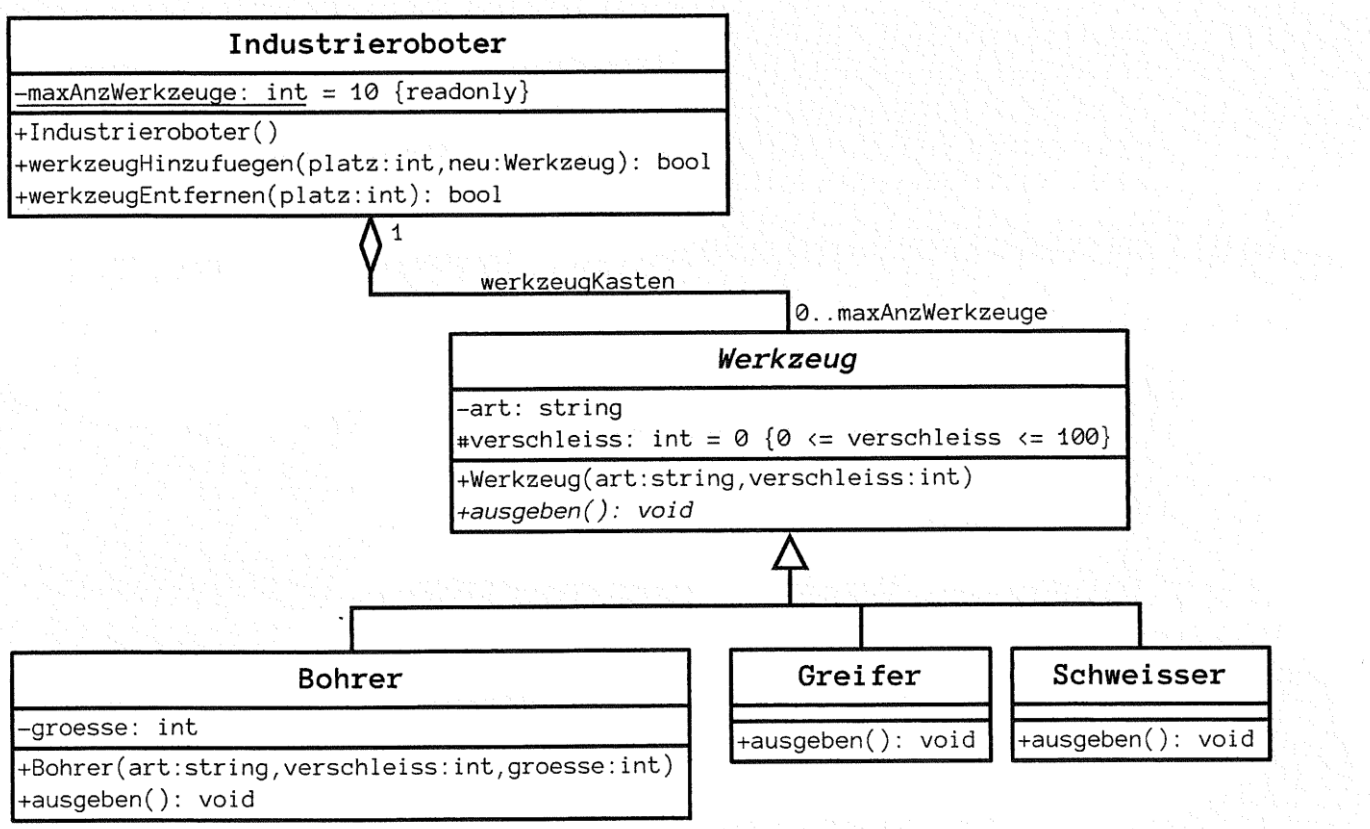
Aufgaben aus alten Fachinformatiker Prüfungen

Übungsblock: Aggregationen

Ziel: UML Diagramme verstehen und Implementierung von Aggregationen

Aufgabe 1 Gegeben ist folgende Klassenhierarchie:



Ein Industrieroboter enthält ein Werkzeugkasten mit einer festen Anzahl von Werkezugplätzen. Die Plätze sind von 0 an durchnummeriert und können auch leer sein. An jedem Platz maximal ein Werkzeug abgelegt werden (z.B. Bohrer, Greifer, Schweißer (Schweiß-Zange)). Werkzeuge können hinzugefügt oder entfernt werden.

1. Implementieren Sie Klassen Industrieroboter, Werkzeug und Bohrer. Als Zusatzinformation stehen Ihnen folgende Tabellen zur Verfügung.

Klasse Industrieroboter

maxAnzWerkzeuge	Die maximale Anzahl der Werkzeuge, die der Roboter verwenden kann.
Industrieroboter	Initialisiert die Attribute falls erforderlich.
werkzeugHinzufuegen	Setzt das übergebene Werkzeug an den übergebenen Platz des Werkzeugkastens, wenn der Platz existiert und nicht bereits ein Werkzeug enthält. Der Rückgabewert ist bei erfolgreichem Hinzufügen TRUE, sonst FALSE. Es wird ausgegeben, welches Werkzeug wo hinzugefügt wurde bzw. warum das Werkzeug nicht hinzugefügt werden konnte.
werkzeugEntfernen	Entfernt das Werkzeug vom übergebenen Platz des Werkzeugkastens, aber nur wenn der der Platz existiert und bereits ein Werkzeug enthält. Der Rückgabewert ist bei erfolgreichem Entfernen TRUE, sonst FALSE. Es wird ausgegeben welches Werkzeug wo entfernt wurde bzw. warum das Werkzeug nicht entfernt werden konnte.

Klasse Werkzeuge

art	Identifiziert die Art des Werkzeugs.
verschleiss	Eine Prozentangabe die den Verschleiß des Werkzeugs angibt, wobei 0 bedeutet, dass das Werkzeug keinen Verschleiß aufweist und 100, dass das Werkzeug komplett verschlissen ist.
Werkzeuge	Initialisiert die Attribute mit übergebenen und/oder Standardwerten.

Klasse Bohrer

groesse	Größe des Bohrers in Millimetern.
Bohrer	Initialisiert die Attribute mit übergebenen und/oder Standardwerten.
ausgeben	Gibt den Text „Bohrer mit Groesse x (Verschleiss y %)." aus, wobei x und y durch die Werte des entsprechenden Bohrers ersetzt werden.

2. Schreiben Sie ein Testprogramm für folgende Testfälle → Erzeugen Sie jedoch einen Industrieroboter und zwei Bohrer (Bohrer 1 und Bohrer 2, beide mit der Größe 10 und Verschleiß 0).

Testfall:	Erwartete Ausgabe:
Hinzufügen von Bohrer 1 an Platz 5	Hinzugefuegtes Werkzeug auf Platz 5: Bohrer mit Gro-esse 10 (Verschleiss 0 %).
Hinzufügen von Bohrer 2 an Platz 5	Hinzufuegen nicht moeglich, da Platz 5 belegt ist.
Hinzufügen von Bohrer 2 an Platz 10	Hinzufuegen nicht moeglich, da Platz 10 nicht existiert.
Hinzufügen von Bohrer 2 an Platz -1	Hinzufuegen nicht moeglich, da Platz -1 nicht existiert.
Werkzeug entfernen von Platz 5	Entferntes Werkzeug auf Platz 5: Bohrer mit Groesse 10 (Verschleiss 0 %).
Werkzeug entfernen von Platz 5	Entfernen nicht moeglich, da Platz 5 nicht belegt ist.
Werkzeug entfernen von Platz 10	Entfernen nicht moeglich, da Platz 10 nicht existiert.
Werkzeug entfernen von Platz -1	Entfernen nicht moeglich, da Platz -1 nicht existiert.

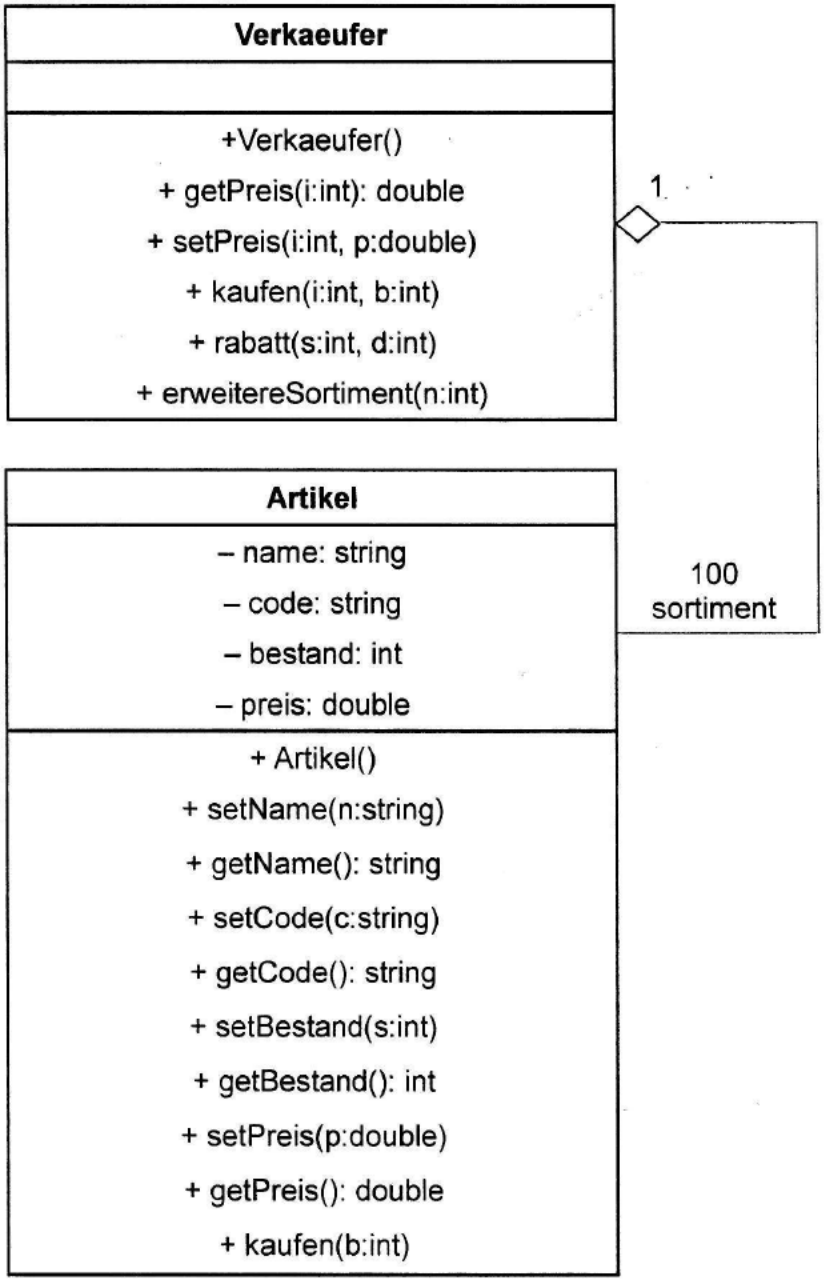
Aufgabe 2:

Die Warenwirtschafts-Software eines zukünftigen Online-Vertriebs soll programmiert werden. Der Projektleiter beauftragt Sie als Teammitglied mit der lmplementierung der Software-Bestandteile für das Artikel-Sortiment und dessen Preisgestaltung.

Erstellen Sie eine Klasse **Verkaeufer** (siehe UMl-Klassendiagramm unten). Die Klasse Verkaeufer soll die folgenden Methoden besitzen:

- Einen Konstruktor **Verkaeufer**, der sämtliche Elemente des Arrays **sortiment** mit Objekten vom Typ **Artikel** initialisiert.
- Eine Methode **getPreis**, die als Parameter einen ganzzahligen Wert i erhält. **getPreis** soll als Rückgabewert den Preis des Artikels liefern, der innerhalb des Arrays **sortiment** den lndex i hat.
- Eine Methode **setPreis**, die als Parameter einen ganzzahligen Wert i und den Fließkommawert p erhält. **setPreis** soll den Preis des Artikels, der innerhalb des Arrays Sortiment den lndex i hat, auf p setzen.
- Eine Methode **kaufen**, die als Parameter zutei ganzzahlige Werte i und b erhält. **kaufen** soll den Bestand des Artikels, der innerhalb des Arrays **sortiment** den lndex i hat, um b erhöhen.
- Eine Methode **rabatt**, die den Preis jedes Artikels innerhalb des Arrays **sortiment**, dessen Bestand größer als s Exemplare ist, um den Prozentwert d reduziert.
- Eine Methode **erweitereSortiment**, die das aktuelle Array **sortiment** um n Elemente erweitert.

Übernehmen sie vorhandene Artikel und initialisieren sie die neu hinzugekommenen Elemente mit Objekten vom Typ Artikel.



Hinweise:
Um die Ausgabe in einem geeigneten Währungsformat müssen Sie sich nicht kümmern. Deklarieren und implementieren Sie jeweils vollständig die Klasse und die Methoden.
Sie können davon ausgehen, dass keine ungültigen Parameter übergeben werden.