

Objektorientierte Softwareentwicklung (OOSE)**Aufgaben**

- 2 Die Geschäftsprozesse der Ferienhausverwaltung sollen zukünftig digital ausgeführt werden. Hierfür soll eine eigene Anwendung erstellt werden.

Hinweis:

Die Klasse `List` aus Material 2 kann zum Bearbeiten der Aufgaben verwendet werden.

- 2.1 Es liegt folgender erster Entwurf für ein Klassendiagramm vor (Material 3). Bei diesem Entwurf ist es aufwendig, mehrere Häuser zu verwalten.

Entwickeln Sie ein Klassendiagramm, mit dessen Hilfe neue Häuser hinzugefügt und verwaltet werden können.

(6 BE)

- 2.2 In der Ferienanlage, die an einen See angrenzt, werden Paddelboote verliehen. Der Verleih soll mithilfe einer Anwendung ermöglicht werden.

Für einen Teil der Anwendung liegt ein Klassendiagramm vor (Material 4).

- 2.2.1 Überführen Sie die Attribute sowie den Konstruktor der Klasse `Gast` in eine im Unterricht verwendete objektorientierte Programmiersprache.

(5 BE)

- 2.2.2 In der Methode `generiereBeispieldaten():void` der Klasse `Buchungsverwaltung` werden zwei Objekte der Klasse `Gast` erzeugt und der `gastliste` hinzugefügt:

- Merve Mueller, `mm@hessen.de`
- Anna Schmidt, `schmidt@hessen.de`

Überführen Sie die Methode `generiereBeispieldaten():void` in eine im Unterricht verwendete objektorientierte Programmiersprache.

(4 BE)

- 2.2.3 Die Methode `getAnzahlBuchungen():int` der Klasse `Buchungsverwaltung` gibt die Anzahl aller Buchungen zurück.

Implementieren Sie die Methode `getAnzahlBuchungen():int`.

(5 BE)

- 2.2.4 Die Methode `getGaesteOhneBuchung():List<Gast>` liefert eine Liste mit Gästen zurück, die noch keine Buchungen vorgenommen haben.

Implementieren Sie die Methode `getGaesteOhneBuchung()`.

(5 BE)

- 2.2.5 Die Methode `berechneDauerAllerBuchungen() : long` gibt die Dauer aller Buchungen der Paddelboote in Minuten zurück. In einer Buchung wird bei den Attributen der Anfangs- (von) und Endzeitpunkt (bis) in Form eines Timestamps (Millisekunden seit dem Jahr 1970) gespeichert.

Implementieren Sie die Methode.

Hinweis:

1000 Millisekunden entsprechen einer Sekunde.

(5 BE)

- 2.2.6 Durch ein fehlerhaftes Verfahren wurden in dem Programm Gäste doppelt angelegt. Diese Duplikate besitzen die gleiche E-Mail-Adresse. Es soll deshalb die Methode `loescheDuplikate() : void` geschrieben werden. In dieser werden Duplikate gelöscht und die Buchungen in dem verbleibenden Gast-Objekt zusammengeführt.

Implementieren Sie die Methode `loescheDuplikate() : void`.

(6 BE)

- 2.3 Ein Teammitglied hat folgenden Pseudoquelltext für den Verleih erstellt:

```
01 | String quelle = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
02 | String pw = ""
03 | for(i = 0; i < 5; i++){
04 |     int pos = zufallszahl(quelle.laenge())
05 |     String ziffer = quelle[pos]
06 |     pw = pw + ziffer
07 | }
```

Erläutern Sie die Intention hinter der Anwendung.

(4 BE)

Material 2

Die Container-Klasse `List`

Mithilfe der Klasse `List` können Objekte im Arbeitsspeicher in einer linearen Liste verwaltet werden.

`List` repräsentiert eine generische Liste mit Elementen des Typs `T` in festgelegter Reihenfolge, auf die sowohl wahlfrei als auch sequenziell zugegriffen werden kann.

Kurzbeschreibung der Klasse `List`:

`add(obj: T)`

hängt das Objekt `obj` vom Typ `T` am Ende der Liste an.

`contains(obj: T): boolean`

liefert `true`, wenn das Objekt `obj` in der Liste enthalten ist, sonst `false`.

`get(index: int): T`

liefert das Listenelement an der Position `index` zurück bzw. `null`, falls `index` negativ oder größer gleich der Anzahl der momentan enthaltenen Elemente ist.

`remove(index: int): T`

entfernt das Listenelement an der Position `index`. Liefert das entfernte Element zurück bzw. `null`, falls `index` negativ oder größer gleich der Anzahl der momentan enthaltenen Elemente ist.

Nachfolgende Elemente rücken auf.

`remove(obj: T): boolean`

entfernt das Objekt `obj` aus der Liste. Falls `obj` mehrmals in der Liste enthalten ist, wird nur das erste Vorkommen entfernt. Der Rückgabewert ist `true`, falls das Objekt gefunden und entfernt wurde, sonst `false`. Nachfolgende Elemente rücken auf.

`size(): int`

gibt die Anzahl der Elemente in der Liste zurück.

List<T>
+ List<T>() + add(obj: T) + contains(obj: T): boolean + get(index: int): T + remove(index: int): T + remove(obj: T): boolean + size(): int

Material 3

Die Klassenkarte der Klasse Ferienhausverwaltung

Ferienhausverwaltung
+haus1: String +haus1Preis: double +haus2: String +haus2Preis: double +haus3: String +haus3Preis: double +haus4: String
+Ferienhausverwaltung() +getHaus1():String +setHaus1(String):void +getHaus1Preis():double +setHaus1Preis(double):void +getHaus2():String +setHaus2(String):void +getHaus2Preis():double +setHaus2Preis(double):void +getHaus3():String +setHaus3(String):void +getHaus3Preis():double +setHaus3Preis(double):void +getHaus4():String +setHaus4(String):void

Material 4

Klassendiagramm

