Modul 320 OOP Projekt

SCHOOL DATABASE

BBZW
Ayla Seckanovic
Berna Imeri

Inhalt

Projektbeschreibung M320

In diesem Projekt geht es darum, das Objektorientierte Programmieren richtig anzuwenden. Wir haben für die Themenwahl freie Wahl. Somit haben wir uns für ein Schulverwaltungssystem entschieden, welches wir mit Python programmieren. Dieses Projekt haben wir zu zweit gemacht, da man davon mehr profitieren kann, somit haben wir pair-developement gemacht.

Link zum Projekt : https://github.com/kakashxii/M320-Projekt

Anforderungen

- OOP
- Eingebaute Schnittstellen
- Konsolenanwendung

Idee

Die Idee ist es ein Schulverwaltungssystem zu erstellen, um die Daten einer fiktiven Schule zu verwalten. Der Benutzer kann Schüler und Lehrer suchen, löschen oder auch aktualisieren. Wir haben uns an einem Beispiel von einer Seite orientiert.

Wir haben uns entschieden, eine einfache Konsolenanwendung zu erstellen, die für jeden Benutzer leicht und verständlich ist. Außerdem führen wir eine Hilfefunktion ein, die bei der Verwendung der Anwendung hilfreich sein kann.

Natürlich ist es nicht unser Ziel, diesen Code zu kopieren, er dient nur zur Inspiration und als kleine Unterstützung für das Projekt. Zusätzlich würden wir gerne eine Klasse für Unittests hinzufügen, wenn es die Zeit erlaubt.

Python code Erklärung

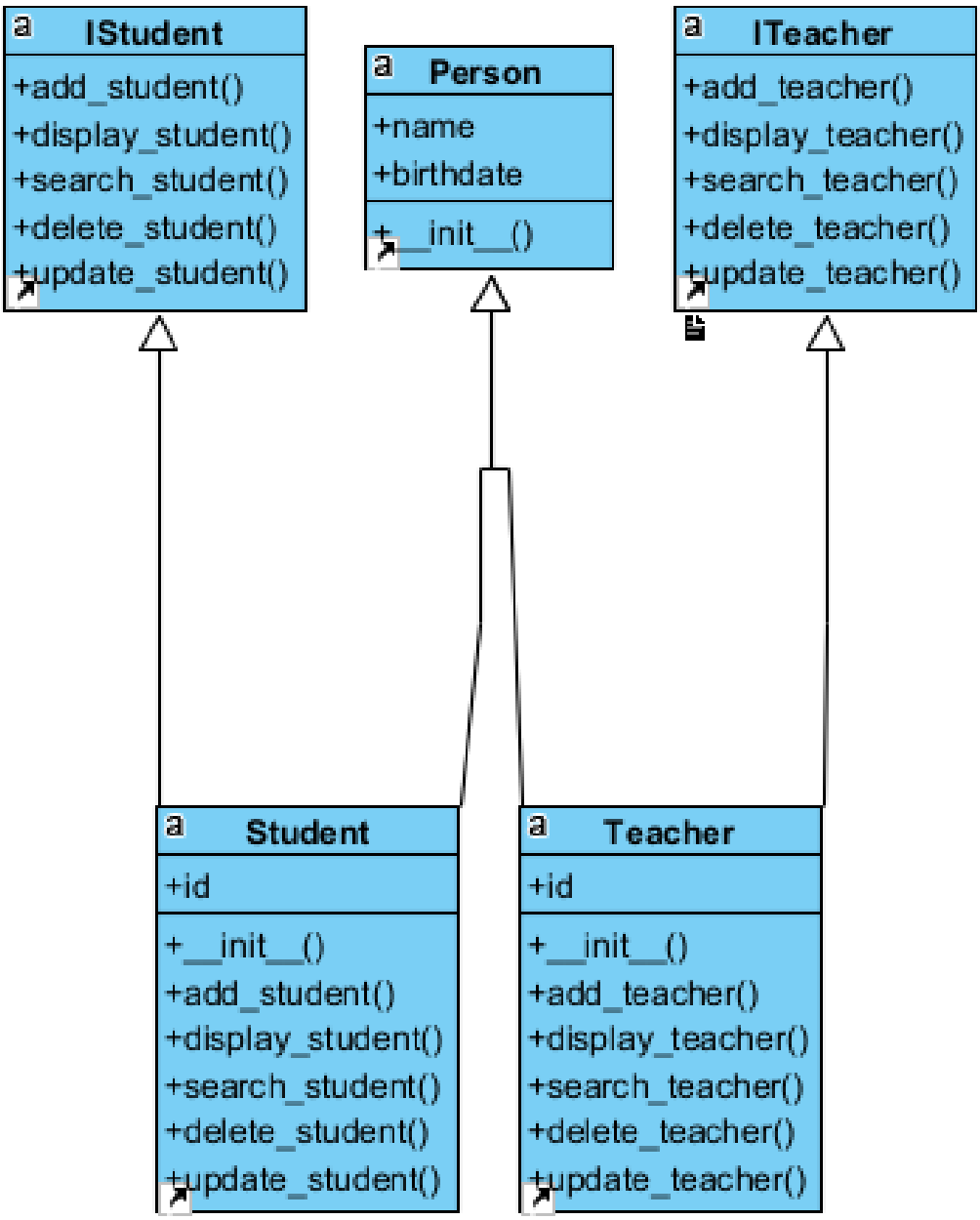
Dies ist ein Python-Programm, das mehrere Klassen zur Darstellung von Lehrern und Schülern definiert. Die Klassen ITeacher und IStudent sind Schnittstellenklassen (Interfaces), die verschiedenen Methoden wie add_teacher, display_teacher, search_teacher, delete_teacher und update_teacher für die Arbeit mit Lehrern definieren. Die gleichen Methoden sind auch für die Arbeit mit Schülern definiert.

Die Klasse Person ist eine Elternklasse für die Klassen Teacher und Student und enthält den Namen und das Geburtsdatum als Eigenschaften sowie eine init-Methode, die diese Eigenschaften festlegt, wenn ein Objekt der Klasse Person erstellt wird.

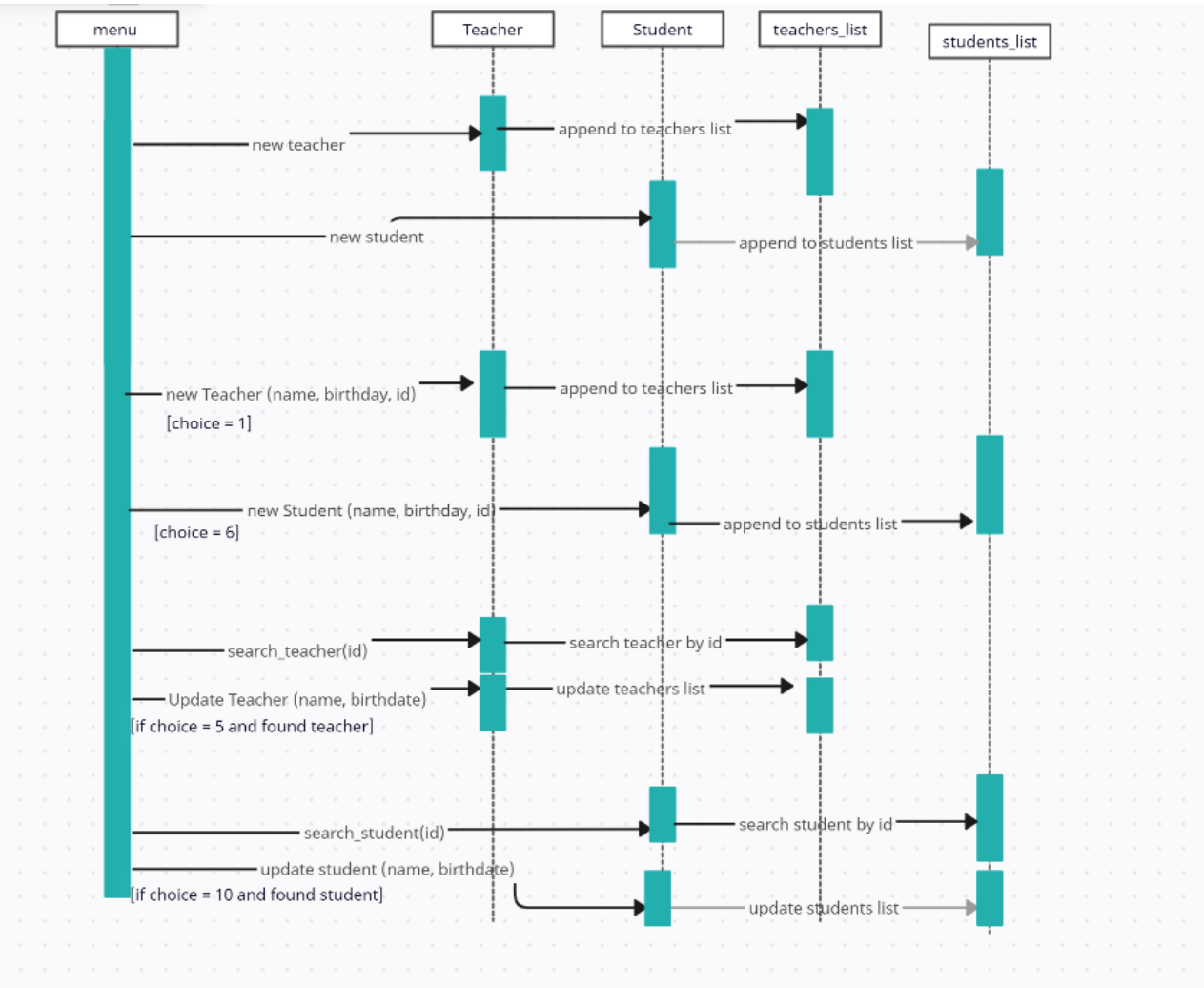
Die Klassen Teacher und Student sind Unterklassen von Person bzw. ITeacher und IStudent. Sie haben die gleichen Methoden wie ITeacher und IStudent, aber jede von ihnen ist für die spezifische Klasse implementiert, d. h. Teacher für Lehrer und Student für Schüler.

Das Programm erstellt außerdem eine globale Liste mit dem Namen teachers_list, in der alle Teacher-Objekte gespeichert werden, und student_list, in der alle Student-Objekte gespeichert werden. Das teacher_obj ist eine Instanz der Klasse Teacher.

Klassendiagramm



Sequenzdiagramm



Interfaces (Python)

Es ist möglich, Schnittstellen in Python auf ähnliche Weise zu verwenden, wie sie in C# verwendet werden. In Python sind Schnittstellen nicht eingebaut wie in anderen Sprachen, sondern sie können durch abstrakte Klassen und abstrakte Methoden implementiert werden.

Wenn eine Klasse von einer abstrakten Klasse erbt, muss sie für jede der abstrakten Methoden, die in der Elternklasse definiert sind, eine Implementierung bereitstellen.

Wir haben eine Schnittstelle namens ITeacher definiert, die dieselben fünf Methoden wie die Klasse Teacher hat, wobei alle den @abstractmethod-Dekorator haben. Die Klasse Teacher hat all diese Methoden implementiert, nun muss sie sowohl von ITeacher als auch von Person Class erben, auch weil alle Methoden in ITeacher als abstrakt deklariert sind und von jeder konkreten Klasse implementiert werden müssen, die von ihr erbt.

Wir haben from abc import ABC, abstractmethod hinzugefügt, um eine abstrakte Basisklasse (ABC) und den Dekorator @abstractmethod zu definieren.

Das gleiche passiert bei der Klasse IStudent.

Die Implementierung für die in ITeacher und IStudent deklarierten Methoden haben wir in den Klassen Teacher und Student hinzugefügt.