NAME DER DOZENTEN: Prof. Dr. Joachim Sauer
Prof. Dr. B. Trancón Widemann

NORDAKADEMIE
HOCHSCHULE DER WIRTSCHAFT

Klausur: Modul Softwarequalität (A104)

QUARTAL: II/2021

Name des Prüflings: **Matrikelnummer:** **Zenturie:**

Dauer: 90 Minuten Seiten der Klausur **ohne** Deckblatt: 5 **Datum:** 23.06.2021

Hilfsmittel: NORDAKADEMIE-Taschenrechner, verschiedenfarbige (nicht rote) Stifte.

Bemerkungen:

- *Bitte prüfen Sie zunächst die Klausur auf Vollständigkeit.*
- *Bitte beantworten Sie die Fragen unter Angabe der Fragennummer auf nummererierten, mit Ihrem Namen und Matrikelnummer versehenen Zetteln.*
- *Ihre Lösungen müssen nach Ablauf der Bearbeitungszeit eingescannt und in Moodle hochgeladen werden.*

Es sind 90 Punkte erreichbar.
Zum Bestehen der Klausur sind 45 Punkte ausreichend.

Aufgabe	Erreichbare Punkte	Erreichte Punkte
1	10	
2	9	
3	11	
4	9	
5	14	
6	10	
7	6	
8	11	
9	10	
Summe	90	

Note: _____ Prozentsatz: _____ Ergänzungsprüfung: _____

Datum: _____ Unterschrift: _____

Datum: _____ Unterschrift: _____

Prof. Dr. Joachim Sauer
Prof. Dr. B. Trancón Widemann

NORDAKADEMIE
HOCHSCHULE DER WIRTSCHAFT

23.06.2021

1. **Softwarequalität**

   (1.1) (2 Punkte) Definieren Sie den Begriff „Softwarequalität“.

   (1.2) (4 Punkte) Erläutern Sie, warum man bei der Qualitätsbewertung von Software nicht nur die eigentliche Anwendungssoftware betrachten sollte.

   (1.3) (4 Punkte) Erläutern Sie, warum es betriebswirtschaftlich nicht sinnvoll ist, den Aufwand für Qualität zu maximieren.

2. **Klassifikation von Anforderungen**

   (2.1) (6 Punkte) Erläutern Sie das Kano-Modell der Kundenzufriedenheit anhand einer Skizze.

   (2.2) (3 Punkte) Geben Sie für jede der drei von Kano definierten Arten von Anforderungen ein Beispiel für eine angenommene „Tracing-App für Corona-Infektionsfälle“. Begründen Sie Ihre Wahl.

3. **Qualitätsstandards**

   (3.1) (3 Punkte) Beschreiben Sie den Zusammenhang von Qualitätsmodell, Qualitätsmerkmal und Qualitätsindikator.

   (3.2) (2 Punkte) Welches Modell ist das momentan bedeutendste für die Qualität von Systemen und Software?

   (3.3) (6 Punkte) Nennen und erläutern Sie drei Qualitätsmerkmale aus diesem Modell. Geben Sie Beispiele für Qualitätsindikatoren zur Ermittlung der Merkmale.

4. **Aussagen zum Thema Test**

   Notieren sie für jede Teilaufgabe die Kennbuchstaben der wahren Aussagen. Bei jeder Teilaufgabe können keine, eine, mehrere oder alle Aussagen wahr sein. Sie erhalten nur dann einen Punkt für eine Teilaufgabe, wenn alle wahren Aussagen dieser Teilaufgabe genannt und alle unwahren Aussagen nicht genannt sind.

   (4.1) (1 Punkt) Im Teufelsquadrat werden u. a. . . .

   A: Kosten,
   B: Zeit,
   C: Codeumfang,
   D: Funktionalität,
   E: und Qualität

   . . . gegenübergestellt.

   (4.2) (1 Punkt) Die Norm ISO-25010 . . .

   A: . . . spezifiziert ausschließlich Merkmale für die Beschreibung der Funktionalität einer Software.
   B: . . . spezifiziert ausschließlich Merkmale, die den Entwicklungsprozesses einer Software charakterisieren.
   C: . . . / definiert 10 Qualitätsmerkmale von Software.

(4.3) (1 Punkt) Die Norm ISO-25010 definiert u. a. die folgenden Qualitätsmerkmale von Software:

A: Kompatibilität

B: Wartbarkeit

C: Sicherheit

(4.4) (1 Punkt) Integrationstests überprüfen, ob ...

A: ... ein System nach Integration in die Infrastruktur des Kunden alle Anforderungen erfüllt.

B: ... alle integralen Bestandteile eines Systems vorhanden sind.

C: ... Komponenten erwartungsgemäß zusammenarbeiten.

(4.5) (1 Punkt) Bei der Verifizierung wird geprüft, ...

A: ... ob ein System die Anforderungen bzgl. der beabsichtigen Nutzung erfüllt, also nützlich für den Anwender ist.

B: ... ob ein System die Vorgaben erfüllt, also korrekt arbeitet.

C: ... welche Teile eines Systems testbar sind.

(4.6) (1 Punkt) Fehlerzustände ...

A: ... äußern sich durch Fehlerwirkungen.

B: ... sind z. B. fehlerhafte Ausgaben eines Programmes zur Laufzeit.

C: ... werden durch Fehlhandlungen (von Entwicklern und Anwendern) verursacht.

(4.7) (1 Punkt) Testautomatisierung ...

A: ... ermöglicht Zeit- und Kostenersparnisse durch automatisierten Testablauf.

B: ... führt zu langfristigen Aufbau technischer Schulden, da der Testcode gewartet werden muss.

C: ... ermöglicht Reproduzierbarkeit der Ergebnisse durch die Implementierung der Tests als Code.

(4.8) (1 Punkt) Technische Schulden ...

A: ... sollten stetig abgebaut werden, da Zins und Zinseszins die Schulden stetig erhöhen.

B: ... entstehen bei der Bank bei Beschaffung neuer IT-Technik.

C: ... entstehen u. a. bei Code-Änderungen, die aufgrund eines kurzfristigen Software-Releases nicht durch ausreichend Tests abgedeckt werden.

(4.9) (1 Punkt) Die SQALE-Methodik ...

A: ... ermöglicht die automatisierte Testcode-Erstellung anhand von Anforderungsspezifikationen.

B: ... ermöglicht eine Bewertung von technischen Schulden in Softwaresystemen auf der Basis definierter Regeln.

C: ... ermöglicht die Verfolgung der Qualität von Software anhand von Kennzahlen.

5. Whitebox-Testing

Folgende Java-Methode berechnet eine Zeile des Pascalschen Dreiecks, die man als Parameter `row` übergeben kann.

Listing 1: Pascalsches Dreieck

```
1  ArrayList PascalTri(int row){
2    ArrayList triangle = new ArrayList();
3    triangle.add(0);
4    triangle.add(1);
5    triangle.add(0);
6    if (row > 1){
7      for (int i=0; i<row; i++){
8        ArrayList newTri = new ArrayList();
9        newTri.add(0);
10       for (int j=0; j<triangle.size()-1; j++){
11         int a=((int)triangle.get(j)
12            + (int)triangle.get(j+1));
13         newTri.add(a);
14       }
15       newTri.add(0);
16       triangle = newTri;
17     }
18   }
19   return triangle;
20 }
```

(5.1) (5 Punkte) Geben Sie den Kontrollflussgraphen zu dieser Methode an.

(5.2) (2 Punkte) Wie hilft Ihnen der Kontrollflussgraph zur Definition von Testfällen für eine 100%-ige Anweisungsüberdeckung? Geben Sie eine Menge minimaler Testfälle an, um diesen Grad der Anweisungsüberdeckung zu erreichen. Geben Sie bitte pro Testfall nur den Übergabeparameter `row` an. Den erwarteten Rückgabewert `triangle` als Liste von Integern müssen Sie nicht angeben.

(5.3) (3 Punkte) Wie hilft Ihnen der Kontrollflussgraph zur Definition von Testfällen für eine 100%-ige Entscheidungsüberdeckung? Geben Sie eine Menge minimaler Testfälle an, um diesen Grad der Überdeckung zu erreichen. Auch hier gilt, dass die Angabe der Eingabewerte für `row` ausreicht.

(5.4) (4 Punkte) Warum ist die Definition von Testfällen für eine 100%-ige Pfadüberdeckung in diesem Beispiel nicht möglich?

6. **Blackbox-Testing**

   Sie sind Testentwickler in einem Team, das eine Funktion testen soll, die das Minimum zweier nicht-negativer Zahlen berechnet.

   Die Funktion besitzt die folgende Schnittstelle:

   ```
   int min(int a, int b);
   ```

   Die Funktion gibt die kleinere der beiden Zahlen `a` und `b` zurück oder wirft eine `NumberIsNegativeException` im Fall, dass a oder b negativ sind.

   (6.1) (5 Punkte) Erläutern Sie kurz die Äquivalenzklassenbildung zur Erstellung von Testfällen. Welche Äquivalenzklassen gibt es in diesem Beispiel?

   (6.2) (5 Punkte) Grenzen Sie die Grenzwertanalyse kurz vom Verfahren der Äquivalenzklassenbildung ab. Definieren Sie Testfälle für eine erfolgreiche Grenzwertanalyse. Jeder Testfall besteht aus Eingabewerten und der erwarteten Ausgabe.

7. **Integrationstests**

   (7.1) (3 Punkte) Welche Fehlertypen können vor allem mit Integrationstests erkannt werden? Erläutern Sie diese kurz.

   (7.2) (3 Punkte) Nennen Sie drei unterschiedliche Strategien für Integrationstests und grenzen Sie sie kurz voneinander ab.

8. **Resilient Software Design**

   Eine Festplatte des Herstellers HorribleDD laufe im Durchschnitt 15 000 Stunden, bis sie einen Defekt aufweist, der alle Daten auf der Festplatte unbrauchbar macht. Ein Serversystem mit so einer Festplatte fällt statistisch nach diesen 15 000 Stunden also vollständig aus, bis der Administrator die Festplatte getauscht und die Daten wieder eingespielt hat. Dieser manuelle Vorgang dauert ca. 3 Stunden.

   (8.1) (5 Punkte) Wie lauten MTTF und MTTR des Serversystems? Was bedeuten diese Kennzahlen? Berechnen Sie dessen Verfügbarkeit in Prozent (runden Sie den Prozentwert auf zwei Stellen nach dem Komma). Erläutern Sie anhand eines der vier Grundprinzipien der Resilienz, wie dieses dabei helfen kann, die Verfügbarkeit dieses Serversystems zu erhöhen.

   (8.2) (6 Punkte) Erläutern Sie die Bedeutung von Resilienz für die Softwareentwicklung. Gehen Sie dabei insbesondere auf Möglichkeiten zur Erhöhung der Verfügbarkeit einer Anwendung ein.

9. (10 Punkte) **Bewertung eines Softwareproblems**

Im November 2019 berichtete heise online „Tödlicher Crash mit autonomem Auto: Fußgänger auf Fahrbahn nicht vorgesehen“:

> *In einem Unfall mit einem Roboterauto von Uber kam 2018 eine Frau ums Leben. Das Auto konnte verkehrswidrig die Straße überquerende Personen nicht erkennen.* [...] *Das geht aus einem nun veröffentlichten Report der US-Behörde für Transportsicherheit (NTSB) hervor, der eine Reihe von Ergebnissen der Untersuchungen zusammenfasst.* [...]
>
> *Dem Bericht zufolge fuhr der autonom fahrende, umgebaute Volvo-SUV von Uber 70 km/h, als die Frau 5,6 Sekunden vor dem Crash erstmals erfasst wurde. Sie wurde* [...] *nur als „Vehikel“ klassifiziert, dem keine Bewegungsrichtung zugeschrieben wurde. Innerhalb der nächsten Sekunden wurde diese Klassifikation andauernd geändert.* [...] *Mit jeder neuen Klassifikation wurden die zuvor registrierten Ortsangaben zurückgesetzt. Das Roboterauto meinte deswegen, andauernd ein neues stationäres „Vehikel“, „unbekanntes Objekt“ oder „Fahrrad“ zu erkennen. Die Bewegung in Richtung der Autospur wurde sekundenlang nicht vorhergesehen.*
>
> *Erst 1,5 Sekunden vor dem Crash – und immer noch bei 70 km/h – wurde ein „unbekanntes Objekt“ erfasst, das sich „teilweise in die Spur des SUV“ bewegte. Die Algorithmen berechneten deswegen ein Ausweichmanöver, heißt es bei der NTSB. Genau 1,2 Sekunden vor dem Crash erkannte das System dann ein Fahrrad, das sich „voll auf dem Weg in die Spur“ befände, das Ausweichmanöver war nicht mehr möglich.* [...] *Wenn das System solch ein gefährliche Situation erkannte, pausierte es für eine Sekunde, um dem im Auto sitzenden Sicherheitsfahrer Zeit für ein Eingreifen zu geben. Ein Alarm war aber nicht vorgesehen. So sollten ungewollte Konsequenzen eines Fehlalarms vermieden werden. Der in dem Volvo eingebaute Notbrems-Assistent war abgeschaltet.*
>
> *0,2 Sekunden vor der Kollision endete die einsekündige Pause, in der die anwesende Sicherheitsfahrerin nichts unternommen hatte, sie hatte die Straße nicht im Blick. Die Software war nun so programmiert, dass sie nur voll in die Bremsen steigt, wenn die Kollision auf diesem Weg verhindert werden konnte. Andernfalls war eine akustische Warnung vorgesehen und nur eine graduelle Bremsung. Im konkreten Fall übernahm die Sicherheitsfahrerin in diesem Moment das Steuer und deaktivierte damit die autonome Steuerung. Es kam zum tödlichen Crash und erst 0,7 Sekunden danach, bei einer Geschwindigkeit von immer noch 60 km/h, begann die Sicherheitsfahrerin das Auto abzubremsen.*

Wenden Sie Ihr gesammeltes Wissen über Softwarequalität an und analysieren Sie die Situation.

Was ist schief gegangen, um welche Art von Fehler handelt es sich wohl? Wie hätte das vermieden werden können? Wie sollte die Softwarequalität bei autonom fahrenden Autos in der Zukunft überprüft werden? Ist eine rechtliche Regelung notwendig? Wie könnte diese aussehen?