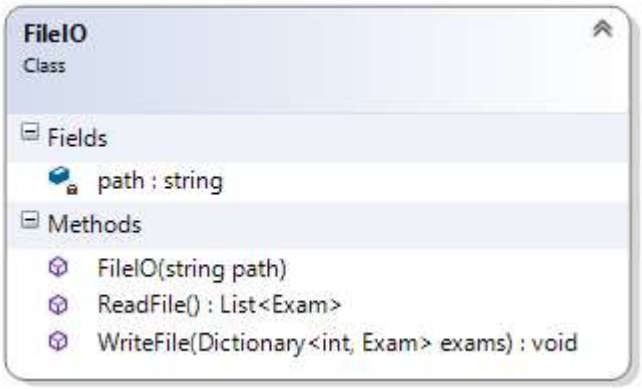
Mit dem Button Rate kann er die Note zur ausgewählten Prüfung festlegen.

Mit dem Button Select werden nur für einen bestimmten Schüler die Prüfungen aufgelistet.

Mit dem Button Reset werden wieder alle Prüfungen von allen Schülern aufgelistet.

Im Result-Feld wird immer die Durchschnittsnote angezeigt.

Klasse FileIO



Generell gilt, dass Änderungen in der Liste der Exams sofort gespeichert werden. Erstelle und verwende dafür die Klasse FileIO.

Felder:

- filePath – Der Pfad zur Datei

Methoden:

- Der Konstruktor initialisiert alle Felder.
- ReadFile – liest alle Exams aus der Datei und liefert sie als Liste.
- WriteFile – speichert alle Exams im Dictionary exams in der Datei. Der Inhalt der Datei wird überschrieben.

Achtung: Achte darauf, dass im Fehlerfall immer eine passende Exception geworfen bzw. auf der GUI eine passende Fehlermeldung ausgegeben wird!

Gutes Gelingen!