2025/11/14 15:45 1/7 LU09.A01 - Schulverwaltung

LU09.A01 - Schulverwaltung

Sie können in einer komplexen Anwendung selbständig



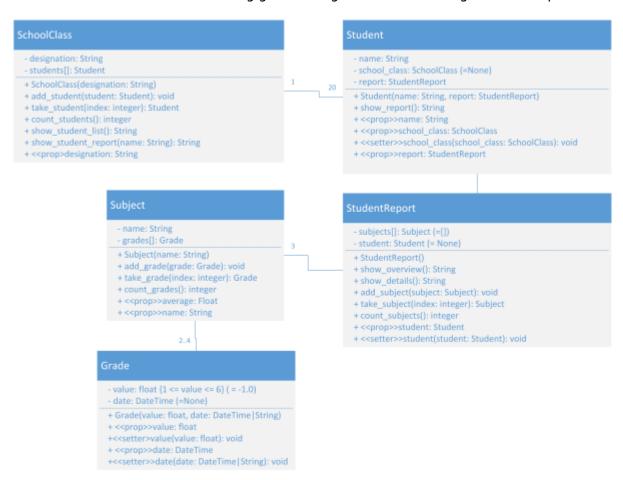
- die Klassen erstellen
- die Beziehungen einpflegen (einseitig, zweiseitig, mehrfache)
- den nötigen Ablauf selbst festlegen
- die gefordeten Ausgaben erzeugen

Vorgehen

- Studieren Sie jeweils das UML-Diagramm sowie die Erklärungen zu den einzelnen Methoden.
- Halten Sie sich an die Reihenfolge bei der Erstellung der Klassen.
- Testen Sie die jeweilige Klasse, bis alle Test erfolgreich ablaufen.

Auftrag

Es ist eine einfache Schulverwaltung gemäss folgendem Klassendiagramm zu implementieren.



Dabei nutzen Sie Ihr Wissen zu ein- und zweiseitiger Beziehung sowie den 4 gezeigten Fällen der

Referenzzuweisung. Ebenso verwenden Sie Mehrfachbeziehungen.



Vorgehen

Die Beschreibung der Klassen und Methoden erfolgt von links nach rechts und von oben nach unten. Erstellen Sie selbständig einen Plan, in welcher Reihenfolge Sie die Klassen erstellt wollen. Ein paar Tipps:

- Es lohnt sich zunächst das Gerüst einer Klasse mit allen Methoden zu bauen.
- Klassen die keine Referenzen zu anderen Klassen enthalten, sollten zuerst umgesetzt werden.
- Versuche danach Klassen umzusetzen, bei denen die referenzierte Klasse bereits erstellt ist.
- Der Konstruktor einer Klasse sollte in der Regel zuerst erstellt werden.
- Danach folgen Methoden ohne Logik, z.B. property und setter.
- Bei Methoden die eine Verarbeitungslogik enthalten, wird zunächst einfach ein hart codierter Returnwert zurück gegeben.

Hinweise

- Testen Sie jede Klasse/Methode mit den jeweiligen Unit Tests. Führen Sie die Tests einzeln aus, da vor allem zu Beginn sehr viele Tests scheitern werden.
- Die Methoden show_... liefern immer einen String als Returnwert. Der print-Befehl wird nur im main() genutzt.

https://wiki.bzz.ch/ Printed on 2025/11/14 15:45

2025/11/14 15:45 3/7 LU09.A01 - Schulverwaltung

SchoolClass

Konstruktor

• Die Schreibweise students[] : Student im Klassendiagramm zeigt an, dass es sich um eine Liste (Array) handelt.
Initialisieren Sie das Attribut als leere Liste.

add_student

- Fügt einen Studenten in die Liste ein.
- Beachten Sie, dass gemäss Klassendiagramm max. 20 Studenten möglich sind. Das müssen Sie beim Zufügen von Studenten umsetzen.

count_student

• Gibt die Anzahl Studenten zurück.

take student(index)

- Liefert den Studenten beim angegebenen Index.
- Stellen Sie sicher, dass ein ungültiger Index zu keinem Laufzeitfehler führt. Stattdessen soll bei einem ungültigen Index der Wert None zurück gegeben werden.

show_student_list

• Diese Methode liefert eine Liste aller Studenten an. Die Ausgabe könnte wie folgt aussehen:

Max

Pia

Cem

show_student_report(name)

• Diese Methode liefert das Zeugnis für einen Studenten mit allen Fächern und dem Notenschnitt an. Die Ausgabe könnte wie folgt aussehen:

• • •

Student

Konstruktor

• Beachten Sie die Parameter und Defaultwerte.

show_report

• Diese Methode liefert das Zeugnis dieses Studenten mit allen Fächern und dem Notenschnitt.

Die Ausgabe könnte wie folgt aussehen:

...

StudentReport

Konstruktor

• Die Schreibweise subjects[] : Subject im Klassendiagramm zeigt an, dass es sich um eine Liste (Array) handelt.
Initialisieren Sie das Attribut als leere Liste.

add_subject

- Beachten Sie, dass gemäss Klassendiagramm max. 3 Fächer möglich sind.
- Beim Versuch mehr als 3 Fächer (Subject) einzufügen, soll die Methode einen ???Error werfen.

take_subject

• Stellen Sie sicher, dass ein ungültiger Index zu keinem Laufzeitfehler führt. Stattdessen soll bei einem ungültigen Index der Wert None zurück gegeben werden.

show overview

. . .

Diese Methode liefert ein Zeugnis mit allen Fächern und dem entsprechenden Notenschnitt.
 Eine mögliche Ausgabe kann wie folgt aussehen:

show details

• Diese Methode liefert alle Fächern mit den einzelnen Noten. Eine mögliche Ausgabe kann wie folgt aussehen:

```
•••
```

https://wiki.bzz.ch/ Printed on 2025/11/14 15:45

2025/11/14 15:45 5/7 LU09.A01 - Schulverwaltung

Subject

Konstruktor

Die Schreibweise grades [] : Grade im Klassendiagramm zeigt an, dass es sich um eine Liste (Array) handelt. Initialisieren Sie das Attribut als leere Liste.

add_grade

- Beachten Sie, dass gemäss Klassendiagramm max. 4 Noten möglich sind. Das müssen Sie beim Zufügen von Noten (Grade-Objekte) umsetzen.
- Die untere Grenze von 2 Noten müssen Sie (noch) nicht beachten.

take_grade

• Stellen Sie sicher, dass ein ungültiger Index zu keinem Laufzeitfehler führt. Stattdessen soll bei einem ungültigen Index der Wert None zurück gegeben werden.

calc_average

- Die Methode berechnet den Notendurchschnitt aus allen Grade-Objekten in der Liste.
- Falls keine Grade-Objekte in der Liste vorhanden sind, wird der Wert 0.00 zurück gegeben.

Unit tests

Testen Sie die Klasse Subject mit den Testfällen in test_subject.py

Grade

Die Klasse Grade wird als @dataclass realisiert.

Konstruktor

- Initialisieren Sie die Werte value und date gemäss Klassendiagramm.
- Achten Sie auf die Zusicherung für den Wert von value. Diese nehmen Sie im <u>post_init</u> vor, d.h. dass Sie keine setter-Methode schreiben (ist gemäss Klassendiagramm nicht vorgesehen).

main

In der main-Methode erzeugen Sie die verschiedenen Objekte und zeigen die Zeugnisse an.

Erzeugen Sie die Objekte in der Reihenfolge, wie sie auch für die Zuweisung in den

Konstruktoren nötig sind. Wenn Sie unsicher sind, skizzieren Sie sich den Ablauf des Programms als Sequenzdiagramm auf.

Ausgabe

Das Programm liefert

- eine Liste der Studenten
- pro Student das Zeugnis (Report) mit dem Notenschnitt
- für einen Studenten alle Einzelnoten. Sie können hier frei wählen, für wen die Noten ausgegeben werden.

Die Ausgabe soll in etwa wie folgt aussehen:

```
Max
Pia
Cem
Zeugnis für: Max
   Mathe: 4.25
   Deutsch: 5.0
   Turnen: 5.0
Zeugnis für: Pia
   Mathe: 5.5
   Deutsch: 5.3333333333333333
   Turnen: 5.25
Zeugnis für: Cem
   Mathe: 4.25
   Deutsch: 5.5
   Turnen: 5.5
   Fach: Mathe mit 2 Noten
       1: 5.0 1.1.11
       2: 3.5 2.2.22
   Schnitt: 4.25
   Fach: Deutsch mit 3 Noten
       1: 5.5 3.3.33
       2: 6.0 4.4.44
       3: 5.0 5.5.55
   Schnitt: 5.5
   Fach: Turnen mit 4 Noten
       1: 4.5 6.6.66
       2: 6.0 7.7.77
       3: 6.0 8.8.88
       4: 5.5 9.9.99
    Schnitt: 5.5
```

Dauer

https://wiki.bzz.ch/ Printed on 2025/11/14 15:45

2025/11/14 15:45 7/7 LU09.A01 - Schulverwaltung

4 - 6 Stunden

Abgabe

Mittels Push ins GitHub Repository

⇒ GitHub Repo für externe Besucher

GitHub Repository https://github.com/templates-python/m319-lu10-a02-reader-module

Lernende am BZZ müssen den Link zum GitHub Classroom Assignment verwenden

M320-LU09



René Probst, bearbeitet durch Marcel Suter

https://wiki.bzz.ch/ - BZZ - Modulwiki

Permanent link:

https://wiki.bzz.ch/modul/m320_2024/learningunits/lu09/aufgaben/schulverwaltung?rev=1726485988

Last update: 2024/09/16 13:26

