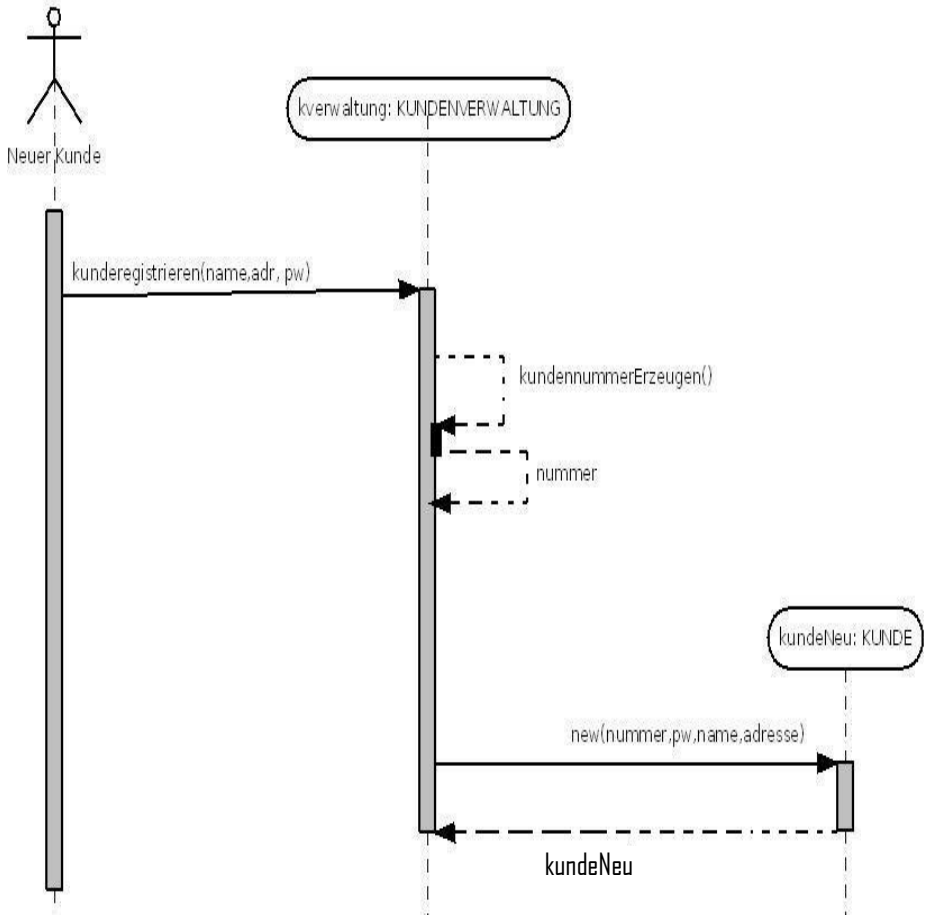
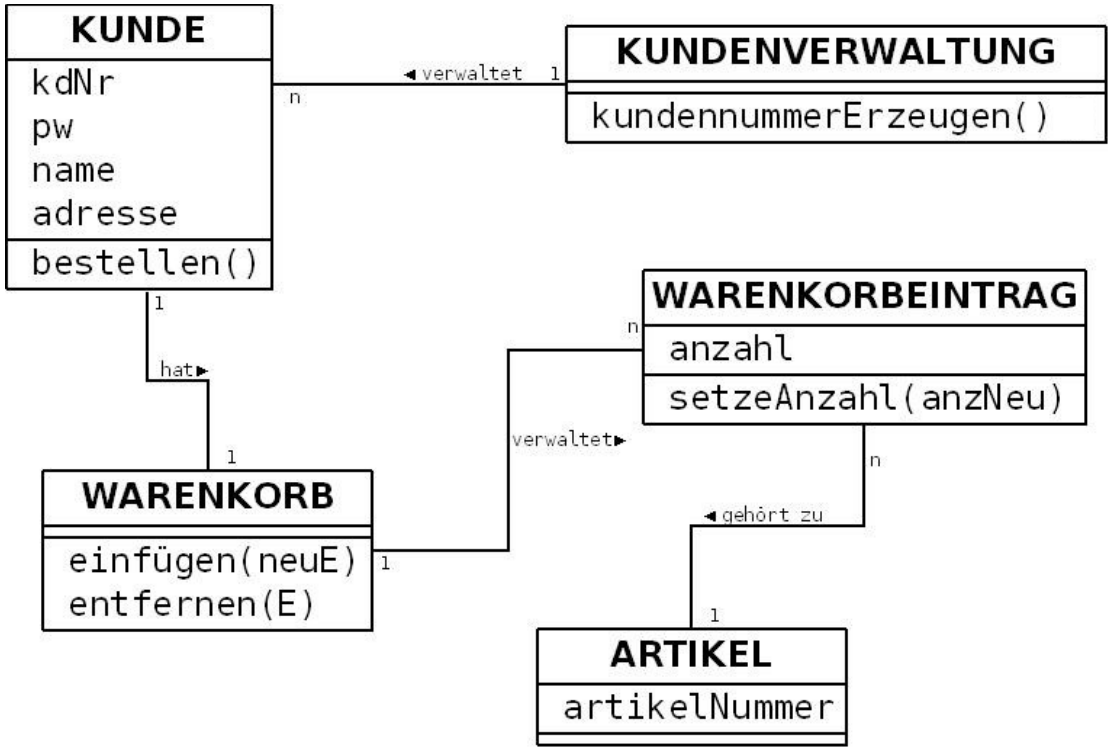
Informatik Abitur Bayern 2014 / II - Beispiellösung

Autorin: Hippeli

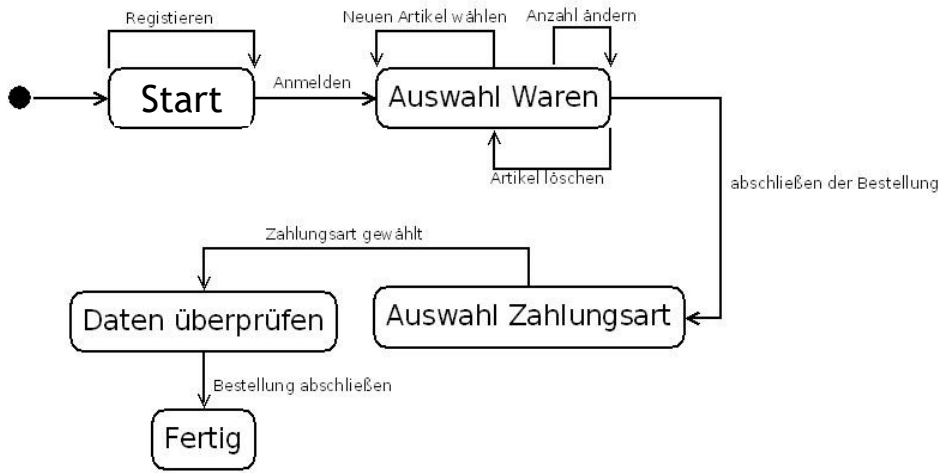
1a 5



1b 8



1c 8



Hier lassen sich natürlich auch andere Zustandsbezeichnungen finden.

1d **Klasse WARENKORB:** 12

```
float gesamtbruttopreisGeben(){
        return anfang.bruttopreisGeben(); //rekursiver Methodenaufruf
}

WARENKORBEINTRAG artikelSuchen(int nr){
        return anfang.artikelSuchen(nr); //rekursiver Methodenaufruf
}
```

Klasse LISTENELEMENT:

```
abstract float bruttopreisGeben();
abstract WARENKORBEINTRAG artikelSuchen(int nr);
```

Klasse ABSCHLUSS:

```
float bruttopreisGeben(){
        return 0.0f; //da kein Artikel zum Abschluss gehört
}

WARENKORBEINTRAG artikelSuchen(int nr){
        return null; //da kein Artikel zum Abschluss gehört
}
```

Klasse KNOTEN:

```
float bruttopreisGeben(){[
        return nachfolger.bruttopreisGeben() +daten.bruttopreisGeben();
//Es wird für den Nachfolger der Bruttopreis berechnet und der Bruttopreis für den aktuellen
//Knoten dazu addiert
}

WARENKORBEINTRAG artikelSuchen(int nr){
        if(daten.artikelHatNummer(nr)){ //ist der Knoten der der gesuchte?
                return daten;
        } else {
                return nachfolger.artikelSuchen(nr); //teste den Nachfolger
        }
}
```

Klasse WARENKORBEINTRAG:

```
float bruttopreisGeben(){
        return artikel.bruttopreisGeben()*anzahl;
}

WARENKORBEINTRAG artikelhatNummer(int nr){
```

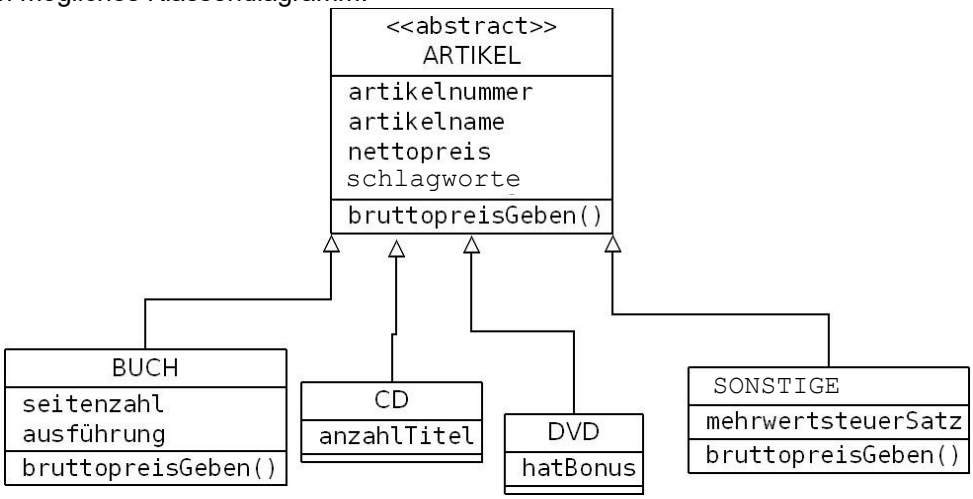
```
            return artikel.NummerGeben()==nr  //rekursive Methodenaufruf
        }
```

Klasse ARTIKEL:

```
        WARENKORBEINTRAG NummerGeben(){
            return nummer;
        }
```

2a Die verschiedenen Artikel haben bestimmte Attribute und Methoden gemeinsam (z.B. besitzt jeder Artikel eine Artikelnummer). Sie sind jedoch Spezialversionen der Klasse ARTIKEL, denn sie besitzen zusätzliche Attribute, auch muss die Ausführung einzelner Methoden angepasst werden. 7

Ein mögliches Klassendiagramm:



Für die Objekte der Klassen CD und DVD wird der Bruttopreis gleich berechnet, daher wird die entsprechende Methode in der Klasse ARTIKEL implementiert und in den anderen Klassen überschrieben, um an die Situation dort angepasst zu sein.

2b **Achtung Mathematik:** Wird ein Preis um x% erhöht so muss das 1,x-fache vom Preis berechnet werden!! 5

Klasse ARTIKEL:

```
        float bruttopreisGeben(){
            return nettopreis * 1.19f;
        }
```

Klasse BUCH:

```
        float bruttopreisGeben(){
            return nettopreis * 1.07f;
        }
```

Klasse SONSTIGE:

```
        float bruttopreisGeben(){
            return nettopreis * (1.0f + mehrwertsteuerSatz/100);
        }
```

3a <u>z.B.: Das Wasserfallmodell</u> 8

- Anforderungsanalyse: Festlegung der Anforderungen an das System. Ergebnis ist eine Anforderungsbeschreibung → Dient als "Vertrag" zwischen Anwender/Entwickler
- Entwurf: Modellierung des Systems
- Implementierung: Codierung des Entwurfs in einer Programmiersprache
- Test und Integration: Test der einzelnen Komponenten, Einbau, Systemtest
- Einsatz & Wartung: Fehlerbeseitigung nach Inbetriebnahme

Hinweis: Da die Wartung nicht mehr zur eigentlichen Erstellung gehört, muss sie daher nicht aufgeführt sein.

Zu welchen Zeitpunkt eine Projekts ein bestimmter Teil erstellt sein muss, legen die Meilensteine fest. Im Wasserfallmodell sind das typischerweise der Übergang von einer Phase in die nächste.

3b Durch die Bearbeitung der Klasse GESPRÄCHSVERWALTUNG entstehen kritische Abschnitte, da diese Ressource, während sie von einem Team geändert wird, von anderen Teams nicht verändert werden darf. 4
Zur Koordination eignen sich Monitore; Methoden werden durch das Schlüsselwort synchronized geschützt; zu einer Zeit kann nur eine der geschützten Methoden ausgeführt werden.

4a Der Baum ist nach Artikelnummern sortiert, daher liefert ein Durchlauf nach der Inorder-Stategie die Artikel in der gewünschten Reihenfolge. Die bedingte Anweisung stellt dabei sicher, dass nur Artikel ausgegeben werden, welche die Suchbedingung erfüllen. 8

Klasse ARTIKELBAUM

```
void artikelAuflisten(String schlagwort){
        anfang.artikelAuflisten(String schlagwort); //rekursive Methodenaufruf
}
```

Klasse BAUMELEMENT:

```
abstract artikelAuflisten(String schlagwort);
```

Klasse ABSCHLUSS:

```
void artikelAuflisten(String schlagwort){
        //fertig, da für Abschluss nichts zu tun ist;
}
```

Klasse KNOTEN:

```
void artikelAuflisten(String schlagwort){
        linkerNF.artikelAuflisten(String schlagwort); //ruft die Methode für den
                                                      //LINKEN Nachfolger auf
i       if(daten.istSchlagwortVorhanden(schlagwort)){ //Wird Suchbedingung erfüllt?
            daten.nameAusgeben()
        }
        rechterNf.artikelAuflisten(String schlagwort); //ruft die Methode für den
                                                       //RECHTEN Nachfolger auf
}
```

4b SQL-Abfrage allgemein (nicht gefordert)_ 2

```
SELECT <Ausgabeliste>
FROM <Liste der verwendeten Tabellen>
WHERE <Liste von Bedingungen>
```

Konkrete SQL-Abfrage:

```
SELECT *
FROM artikel
WHERE artikelnummer = 123456
```

Hinweis: Wir suchen nur über eine Tabelle(hier „artikel“) und möchte ALLE Informationen über Artikel mit der Artikelnummer 123456; daher verwenden wir das *-Symbol im SELECT-Statement.

5a Die Matrix enthält aus Übersichtsgründen nur jene Einträge, die eine Kante mit Gewichtung repräsentieren 3

	A	B	C	D	E	L
A		70		60		40
B	70		50			50
C		50		30	20	30

D	60		30			
E			20			
L	40	50	30			

Da der Graph ungerichtet ist, ist die Matrix symmetrisch zur Diagonale.

5b
```
Class GRAPH{
        KNOTEN[] knotenliste;
        int[][] adjazenzmatrix;
        GRAPH(){
                knotenliste = new KNOTEN [6];
                adjazenzmatrix = new int[6][6];
}
```
4

5c Mit dem Algorithmus **Tiefensuche** können alle Knoten eines zusammenhängenden Graphen ausgehend von einem Startknoten systematisch besucht werden. Der Name „Tiefensuche“ rührt daher, dass das Verfahren zunächst in die „Tiefe“ und erst dann in die „Breite“ des Graphen läuft. Man geht also vom jeweiligen Knoten erst zu einem noch nicht besuchten Nachbarknoten und setzt dort den Algorithmus rekursiv fort. 6