Programmieren 1
Selbststudium Semesterwoche 11

Kapitel 7.1 bis 7.4

1.
zu bearbeitende Aufgaben: 7.1, 7.2, 7.4 (Bauen Sie zwei zusätzliche Räume ein.) und 7.5

2.
Was versteht man unter Kopplung? Wie soll die Kopplung sein?
→ Unter Kopplung von Klassen versteht man die direkte Abhängigkeit, welche Klassen untereinander haben können. Gute Programmierer versuchen, Klassen so unabhängig wie möglich von einander zu machen. Dies nennt man dann „eine lose Kopplung".
→ Gründe dafür gibt es einige: Beispielsweise können Klassen besser erweitert werden ohne dass dabei diverse andere Klassen auch angepasst werden müssen.
http://de.wikipedia.org/wiki/Lose_Kopplung

3.
Was versteht man unter Kohäsion? Wie soll die Kohäsion sein?
→ In der objektorientierten Programmierung beschreibt Kohäsion, wie gut eine Programmeinheit eine logische Aufgabe oder Einheit abbildet. In einem System mit starker Kohäsion ist jede Programmeinheit (eine Methode, eine Klasse oder ein Modul) verantwortlich für genau eine wohldefinierte Aufgabe oder Einheit.
http://de.wikipedia.org/wiki/Koh%C3%A4sion_(Informatik)

4.
Was ist die Problematik bei Code-Duplikation?
→ Die Wartung wird extrem aufwendig: Änderungen müssen an mehreren/vielen Codestellen durchgeführt werden.

5.
Von was ist Code-Duplikation häufig ein Symptom (schlechter Kopplung oder schlechter Kohäsion)?
Kapitel 7.5 bis 7.11
→ Code-Duplikationen sind Symptome von schlechter Kohäsion.
→ Grund: Es gibt mehrere Methoden, welche dieselbe Aufgabe erfüllen bzw. dasselbe Problem lösen.

6.
zu bearbeitende Aufgaben: 7.6 bis 7.8, 7.11, 7.14, 7.16 und 7.17

7.
Die Klasse Room verwaltet neu benachbarte Räume mit Hilfe einer HashMap. Allerdings mussten dazu an zig Stellen Anpassungen gemacht werden! Ist dies ein Indiz für zu starke oder zu lose Kopplung?
→ Das ist wohl ein Indiz für eine zu starke Kopplung.
http://de.wikipedia.org/wiki/Lose_Kopplung

8.
Was ist Information Hiding bzw. Datenkapselung und was bewirkt dieses grundlegende Prinzip?
→ Das Konzept des „Information Hidings" basiert auf dem Grundsatz, Informationen über ein entsprechendes Objekt nur über definierte Schnittstellen zur Verfügung zu stellen.
→ Ein Entwickler muss sich nicht um den Inhalt einer Klasse kümmern. Er kann die Klasse über die bereitgestellten Schnittstellen nutzen.
→ „Information Hiding" ist das Prinzip – „Encapsulation" ist die Technik.
http://en.wikipedia.org/wiki/Information_hiding

9.
Was meint das Konzept "localizing change"?
→ Wenn eine Änderung am Code vorgenommen werden muss, soll diese so lokal wie möglich bleiben.
→ Verbesserte Wartbarkeit.
→ Verbesserte Zusammenarbeit beim gemeinsamen Entwickeln von Software.

10.
Inwiefern unterscheiden sich explizite und implizite Kopplung?
→ ???

11.
Machen kohäsive Methoden auch Sinn?
→ Ja.

12.
Welches sind die beiden Hauptvorteile von starker Kohäsion?
→ Kohäsive Methode erfüllt eine genau definierte Aufgabe.
→ Verhindert Code-Duplikationen und verbessert so die Wartbarkeit der Software.

Kapitel 7.12 und 7.14

13.
zu bearbeitende Aufgabe: 7.27
14.
Was ist der Auslöser für Refactoring?
→ Erweiterungen

15.
Beschreiben Sie die Vorgehensweise bzw. die Schritte beim Refactoring.
→ Beim Refactoring wird der Quelltext eines Computerprogramms umgestaltet, wobei die tatsächliche Programmfunktion unverändert bleiben soll. Die Umgestaltung des Quelltextes erfolgt meist nach folgenden Gesichtspunkten:

- Lesbarkeit
- Übersichtlichkeit
- Verständlichkeit
- Erweiterbarkeit
- Vermeidung von Redundanz
- Testbarkeit

http://de.wikipedia.org/wiki/Refactoring

16.
Wann ist eine Methode zu lang?
→ Wenn sich die Methode aus mehr als 3 Wörter zusammensetzt. (Das ist allerdings eine sehr subjektive Betrachtung…)

17.
Wann ist eine Klasse zu komplex?
→ Wenn selbst der Urheber deren Funktionalitäten nicht erklären kann.

18.
Illustrieren Sie den Unterschied zwischen stabilem und instabilem Sortieren an einem einfachen Beispiel.
→ Ein stabiles Sortierverfahren ist ein Sortieralgorithmus, der die Reihenfolge der Datensätze, deren Sortierschlüssel gleich sind, bewahrt:

Stabiles Sortierverfahren nach Zahlen:

1 Anton		1 Anton
4 Karl		1 Paul
3 Otto		3 Otto
5 Bernd	→	3 Herbert
3 Herbert		4 Karl
8 Alfred		5 Bernd
1 Paul		8 Alfred

→ Beispiele für stabile Sortierverfahren:
Bubblesort, Countingsort, Cocktailsort, Gnomesort, Insertionsort, Mergesort, Slowsort, Radixsort

Instabiles Sortierverfahren nach Zahlen:

1 Anton	1 Paul		1 Anton		1 Paul		1 Anton
4 Karl	1 Anton		1 Paul		1 Anton		1 Paul
3 Otto	3 Otto		3 Herbert		3 Herbert		3 Otto
5 Bernd →	3 Herbert	oder	3 Otto	oder	3 Otto	oder	3 Herbert
3 Herbert	4 Karl		4 Karl		4 Karl		4 Karl
8 Alfred	5 Bernd		5 Bernd		5 Bernd		5 Bernd
1 Paul	8 Alfred		8 Alfred		8 Alfred		8 Alfred

→ Beispiele für instabile Sortierverfahren:
Binary Tree Sort, Heapsort, Introsort, Quicksort, Shellsort, Smoothsort, Stoogesort

http://de.wikipedia.org/wiki/Stabiles_Sortierverfahren

19.
Wie gross ist der Aufwand bei einfachen/direkten Sortieralgorithmen?
→ Zeitkomplexität = O(n)

20.
Wie gross ist der Aufwand bei höheren Sortieralgorithmen?
→ Zeitkomplexität = $O(n^2)$

21.
Benennen Sie 3 einfache/direkte Sortieralgorithmen.
→ Bubblesort, Insertionsort, Mergesort

22.
Welche der behandelten Sortieralgorithmen arbeiten instabil?
→

23.
Beim Analysieren von Sortieralgorithmen ist wichtig zu wissen, wie man folgende Summe einfach berechnen kann:

$$1+2+3+\ldots+n=\sum_{i=1}^{n} i=?$$

→???

Optional: Kapitel 7.13 (Seite 218 bis 222)

24.
Inwiefern unterscheidet sich ein Klassentyp (class) von einem Aufzählungstyp (enum)?
→ „enum“ ist eine Enumeration eines anderen Datentyps.

25.

Jeder Aufzählungstyp kennt eine Methode values(). Was liefert diese Methode als Rückgabewert?