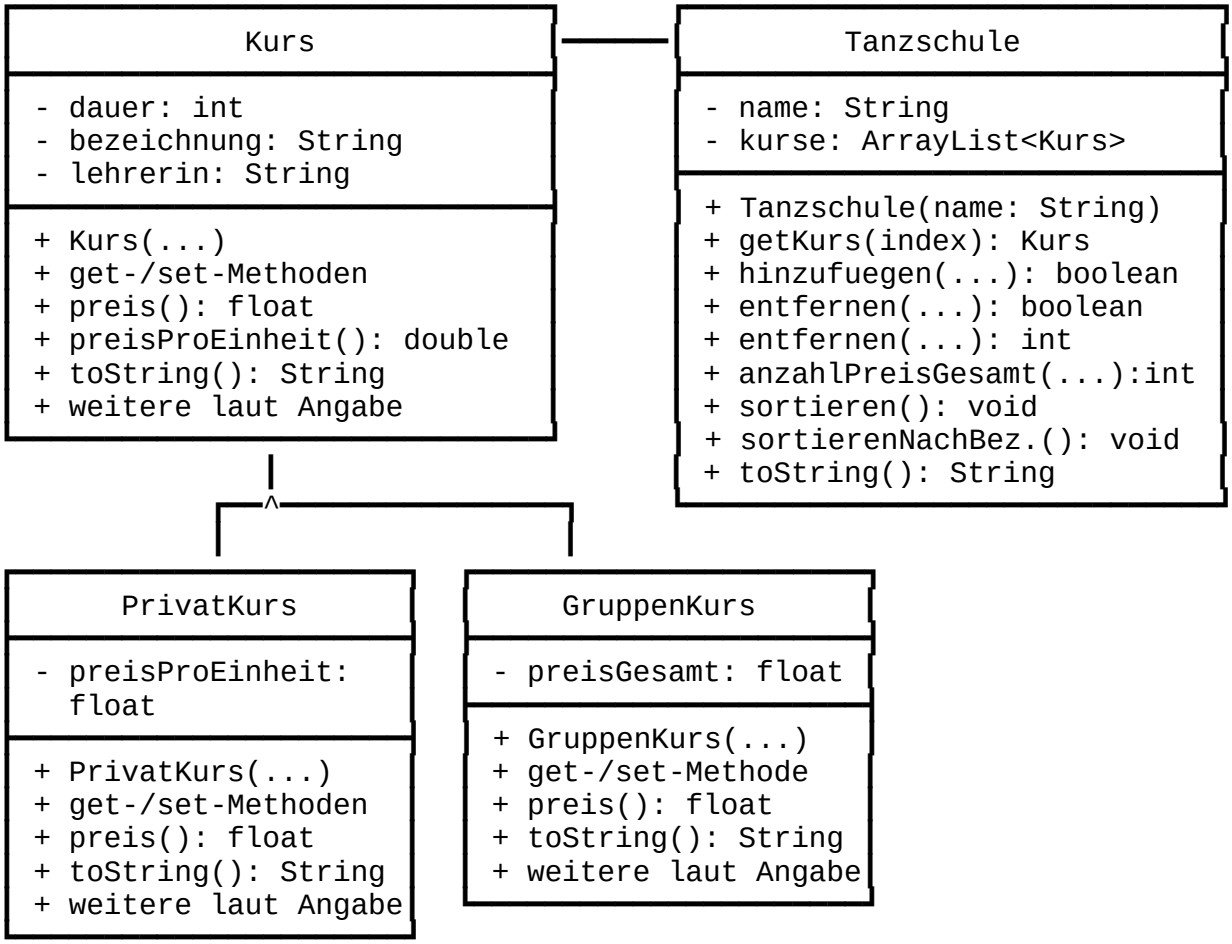
ÜBUNG VERERBUNG „TANZSCHULE“

Eine Tanzschule benötigt für die Verwaltung der Tanzkurse eine Java-Applikation. Eine Tanzschule bietet Kurse an. Derzeit werden Privatkurse und Gruppenkurse angeboten. Gruppenkurse haben einen Pauschalpreis und Privatkurse werden pro Einheit bezahlt. Eine Einheit dauert immer 45 Minuten. Bei allen Kursen wird eine Dauer vergeben, damit ist die Anzahl der geplanten Kurseinheiten gemeint. Gruppenkurse haben üblicherweise eine Dauer von 12 (gemeint ist damit die Anzahl an Kursabenden). Die Dauer von Privatkursen wird individuell zwischen Tanzschüler:innen und Tanzlehrer:innen vereinbart. Üblich sind bei Privatkursen Dauern zwischen 1 und 6 (Anzahl der zwischen Tanzlehrer:in und Tanzschüler:in vereinbarten Tanzeinheiten). Eine Erweiterung der Tanzschule-Verwaltung um z.B. Jugendkurse oder Seniorenkurse wäre in Zukunft möglich, ist aber derzeit nicht geplant.

Die **Klassen und Methoden** im folgenden UML-Diagramm **sind von Ihnen zu implementieren**. Verwenden sind für die Lösung die **IDE IntelliJ** und die **Java API-Doc**.



1. KLASSE KURS

- Ein Kurs existiert nicht als „eigenständiges Objekt“.
- Ein Kurs hat eine Dauer in Einheiten (Attribut dauer: int).
- Die Dauer stellt die „natürliche Ordnung“ von Kursen dar.
- Ein Kurs hat eine Bezeichnung (Attribut bezeichnung: String).
- Ein Kurs wird von einer Tanzlehrer:in durchgeführt (Attribut lehrerin: String).

Konstruktor Kurs(dauer, bezeichnung, lehrerin):

Es wird ein Kurs mit der übergebenen Dauer (Parameter dauer: int), der Bezeichnung (Parameter bezeichung: String) und dem Namen der Lehrer:in (Parameter lehrerin: String) erzeugt.

get- und set-Methoden:

Die folgenden Überprüfungen der Übergabeparameter müssen erfolgen:

- Die Dauer muss mindestens 1 sein.
- Strings dürfen nicht null oder leer sein.

Geben Sie im Fehlerfall erklärende Meldungen auf der Systemkonsole aus.

preis(): float: Die Methode ist hier abstrakt und muss in den ableitenden Klassen implementiert werden.

preisProEinheit(): double: Die Methode liefert den Preis pro Kurseinheit dieses Kurses zurück. Der Kurspreis wird von der Methode preis() geliefert. Die Anzahl der Kurseinheiten enthält das Attribut dauer.

toString(): String: Die Methode liefert alle Informationen eines Kurses in einer Zeile zurück:

Beispiel:

```
2 Einheiten, Bezeichnung: Samba individuell
Tanzlehrer:in: Peter
Preis: 100.0 EUR, Preis pro Einheit: 50.0 EUR
```

2. KLASSE PRIVATKURS

PrivatKurse sind Kurse.

Konstruktor PrivatKurs(dauer, bezeichnung, lehrerin, preisProEinheit):

Es wird ein PrivatKurs mit den übergebenen Attribut-Werten der Superklasse Kurs (dauer: int, bezeichung: String, lehrerin: String) und dem übergebenen Preis pro Einheit (Parameter preisProEinheit: float) erzeugt. Der Wert von preisProEinheit muss größer als 0.0 sein.

preis(): float: Die abstrakte Methode aus der Superklasse wird überschrieben. Der Preis eines Privatkurses ist die Dauer des Kurses multipliziert mit dem Preis pro Einheit des Privatkurses.

toString(): String:

Die Methode liefert alle Informationen eines Privatkurses beginnend mit dem Text „PrivatKurs:“ in einer Zeile zurück, Beispiel:

```
Privatkurs
2 Einheiten, Bezeichnung: Samba individuell
Tanzlehrer:in: Peter
Preis: 100.0 EUR, Preis pro Einheit: 50.0 EUR
```

3. KLASSE GRUPPENKURS

GruppenKurse sind Kurse.

Konstruktor GruppenKurs (dauer, bezeichnung, lehrerin, preisGesamt):

Es wird ein GruppenKurs mit den übergebenen Attribut-Werten der Superklasse Kurs (dauer: int, bezeichung: String, lehrerin: String) und dem übergebenen Preis für den gesamten Kurs (Parameter preisGesamt: float) erzeugt. Der Wert von preisGesamt muss größer als 0.0 sein.

preis(): float: Die abstrakte Methode aus der Superklasse wird überschrieben. Der Preis eines Gruppenkurses ist der Wert des Attributs preisGesamt.

toString(): String:

Die Methode liefert, bis auf den Wert des Attributs preisGesamt, alle Informationen eines Gruppenkurses beginnend mit dem Text „GruppenKurs:“ in einer Zeile zurück, Beispiel:

```
Gruppenkurs
2 Einheiten, Bezeichnung: Samba individuell
Tanzlehrer:in: Peter
Preis: 100.0 EUR, Preis pro Einheit: 50.0 EUR
```

4. KLASSE TANZSCHULE

Eine Tanzschule hat einen Namen (Attribut name) und eine Liste von Kursen (Attribut kurse).

Konstruktor Tanzschule(name: String):

Es wird eine Tanzschule mit dem übergebenen Namen (Parameter name: String) und einer leeren ArrayList<Kurs> erzeugt.

get- und set-Methoden:

Es darf keine Methode getKurse() geben. Die Methode setKurse(kurse: ArrayList<Kurs>) muss den Zugriffsmodifizierer private erhalten.

Die folgenden Überprüfungen der Übergabeparameter müssen erfolgen:

- Übergabeparameter dürfen nicht null sein
- Strings dürfen nicht leer sein

Geben Sie im Fehlerfall erklärende Meldungen auf der Systemkonsole aus.

getKurs(index: int): Kurs: Liefert die Referenzvariable des Kursobjekts beim übergebenen Index aus der ArrayList kurse zurück. Ist der übergebene index ungültig, dann wird null zurückgeliefert.

hinzufuegen(kurs: Kurs): boolean:

- Die Methode fügt die übergebene Objektreferenz der ArrayList hinzu und liefert true zurück.
- Ist der Kurs schon vorhanden, dann wird dieser nicht hinzugefügt und es wird false zurückgeliefert.
- Es kann mit der derzeitigen Personalsituation der Tanzschule maximal 50 Kurse geben.
- Ist kein Platz mehr für einen weiteren Kurs oder ist der übergebene Parameter null, dann wird false zurückgeliefert.

entfernen(kurs: Kurs): boolean: Entfernt den übergebenen Kurs aus der Tanzschule und liefert true zurück. Ist der übergebene Parameter null oder ist der Kurs nicht vorhanden, dann wird false zurückgeliefert.

entfernen(lehrerin: String): int: Entfernt alle Kurse der Tanzlehrer:in mit dem übergebenen Namen aus der Tanzschule und gibt die Anzahl der entfernten Kurse zurück. Ist der übergebene Parameter null dann wird der Fehlercode -99 zurückgeliefert.

anzahlPreisGesamt(preisGesamt: float): int: Liefert die Anzahl jener Gruppenkurse der Tanzschule zurück, deren Attribut-Wert bei preisGesamt genau dem übergebenen Parameterwert von preisGesamt entspricht. Ist der übergebene Parameter ungültig oder gibt es keine Gruppenkurse in der Tanzschule, dann wird der Fehlercode -99 zurückgeliefert.

sortieren(): void: Sortiert die Kurse in der ArrayList kurse nach deren natürlicher Ordnung. Die Dauer stellt die „natürliche Ordnung“ von Kursen dar.

sortierenNachBezeichnung(): void: Sortiert die Kurse in der ArrayList kurse nach deren Bezeichnung aufsteigend.

toString(): String: Die Methode liefert die Werte aller Attribute der Klasse Tanzschule in einer übersichtlichen Form zurück. Die vorhandenen Kurse soll untereinander dargestellt werden.

5. TESTKLASSE

Erstellen Sie die **JUnit-Testklasse** TanzschuleTest mit den erforderlichen Testmethoden zum Testen folgender Methoden der Klasse Tanzschule:

- hinzufuegen(kurs: Kurs): boolean
- entfernen(lehrerin: String): int
- anzahlPreisGesamt(preisGesamt: float): int
- sortierenNachBezeichnung(): void

Gutes Gelingen und viel Spaß!