

PGdP Woche #6 – Objekte und Datenkapselung

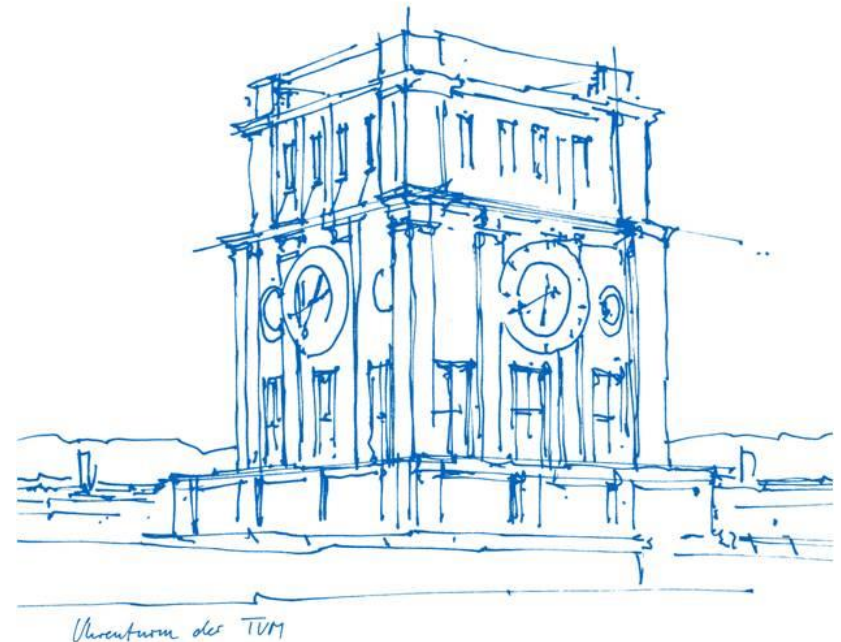
Sebastian Oßner – ossner@in.tum.de

Technische Universität München

Garching, 25. November 2019

Ablauf:

1. P01 – Cäsar Chiffre
2. P02 – Parkhaus
3. P03 – Datenkapselung
4. P04 – Doppelt verkettete Liste



Cäsar Chiffre

Encrypt:

“Please don’t stab me“, key = 5

-> “Uqjfxj ist’y xyfg rj“

“Et tu, Brute?“, key = 41

-> “Ti ij, Qgjit?“

Zu schreibende Methoden:

```
// wandelt String zu char array um
```

```
public static char[] toArray(String input)
```

```
// verschlüsselt char Array mit shift um key
```

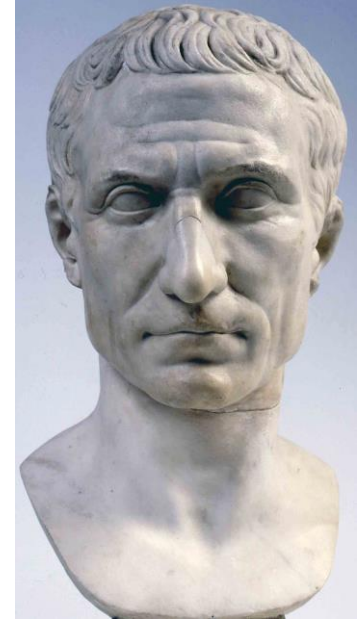
```
public static String encrypt(char[] input, int key)
```

Constraints:

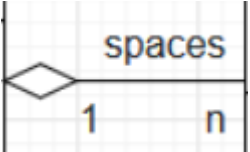
Nur String.length() und String.charAt(int index)

Großbuchstaben bleiben groß

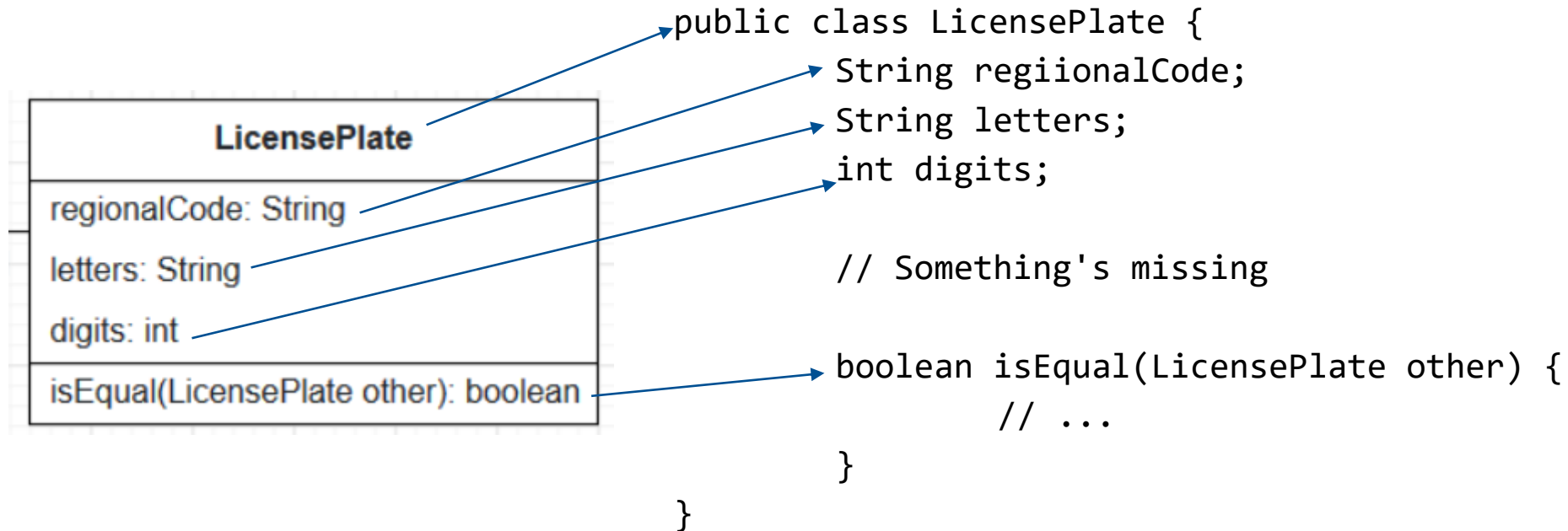
nicht-Buchstaben werden nicht verändert



P02 – Parkhaus (Objekte, Objekte, Objekte)

- UML Kästen sind Objekte
 - Objekte haben Attribute und Methoden
 - Attribute und Methoden können wiederum Objekte sein (bzw. zurückgeben)
- Aggregation (Verbindung zweier Objekte mit weißem Diamanten)
 - Das eine Objekt besteht aus mehreren der anderen Objekte, kann aber auch ohne sie existieren (Parkplatz ohne Autos, Auto ohne Nummernschild)

P02 – Parkhaus II



P03 – Datenkapselung

public: Jede Methode in einer anderen Klasse kann darauf zugreifen/verändern/lesen

private: Keine Methode in einer *anderen* Klasse kann darauf zugreifen

(package-private): Nur Methoden in Klassen in dem selben Ordner können darauf zugreifen

Getter:

public Methode, um **private** Attribute aus einer anderen Klasse zu **lesen**


Setter:


public Methode, um **private** Attribute aus einer anderen Klasse zu **schreiben**

Doppelt verkettete Liste

- Objektorientierte Datenstruktur, in der Informationen minimalistisch gespeichert werden

Ein Element enthält also nur Informationen über seinen Vor-/Nachgänger in der Liste und das Datenobjekt, dass in der Zelle gespeichert werden soll. In der Liste an sich passiert das meiste

 IntDoubleList
- head: IntDoubleListElement - tail: IntDoubleListElement
+ append(...) + size(): int + get(...): int + remove(...) + sum(): int ...

 IntDoubleListElement
- info: int + next: IntDoubleListElement + prev: IntDoubleListElement
+ isEqual(...): boolean