

23. Juli 2014

Vorname:

Nachname:

Matrikelnr.:

-
- Bearbeitungszeit: 90 Minuten
 - Erlaubte Hilfsmittel: Zwei DIN-A4-Blätter, beidseitig handschriftlich beschrieben.
 - Bitte Studentenausweis schon mal auf den Tisch legen.
 - Bitte Name und Matrikelnummer auf die mitgebrachten leeren Blätter schreiben.

Aufgabe	1..5	6..11	12..16	Σ
mögliche Punktzahl	20	30	20	70
erreichte Punktzahl				

Verschiedenes

Beantworte die folgenden Fragen **kurz**, gerne auch **stichpunktartig**, und **verständlich**:

1. Welche Testebenen haben wir in der Vorlesung unterschieden? Erläutere **kurz** jede dieser Testebenen. (4)
2. Nenne zwei wesentliche Gemeinsamkeiten und zwei wesentliche Unterschiede von/- zwischen Klassen- und Objekt-Diagramm. (4)
3. Nenne drei Vorgehensmodelle. (3 Namen) Erläutere eines davon kurz. (max. 10 Wörter) (4)
4. Nenne drei GRASP-Muster. (3 Namen) Erläutere eines davon kurz. (max. 10 Wörter) (4)
5. Erkläre kurz den Begriff Repräsentationsinvariante. (max. 10 Wörter) (4)

Anforderungsanalyse und Grobentwurf

Es geht um die Anwendung „Wikipedia-App“, die so funktionieren soll:

„Der Benutzer greift über eine App mit grafischer Benutzeroberfläche (indirekt) auf die Wikipedia-Artikel zu. Es kann nach Wikipedia-Artikeln durch Eingabe eines Suchbegriffs gesucht werden. Die Suchanfrage wird mit Hilfe des Wikipedia-Servers bearbeitet. Außerdem können Artikel auch geteilt und verändert werden.“

6. Erstelle ein Anwendungsfall-Diagramm. (3 Anwendungsfälle, 2 Akteure) (4)

Betrachten wir nun lediglich den Anwendungsfall „Artikel suchen“:

7. Erstelle eine textuelle Anwendungsfall-Beschreibung. (gemäß in Vorlesung vorgestellter Anwendungsfallschablone, 3-5 Schritte im Standardblauf und 2-3 alternative Ablaufschritte) (6)
8. Stelle den Standard-Ablauf zusammen mit den alternativen Ablaufschritten als **ein** Aktivitäts- **oder** als **ein** Zustands-Diagramm dar. (4-8 Aktionen bzw. Zustände) (4)
9. Stelle eine beispielhafte Ausprägung des Standardablaufs als Sequenz-Diagramm dar. Falls der Wikipedia-Server an der Interaktion beteiligt ist, soll auch das mit dargestellt werden. (4 - 8 Pfeile) (4)
10. Erstelle einen Grobentwurf in Form eines FMC-Blockdiagramms. Die Verbindungen zum Benutzer und zum Wikipedia-Server sollen auch dargestellt werden. Es genügt, alle Komponenten darzustellen, die für den Anwendungsfall „Artikel suchen“ benötigt werden. (7-10 Agenten [= Rechtecke], 1-3 Speicher [= große, runde Komponenten]) (6)

Die folgenden Aufgaben haben nichts mehr mit der Anwendung „Wikipedia-App“ zu tun.

Systemtest

Betrachten wir jetzt für eine völlig andere Anwendung den Anwendungsfall „Registrierung“. Abbildung 1 zeigt den GUI-Entwurf für ein zugehöriges Fenster. Tabelle 1 zeigt einen Ausschnitt aus der zugehörigen Anwendungsfallbeschreibung.

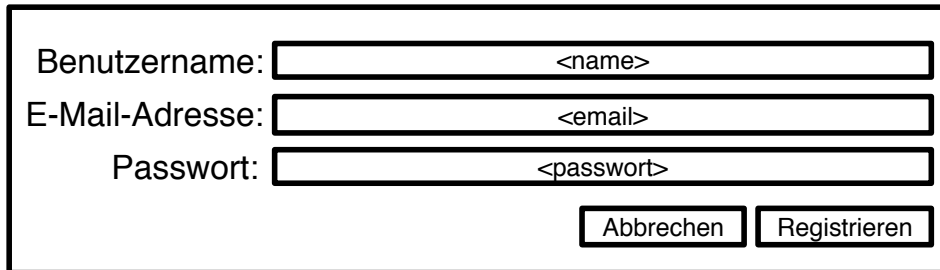


Abbildung 1: Fenster „Registrierung“

Standardablauf	... 3. Der Benutzer gibt ein: Benutzername, E-Mail-Adresse, Passwort. 4. Der Benutzer klickt auf „Registrieren“. 5. Das System überprüft die Eingabe. ...
Alternative Ablaufschritte	... 5a. Benutzername leer. → Fehlermeldung. Zurück zu 3. 5b. Benutzername schon vorhanden. → Fehlermeldung. Zurück zu 3. 5c. E-Mail-Adresse leer. → Fehlermeldung. Zurück zu 3. 5d. E-Mail-Adresse ist keine gültige E-Mail-Adresse. → Fehlermeldung. Zurück zu 3. 5e. Passwort leer. → Fehlermeldung. Zurück zu 3. ...

Tabelle 1: Auszug aus der Anwendungsfallbeschreibung „Registrierung“

11. Liste alle **kombinierten** logischen Testfälle auf, die sich aufgrund einer Black-Box-Betrachtung dieses Ausschnitts der Anwendungsfallbeschreibung ergeben. (Keine erwarteten Ergebnisse.)

(6)

Integrationstest

12. Betrachte die Klassen in Listing 1:

```

1  class KundenDB {
2      Map<Integer,Kunde> db=new HashMap<Integer,Kunde>();
3      void insert(int kundennr, String name) throws IllegalArgumentException {
4          Kunde kunde=new Kunde(kundennr, name);
5          if (!kunde.check()) {throw new IllegalArgumentException();}
6          if (db.get(kundennr)!=null) {throw new IllegalArgumentException();}
7          db.put(kundennr,kunde);
8      }
9  }
10 class Kunde {
11     int kundennr;
12     String name;
13     Kunde(int knr, String n) {kundennr=knr; name=n;}
14     boolean check() {
15         if (kundennr<=0 || name==null) {return false;}
16         if (name.length()==0) {return false;}
17         return true;
18     }
19 }

```

Listing 1: Klassen für Integrationstest

Liste die logischen Testfälle für den Integrationstest von KundenDB und Kunde auf.
(Keine erwarteten Ergebnisse.)

(4)

Feinentwurf & Implementierung

Betrachte den Code in Listing 2:

```

1  public class A {
2      List<B> list=new ArrayList<B>();
3      void add(B b) {list.add(b);}
4      void inform() {for (B b:list) {b.update();}}
5  }
6  public interface B {
7      void update();
8  }
9  public class C implements B {
10     int i=0;
11     public C(A a) {a.add(this);}
12     public void update() {i++;}
13 }

```

Listing 2: Interface B, Klassen A, C

13. Stelle den Code in Listing 2 als Klassendiagramm dar. (4)

14. Kennzeichne im Klassendiagramm alle erkannten Entwurfsmuster (mit jeweils zugehörigen Rollen). (4)

15. Zeichne ein beispielhaftes Objektdiagramm. (3-5 Objekte) (4)

Spezifikation

Betrachte die Methode f in Listing 3:

```
1 public static int f(List<String> liste, String s) {  
2     int n=0;  
3     for (String s1:liste) {  
4         if (s.equals(s1)) {  
5             n++;  
6         }  
7     }  
8     if (n==0) {liste.add(s); return 1}  
9     else {return n;}  
10 }
```

Listing 3: Methode f

16. Spezifiziere die Methode f.

(4)