

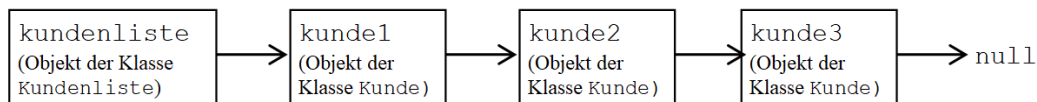
Information

Während der Laufzeit eines Programms werden häufig Objekte in Listen gespeichert. Bisher wurden nur starre Listen verwendet, welche unter dem Begriff Array bekannt sind. Die Javaklasse `ArrayList` verwendet ebenfalls Arrays. Neben den starren Listen gibt es verkettete Listen, welche auch dynamische oder gelinkte Listen genannt werden.

1. Aufgabe

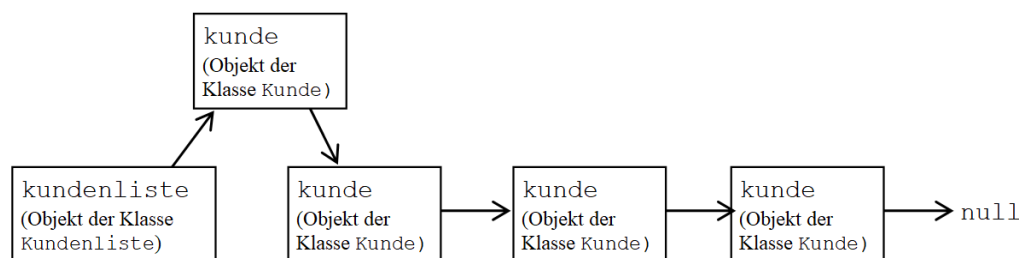
Eine Software soll Kunden in einer einfach verketteten Liste verwalten. Diese Software wurde bereits ansatzweise erstellt und steht Ihnen als Javacode (Moodle) zur Verfügung.

In der folgenden Abbildung ist das Prinzip einer einfach verketteten Liste schematisch (kein genormtes UML-Diagramm) dargestellt.



1.1 Erklären Sie unter Verwendung des oben erwähnten Javacodes und der oben angegebenen Abbildung das Prinzip der einfach verketteten Liste.

1.2 Die folgende Abbildung zeigt, wie ein neues Objekt der Klasse Kunde am Anfang der Liste eingefügt wird.



- a)** Stellen Sie graphisch dar (in Anlehnung an das Bild oben), wie folgende Vorgänge realisiert werden:
- Einfügen eines neuen Objekts am Ende der Liste
 - Löschen eines Objekts am Anfang der Liste
 - Löschen eines Objekts am Ende der Liste
 - Löschen eines Objekts in der Mitte der Liste

- b)** Beim Löschen von Objekten in Java spielt der Garbage-Collector eine wichtige Rolle. Erklären Sie das.

1.3 Die Klasse `Kundenliste` soll jetzt mehrere Methoden erhalten, mit denen die verkettete Liste verwaltet werden kann.

Die Methode `getAnzahl` gibt die Anzahl der gespeicherten Objekte zurück. Diese Anzahl ist in einem Attribut zu speichern.

Den Methoden `ein fuegenAnfang` und `ein fuegenEnde` wird jeweils ein Objekt der Klasse `Kunde` übergeben. Sie fügen dieses Objekt am Anfang bzw. Ende der Liste ein.

Der Methode `suchen` wird eine Kundennummer übergeben. Sie sucht das entsprechende Objekt in der Liste und gibt es zurück.

Der Methode `loeschen` wird eine Kundennummer übergeben. Sie löscht das entsprechende Objekt, falls es existiert und gibt `true` oder `false` zurück.

- a)** Stellen Sie die Klassen `Kunde` und `Kundenliste` in einem Klassendiagramm dar.
- b)** Implementieren Sie die oben beschriebenen Methoden.

1. Aufgabe

Starre Listen und verkettete Listen sollen miteinander verglichen werden.
Nennen Sie für die verkettete Listen je einen Vor- und einen Nachteil.

2. Aufgabe

Welcher Unterschied besteht zwischen einfach und doppelt verketteten Listen?

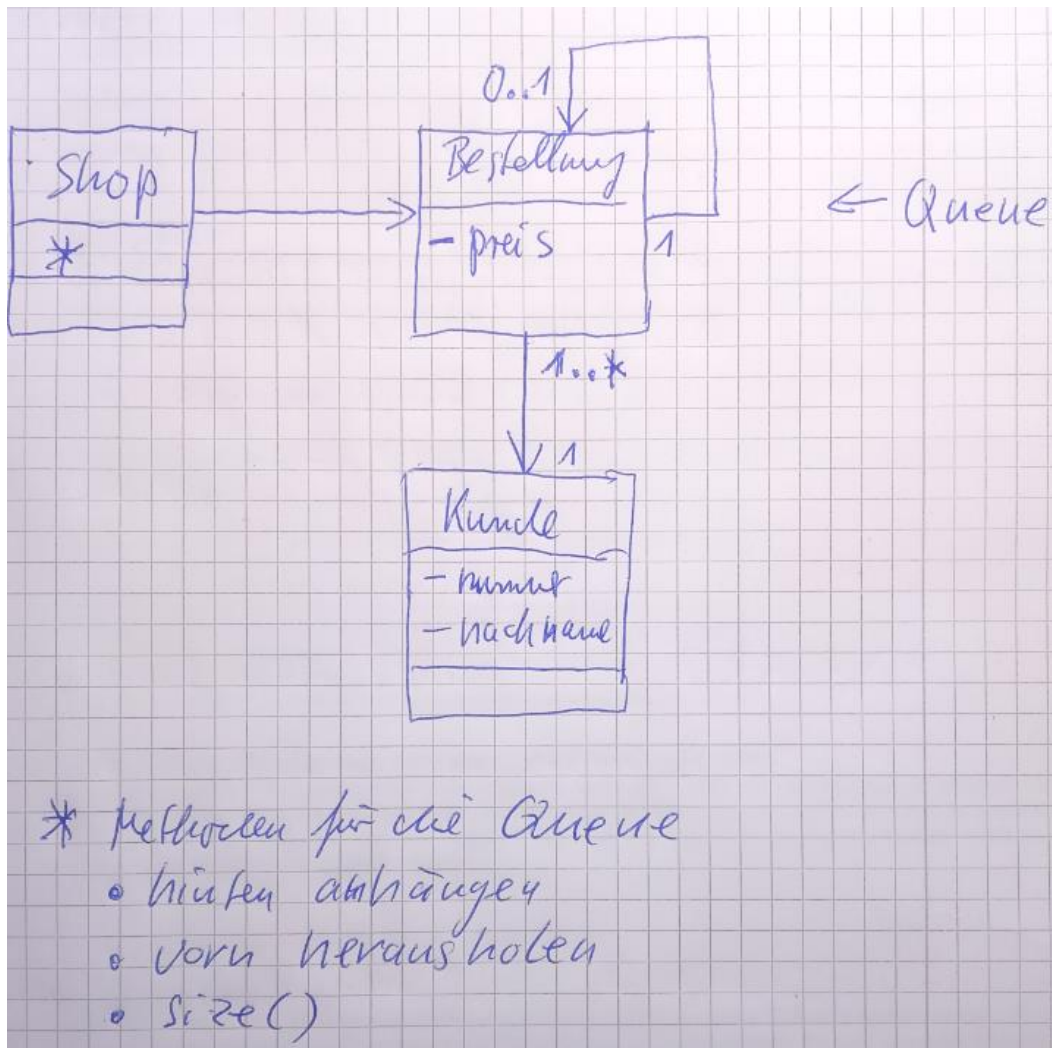
3. Aufgabe

Informieren Sie sich über die Listentypen Queue (Warteschlange) und Stack (Stapel, Kellerspeicher).
Beschreiben Sie die Methoden, welche für diese Listentypen jeweils implementiert werden müssen.

4. Aufgabe

Die Software eines Onlineshops soll alle Bestellungen ihrer Kunden in einer einfach verketteten Liste speichern, welche als Listentyp Queue zu realisieren ist.

Es existiert bereits der Ansatz eines Klassendiagramms. Erstellen Sie den Javacode und testen Sie ihn.



1. Aufgabe

Wird eine verkettete Liste als Queue verwendet, so ist es zweckmäßig, dass von der Listenverwaltung ein direkter Zugriff auf beide Enden der Liste möglich ist.

Die Lösung der Aufgabe 4 von der Seite 2 soll angepasst werden, vom Objekt der Klasse Shop muss jetzt ein direkter Zugriff auf die erste und die letzte Bestellung möglich sein.

Ändern Sie entsprechend das Klassendiagramm und den Code.

2. Aufgabe

Neben den einfach verketteten Listen gibt es auch doppelt (zweifach) verkettete Listen.

2.1 Erläutern Sie kurz den Unterschied zwischen beiden Listentypen.

2.2 Das für die Aufgabe 1 von Seite 1 erstellte Programm soll so verändert werden, dass es jetzt mit einer doppelt verketteten Liste arbeitet.

a) Erstellen Sie das Klassendiagramm.

b) **Zusatzaufgabe**

Ändern Sie den Code entsprechend.

Zum Testen ist eine zusätzliche Methode zu erstellen. Mit dieser Methode wird die Liste vom ersten bis zum letzten Objekt und danach wieder rückwärts zum ersten Objekt durchlaufen. Bei beiden Durchläufen sind sämtliche Kundennummern auszugeben.

3. Aufgabe

Thema: Stack

Bearbeiten Sie die Aufgaben 1.1 und 1.2 von dieser Abiturprüfung:

Hessisches Kultusministerium

Landesabitur 2018 (Nachtermin)

**Datenverarbeitungstechnik
Leistungskurs**

**Thema und Aufgabenstellung
Vorschlag A**

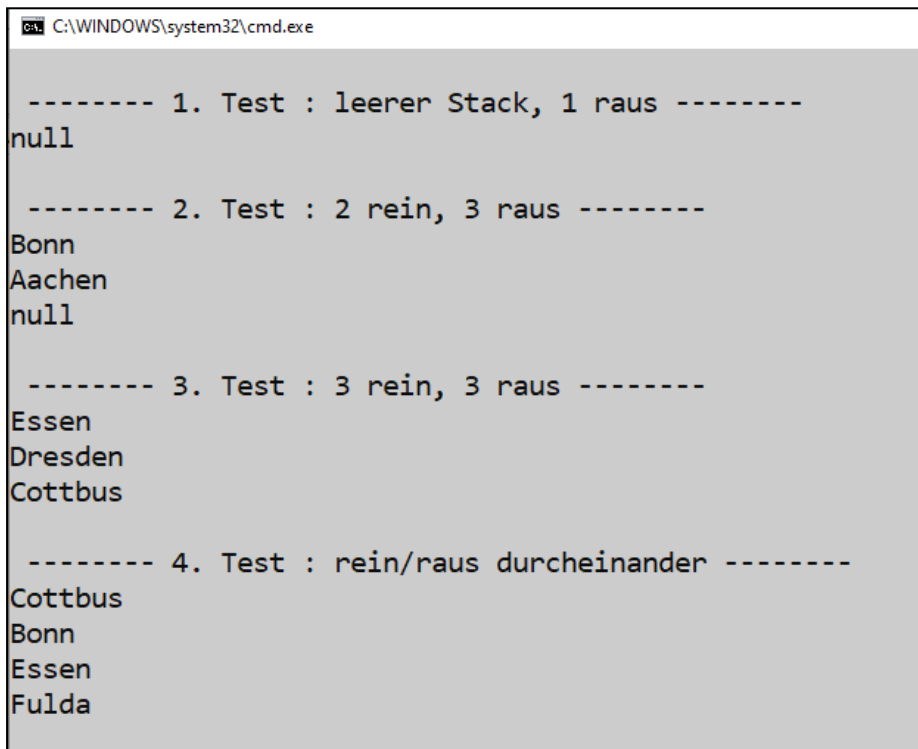
1. Aufgabe

Auf Seite 3 wurden Sie aufgefordert, eine Abiturprüfung (teilweise) zu bearbeiten, welche Ihnen auf Papier ausgehändigt wurde. Diese Prüfungsaufgabe wird jetzt erweitert:

- a) Erstellen Sie den Javacode der Klasse `Gleis`. Falls die Klasse `Waggon` noch nicht erstellt wurde, so muss dies jetzt geschehen.
(In beiden Klassen darf die Methode `toString` entfallen.)
- b) Sie haben ein Testprogramm (Java-Datei `Testklasse` mit der `Main`-Methode) erhalten. Mit diesem Programm werden die Methoden der Klasse `Gleis` (und indirekt auch der Klasse `Waggon`) getestet. Dieses Programm kann selbstverständlich nur dann mit dem von Ihnen erstellten Code funktionieren, wenn alle Vorgaben eingehalten wurden. Wurde z. B. der Bezeichner einer Methode falsch geschrieben, so gibt es logischerweise ein Problem beim kompilieren. In diesem Fall müssen Sie Ihren Code solange überarbeiten, bis das Testprogramm lauffähig ist.

Falls Sie den Code der Klassen `Gleis` und `Waggon` korrekt erstellt haben, so wird das Testprogramm genau die Konsolenausgabe erzeugen, welche nachfolgend abgebildet ist. Bei Abweichungen, insbesondere beim Erzeugen einer Exception (Ausnahmefehler, welcher während der Programmlaufzeit entsteht), müssen Sie Ihren Code entsprechend korrigieren.

Konsolenausgabe für den Fall, dass die Klassen `Gleis` und `Waggon` korrekt implementiert wurden:



```
C:\WINDOWS\system32\cmd.exe

----- 1. Test : leerer Stack, 1 raus -----
null

----- 2. Test : 2 rein, 3 raus -----
Bonn
Aachen
null

----- 3. Test : 3 rein, 3 raus -----
Essen
Dresden
Cottbus

----- 4. Test : rein/raus durcheinander -----
Cottbus
Bonn
Essen
Fulda
```

ZUSATZAUFGABEN

Die folgenden Zusatzaufgaben beziehen sich auf ein Programm, welches Sie bereits beim Bearbeiten der Seite 1 erstellt haben. Dieses Programm ist jetzt zu ändern bzw. zu erweitern.

- a) Erstellen Sie in der Java-Klasse `Kundenliste` eine Methode `sortiertEinfuegen`, welche Kunden-Objekte immer so in die Liste einfügt, dass diese nach der Kundennummer aufsteigend sortiert ist. Warum ist es jetzt wichtig, dass die anderen Methoden zum Einfügen entfernt werden müssen?
- b) Erstellen Sie in der Java-Klasse `Kundenliste` eine Methode `erzeugenTextdatei`, welche alle in der Liste gespeicherten Kundendaten in eine Textdatei schreibt.

Beispielhafter Inhalt der Textdatei:

Kundennummer	Name
3556	Meier
3558	Schulze
3559	Lehmann