Hinweise für den Prüfling

Auswahlzeit: 30 Minuten

Bearbeitungszeit: 240 Minuten

Auswahlverfahren

Wählen Sie von den zwei vorliegenden Vorschlägen einen zur Bearbeitung aus. Der nicht ausgewählte Vorschlag muss am Ende der Auswahlzeit der Aufsicht führenden Lehrkraft zurückgegeben werden.

Erlaubte Hilfsmittel

1. ein Wörterbuch der deutschen Rechtschreibung
2. ein eingeführter Taschenrechner (Bei grafikfähigen Rechnern und Computeralgebrasystemen ist ein Reset durchzuführen.)
3. eine Liste der fachspezifischen Operatoren Fachbereich III

Sonstige Hinweise

keine

In jedem Fall vom Prüfling auszufüllen

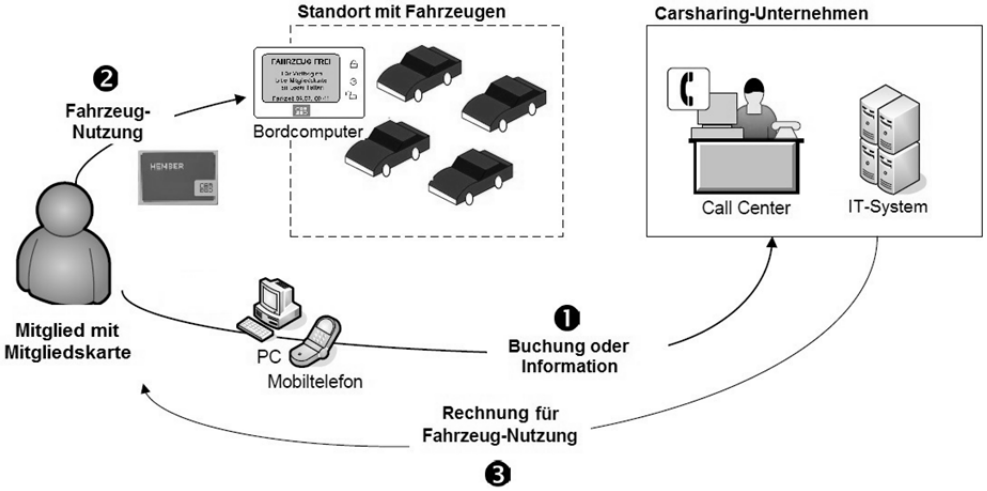
Name: _____ Vorname: _____

Prüferin/Prüfer: _____ Datum: _____

Carsharing-System

Aufgaben

Carsharing bedeutet, dass mehrere Personen Fahrzeuge aus einem Pool nutzen, d. h. sich mehrere Fahrzeuge „teilen“. Eigentümerin oder Eigentümer (in der Folge Eigentümer genannt) der Fahrzeuge ist das Carsharing-Unternehmen. Im vorliegenden Fall ist dies ein Verein, dessen Mitglieder die Nutzungsrechte an den Fahrzeugen besitzen und für deren Benutzung Nutzungsentgelte zahlen.



Die Buchung (❶) eines Fahrzeugs kann durch ein registriertes Mitglied per Internet oder Telefon erfolgen.
Die Fahrzeug-Nutzung (❷) erfolgt nach Buchung ab dem Standort des Fahrzeugs. Nach der Fahrzeug-Nutzung muss das Fahrzeug zurück zum Standort gebracht werden.
Für die Fahrzeug-Nutzung werden vom Carsharing-Unternehmen monatlich Rechnungen (❸) gestellt.

1 Objektorientierte Entwicklung eines Software-Systems
Sie haben die Aufgabe ein Software-System für ein Carsharing-Unternehmen zu entwickeln.

1.1 Anwendungsfalldiagramme dienen dazu, die Anwendungsfälle eines Software-Systems auf einem hohen Abstraktionsniveau abzubilden, um einen Überblick über das Gesamtsystem zu geben.
Beschreiben Sie das in Material 1 dargestellte unvollständige UML-Anwendungsfalldiagramm.

(4 BE)

1.2 Entwickeln und zeichnen Sie für das Carsharing-System auf Grund folgender Anforderungen ein Anwendungsfalldiagramm in korrekter UML-Notation.

- Personen können sich von einer Mitarbeiterin oder einem Mitarbeiter (in der Folge Mitarbeiter genannt) des Carsharing-Unternehmens als Mitglieder registrieren lassen.
- Die Verfügbarkeit von Fahrzeugen kann geprüft werden.
- Mitglieder können verfügbare Fahrzeuge buchen.
- Buchungen können storniert werden.
- Einmal pro Monat wird für jedes Mitglied eine Rechnung ausgestellt.
- Fahrzeuge werden abgeholt und nach ihrer Nutzung wieder zurückgebracht.

Hinweis: Das UML-Anwendungsfalldiagramm in Material 1 ist zu ergänzen.

(6 BE)

1.3 Überführen Sie die Klassen `Mitglied` und `Buchung` (Material 2) in entsprechende Anweisungen einer objektorientierten Programmiersprache und implementieren Sie die angegebenen Methoden.

Hinweise: Die Methode `bucheFahrzeug()` der Klasse `Mitglied` kann das gewünschte Fahrzeug nur dann buchen, wenn es im angegebenen Zeitraum auch frei ist. Die Methode `berechnePreis()` ermittelt den Preis in €. get- und set-Methoden müssen nicht implementiert werden, sind aber vorhanden und können benutzt werden.
Die Klasse `DateTime` ist in Material 6 dokumentiert.

(10 BE)

1.4 Das Carsharing-System soll um weitere Anforderungen ergänzt werden.
In den Nutzungsbedingungen für Mitglieder ist Folgendes zu lesen:

> Bei uns haben Sie rund um die Uhr die Auswahl zwischen verschiedenen Fahrzeugklassen und Modellen. Sie zahlen nur für die Zeit, in der Sie das Auto brauchen.
> Wenn Sie ein Fahrzeug buchen wollen, geben Sie bitte zunächst
> – Ihre Mitgliedsnummer an.
> Anschließend wählen Sie
> – die Fahrzeugklasse (z.B. Mini, Klein, Mittel, Van),
> – den Standort, von dem Sie ein Fahrzeug nutzen wollen (z.B. Bahnhof) und den
> – gewünschten Nutzungszeitraum.
> Fahren können Sie nur, wenn das Buchungssystem Ihre Buchung bestätigt hat.
> Mit der Wahl der Fahrzeugklasse sind die Nutzungskosten festgelegt. Abgerechnet wird nach Nutzungsdauer in Minuten (z. B. in der Mini-Klasse 0,08 € pro Minute). Die Abrechnung erfolgt monatlich.

Entwickeln und zeichnen Sie ein UML-Klassendiagramm, das alle zur Erfüllung der Anforderungen benötigten Klassen mit Attributen, Konstruktoren, Methoden und Assoziationen enthält.

Hinweise: Das UML-Klassendiagramm in Material 2 entsprechend zu ergänzen. Falls Attribute und Methoden bereits bestehender Klassen gestrichen, hinzugefügt oder geändert werden müssen, ist dies unmissverständlich kenntlich zu machen.

(10 BE)

1.5 Jedes Fahrzeug besitzt einen Bordcomputer, welcher das Öffnen und Schließen des Fahrzeugs ermöglicht. Darüber hinaus erlaubt er die Erfassung von Fahrdaten und die Funk-Kommunikation mit der Carsharing-Zentrale.

1.5.1 Bei der Abholung wird das Fahrzeug von außen mittels einer Mitgliedskarte (kontaktlose Chipkarte) geöffnet. Zum Öffnen des Fahrzeugs wurde für die Methode `fetchCar()` der Klasse `Bordcomputer` (Material 3) ein UML-Sequenzdiagramm entwickelt (Material 4).
Beschreiben Sie anhand des im UML-Sequenzdiagramm wiedergegeben Ablaufs den Vorgang zur Abholung eines Fahrzeugs.

(5 BE)

1.5.2 Implementieren Sie anhand des UML-Sequenzdiagramms (Material 4) die Methode `fetchCar()`.

(10 BE)

1.5.3 Mittels der Methode `readCard()` der Klasse `RFIDReader`[1] (Material 3) kann die Mitgliedsnummer einer Mitgliedskarte über die serielle Schnittstelle gelesen werden. Dabei ist das in Material 5 wiedergegebene Datenformat zu beachten. Zu der Folge von Datenbytes wird eine Prüfsumme bestimmt. Stimmt die ermittelte Prüfsumme mit den gelesenen Kartendaten überein, wird die Folge der Datenbytes als String zurückgegeben. Im Fehlerfall wird ein leerer String zurückgegeben.
Entwickeln und zeichnen Sie für die Methode `readCard()` ein Struktogramm.

Hinweis: Es sind nur die Methoden zu verwenden, die in der Klasse `RFIDReader` und in der Klasse `Serial` (Material 6) angegeben sind.

(10 BE)

1.5.4 Bordcomputer sind technische Systeme, die verschiedene Aufgaben bewältigen. Sie steuern das Öffnen und Schließen des Fahrzeugs, erfassen Fahrdaten und kommunizieren mit der Carsharing-Zentrale via Funkmodul (Radio Module). Zum Beispiel fragt das Carsharing-Unternehmen regelmäßig den Aufenthaltsort der Fahrzeuge ab.
Zeigen Sie am Beispiel des Bordcomputers, dass es sinnvoll ist, bestimmte Vorgänge in einem Thread ablaufen zu lassen und beschreiben Sie, wie Threads in der von Ihnen verwendeten Programmiersprache realisiert werden.

(5 BE)

2 Datenbank für das Carsharing-Unternehmen

Sie werden von dem Carsharing-Unternehmen beauftragt, seine Fahrzeug- und Mietverwaltung durch eine relationale Datenbank zu unterstützen. In dem Datenbanksystem soll u. a. aufgenommen werden, welches Fahrzeug von welchem Mitglied gebucht bzw. genutzt wurde. Ihnen wird ein vorläufiges ER-Modell (Material 7) vorgelegt.

2.1 Überführen Sie das gegebene ER-Modell (Material 7) in das relationale Modell.

Hinweise: Die Primärschlüssel und Fremdschlüssel sind zu kennzeichnen. Alle Relationen sind in der folgenden Schreibweise anzugeben: `Relation (PK, Attribut, … , FK#)`

(5 BE)

2.2 Einige Daten der Datenbank sollen geändert bzw. ausgewertet werden.

2.2.1 Geben Sie eine SQL-Abfrage an, die die Fahrzeugklassen ermittelt, die weniger als 0,15 € pro Minute und weniger als 0,20 € pro km kosten.

(2 BE)

[1] RFID (engl. radio-frequency identification) ermöglicht das berührungslose Identifizieren und Lokalisieren von Objekten und Lebewesen mit Radiowellen.

2.2.2 Formulieren Sie eine SQL-Abfrage mit der aller Mitglieder mit Name und Vorname ausgegeben werden, die schon einmal einen Mercedes Smart ausgeliehen haben.

Hinweise: Die Ergebnisliste soll aufsteigend nach Name und Vorname sortiert sein. Die Mehrfachnennung von Mitgliedern ist zu unterbinden.

(3 BE)

2.2.3 Entwickeln Sie eine SQL-Abfrage, die alle Kunden, die schon mindestens vier Mal ein Fahrzeug der Mittelklasse gebucht haben, mit Name, Vorname und der Anzahl dieser Buchungen ausgibt.

(4 BE)

2.2.4 Erläutern Sie den Aufbau und die inhaltliche Bedeutung der folgenden SQL-Anweisung.

```
SELECT buchungsNr, Buchung.mitgliedsNr, name, vorname, kennzeichen
FROM Buchung
INNER JOIN Mitglied ON Buchung.mitgliedsNr = Mitglied.mitgliedsNr
INNER JOIN Fahrzeug ON Buchung.kennzeichen = Fahrzeug.kennzeichen
WHERE Buchung.mitgliedsNr IN (1002, 1007, 1009);
```

(3 BE)

2.2.5 Das Carsharing-Unternehmen hat ein neues Fahrzeug angeschafft. Es handelt sich um einen VW Touran mit dem Kennzeichen F-UT 421. Das Fahrzeug ist ein Neuwagen Baujahr 2016, der Kilometerstand beträgt 120 km. Es ist das erste Fahrzeug der neuen Fahrzeugklasse Van. Für Fahrzeuge dieser Klasse fallen pro Minute 0,15 € und pro gefahrenen Kilometer 0,25 € an. Geben Sie die SQL-Anweisungen an, mit denen diese Informationen in die Datenbank eingetragen werden können.

(3 BE)

2.3 Nachdem Sie das vorläufige ER-Modell implementiert haben, kommen Ihnen Zweifel, ob das entstandene Relationenmodell den Anforderungen der 3. Normalform genügt.
Untersuchen Sie das Modell auf die Einhaltung der 3. Normalform. Geben Sie die gegenüber Ihrer Lösung aus Teilaufgabe 2.1 veränderten bzw. neu entstandenen Relationen in gleicher Schreibweise an.

Hinweis: Es müssen nur die Relationen angeben werden, die von Änderungen betroffen sind.

(5 BE)

2.4 Das vorläufige ER-Modell (Material 7) soll um neue Anforderungen erweitert werden.
Folgende Anforderungen sind aufzunehmen:

- Das Carsharing-Unternehmen verfügt über mehrere Filialen, für die eine eindeutig identifizierende Nummer sowie Anschrift, Telefon- und Faxnummer erfasst werden sollen.
- Ein Mitglied wird durch eine Filiale betreut. Eine Filiale betreut mehrere Mitglieder.
- Die Filialen verwalten mehrere Standorte, an denen die Fahrzeuge abgestellt sind. Die Standorte haben eine Standortbezeichnung, z. B. Hauptbahnhof. Es muss feststellbar sein, an welchem Standort ein Fahrzeug zu finden ist.
- Einem Fahrzeug ist genau ein RFID-Chip zugeordnet. Der RFID-Chip ist durch den RFID-Code eindeutig zu identifizieren und arbeitet auf einer bestimmten Funkfrequenz.
- Abrechnungen können für mehrere Buchungen gemeinsam erstellt werden. Eine Abrechnung muss sich jedoch auf mindestens eine Buchung beziehen.
- Abrechnungen beinhalten Abrechnungsnummer und Datum.
- Für jede Buchung werden der Anfangs- und Endkilometerstand festgehalten. Der Abrechnungsbetrag einer jeden Buchung ergibt sich aus der Nutzungsdauer und den gefahrenen Kilometern. Der Abrechnungsendbetrag ergibt sich aus der Summe der Einzelbeträge der Buchungen.

Entwickeln und zeichnen Sie das entsprechende ERM mit Attributen in [min,max]-Notation.

Hinweis: Es müssen nur die Bestandteile des ERM dargestellt werden, die von der Erweiterung betroffen sind.

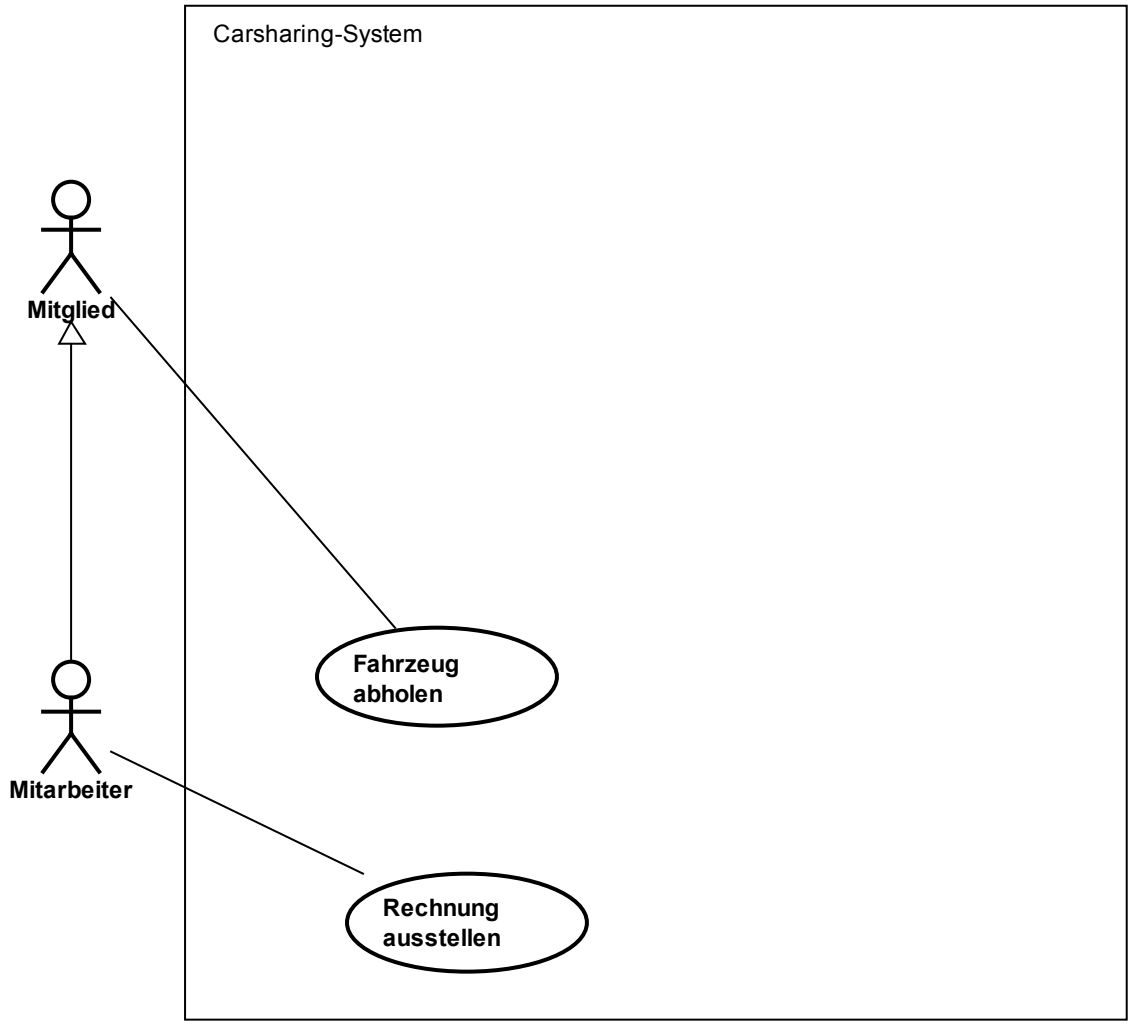
(10 BE)

2.5 Das Carsharing-Unternehmen unterscheidet die Fahrzeuge in Pkws und Transporter. Für Pkws werden neben den zuvor genannten Merkmalen zusätzlich die Anzahl der Sitzplätze, für Transporter zusätzlich deren Ladevolumen gespeichert.
Pkw und Transporter sind Spezialisierungen des Entitätstyps Fahrzeug (*is-a*-Beziehung). Es gibt verschiedene Möglichkeiten diese auf Tabellen abzubilden.
Zeigen Sie zwei Möglichkeiten und diskutieren Sie Vor- und Nachteile der beiden Lösungen.

(5 BE)

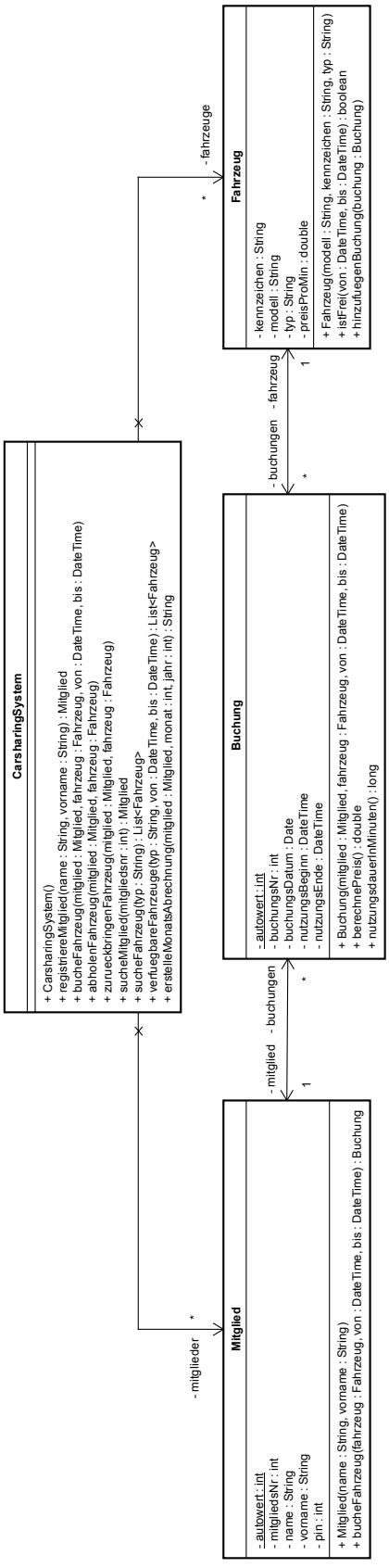
Material 1

UML-Anwendungsfalldiagramm für das Carsharing-System



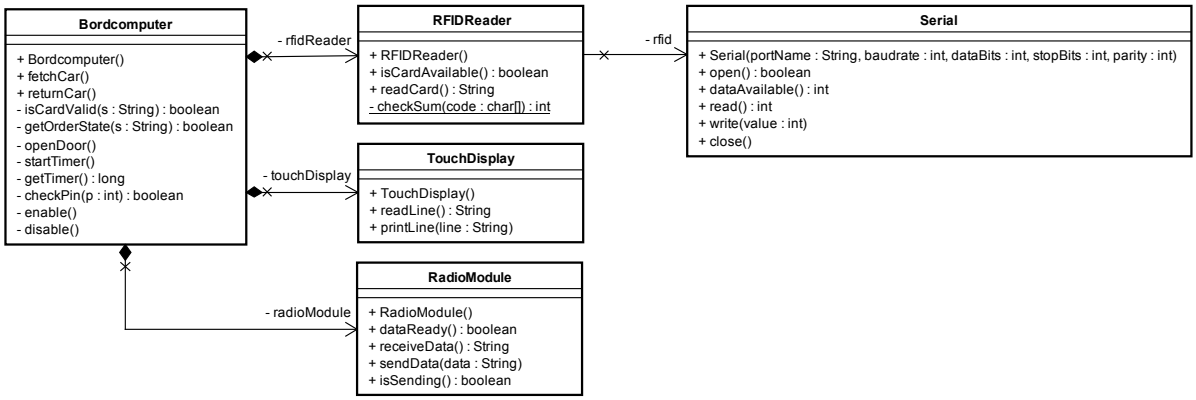
Material 2

UML-Klassendiagramm für das Carsharing-System



Material 3

Klassendiagramm Bordcomputer



Hinweise zum Klassendiagramm:

Methoden der Klasse `Bordcomputer`

- `Bordcomputer()` erzeugt ein `Bordcomputer`-Objekt.
- `fetchCar()` die Methode steuert den Ablauf zur Abholung eines Fahrzeugs.
- `returnCar()` die Methode steuert den Ablauf zur Rückgabe eines Fahrzeugs
- `isCardValid(s: String): boolean` prüft, ob die Mitgliedskarte gültig ist.
- `getOrderState(s: String): boolean` prüft, ob eine Buchung für das Fahrzeug erfolgt ist.
- `openDoor()` öffnet die Zentralverriegelung.
- `startTimer()` aktiviert die Zeitüberwachung.
- `getTimer()` liefert die seit der Aktivierung der Zeitüberwachung abgelaufene Zeit in Sekunden.
- `checkPin(s: String): boolean` überprüft die Korrektheit der PIN.
- `enable()` aktiviert die Wegfahrsperre.
- `disable()` deaktiviert die Wegfahrsperre.

Methoden der Klasse `RFIDReader`

- `RFIDReader()` erzeugt ein `RFIDReader`-Objekt und initialisiert und öffnet ein serielles Schnittstellen-Objekt zum Kartenleser.
- `isCardAvailable(): boolean` prüft, ob eine Mitgliedkarte erkannt wurde.
- `readCard(): String` liefert die Kartendaten als String.
- `checkSum(code: char[]): int` ermittelt zu einer Folge von Datenbytes eine Prüfsumme.

Methoden der Klasse `TouchDisplay`

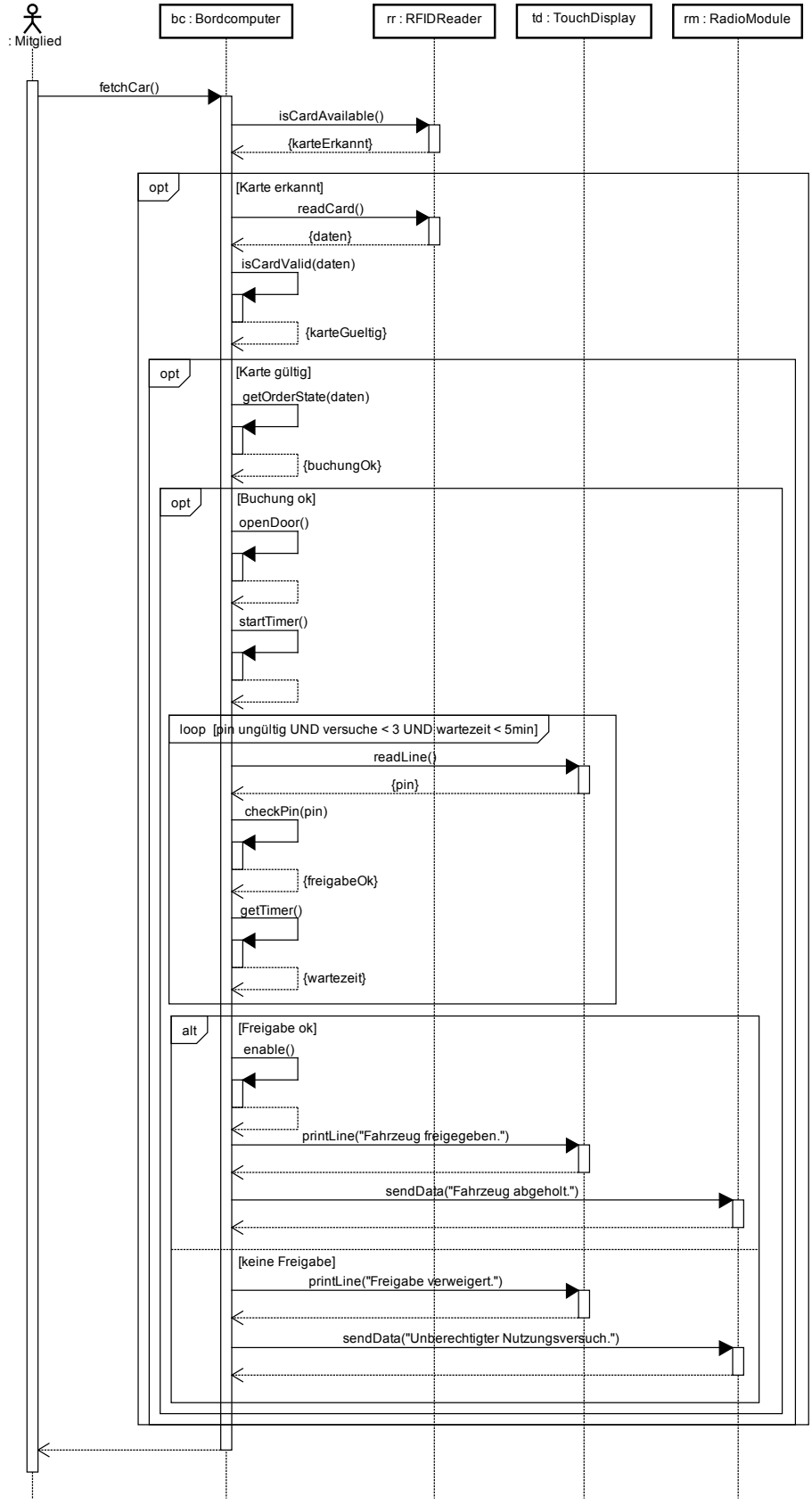
- `TouchDisplay()` erzeugt ein `TouchDisplay`-Objekt.
- `readLine()` liefert eine auf dem Touchdisplay erfolgte Texteingabe.
- `printLine(s: String)` ermöglicht eine Textausgabe auf dem Touchdisplay.

Methoden der Klasse `RadioModule`

- `RadioModule()` erzeugt ein `RadioModule`-Objekt.
- `dataReady(): boolean` prüft, ob Daten zum Lesen bereitsehen.
- `receiveData(): String` liest Daten vom Funkmodul und liefert sie als String zurück.
- `sendData(s: String)` sendet Daten als String über das Funkmodul.
- `isSending(): boolean` prüft, ob das Funkmodul gerade sendet.

Material 4

UML-Sequenzdiagramm der Methode `fetchCar()` der Klasse `Bordcomputer`



Material 5:

TTL Interface RS232 Daten-Ausgabeformat (RDM630 Spezifikation)

1. Übertragungsparameter: 9600 Baud, 8 Datenbits, 1 Stoppbit und kein Paritätsbit.
2. Steuerzeichen: STX (02h) und ETX (03h)
3. Prüfsumme: 10 Daten-Bytes mittels XOR-Operation verknüpft

Übertragungsrahmen:

STX	10 Datenbytes (ASCII-Format)	Prüfsumme	ETX

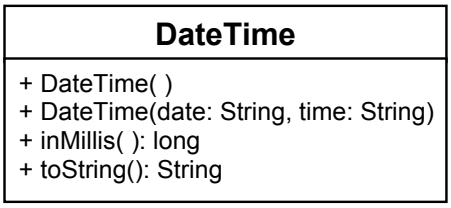
Beispiel:

Kartennummer: 62E3086CED

	Start	Daten										Prüf	Ende
Position	1	2	3	4	5	6	7	8	9	10	11	12	13
Zeichen	STX	'6'	'2'	'E'	'3'	'0'	'8'	'6'	'C'	'E'	'D'		ETX
Zeichencode	02h	36h	32h	45h	33h	30h	38h	36h	43h	45h	44h		03h

Material 6

Die Klasse DateTime



Ein Exemplar der Klasse `DateTime` repräsentiert ein bestimmtes Datum einschließlich einer Zeitangabe.

Kurzbeschreibung der Klasse `DateTime`:

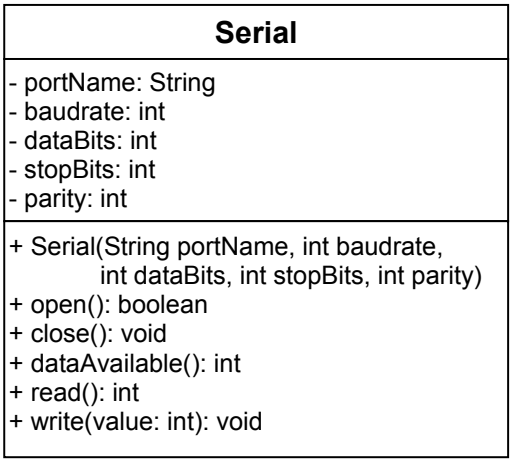
`DateTime()`
erzeugt ein `DateTime`-Objekt mit dem aktuellen Systemdatum und der aktuellen Systemzeit.

`Date(date: String, time: String)`
erzeugt ein `Date`-Objekt aus einem `date`-String mit dem Format `"dd.mm.yyyy"` und einem `time`-String mit dem Format `"hh.mm.ss"`

`inMillis(): long`
liefert die vergangenen Millisekunden relativ zum 1.1.1970.

`toString(): String`
liefert eine String-Repräsentanz des Date-Objekts im Format `"dd.mm.yyyy hh:mm:ss"`.

Die Klasse Serial



Ein Exemplar der Klasse `Serial` ermöglicht die Kommunikation über die serielle Schnittstelle

Kurzbeschreibung der Klasse `Serial`

`Serial(…)`
der Konstruktor initialisiert die serielle Schnittstelle ohne sie zu öffnen.

`portName`:	Name des Ports
`baud`:	Baudrate
`dataBits`:	Datenbitanzahl
`stopBits`:	Stopbitanzahl
`parity`:	Parität

`open(): boolean`
öffnet die serielle Schnittstelle; liefert `true`, wenn die Schnittstelle verwendbar ist.

`close()`
schließt die serielle Schnittstelle; die Schnittstelle ist dann solange nicht mehr verwendbar, bis sie wieder geöffnet wird.

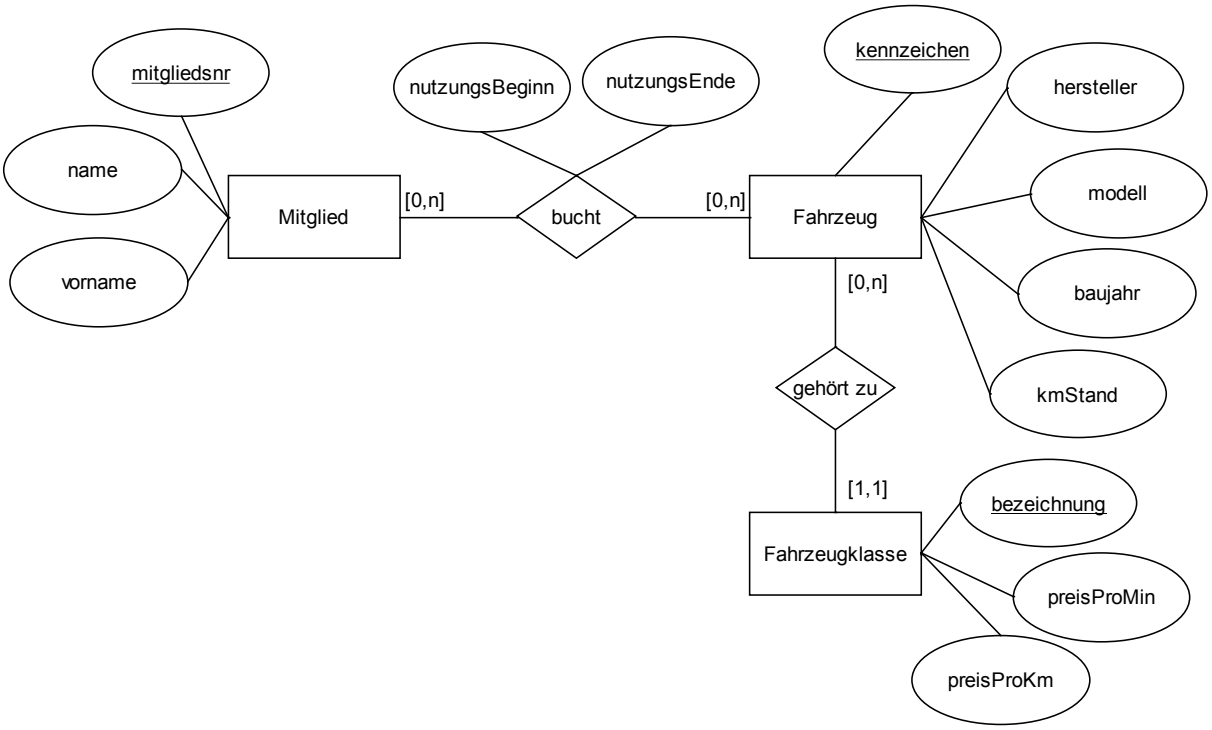
`dataAvailable(): int`
liefert die Anzahl der Bytes, die von der seriellen Schnittstelle gelesen werden können.

`read(): int`
liest ein Byte (0..255) von der serielle Schnittstelle, bzw. –1, wenn die Schnittstelle nicht geöffnet ist. Die Methode blockiert, bis ein Byte verfügbar ist.

`write(value: int)`
schreibt ein Byte auf die serielle Schnittstelle; ist die Schnittstelle nicht geöffnet geschieht nichts.

Material 7

ER-Modell



Hinweis: Die Bezeichnungen von Fahrzeugklassen sind z.B. Mini, Klein, Mittel, Van.