OTH Regensburg

Fakultät für Informatik und Mathematik

Beispiel für eine Prüfung in MST/WMST

bei Prof. Dr. M. Bulenda

Name, Vorname: ……

Matrikelnummer:……..

Hinweise:

- Schreiben Sie nicht mit Bleistift, rotem oder grünem Stift.
- Verwenden Sie für die Lösungen den dafür vorgesehenen Platz direkt bei den jeweiligen Aufgaben.
- Schreiben Sie leserlich.
- Sollten Sie ein oder mehrere weitere Blätter abgeben wollen, so heften Sie die Blätter zur Klausur. Keine lose eingelegten Blätter!
- Die Klausur umfasst 13 Seiten. Überprüfen Sie, ob Ihnen alle Seiten vorliegen.

Mit Ihrer Unterschrift bestätigen Sie, dass Sie die Hinweise gelesen haben.

Unterschrift: ……

Teil	Erreichbare Punkte	Erreichte Punkte	Note
Teil 1 - Theorie	47		
Teil 2 - Praxis	43		
Summe	90		

Unterschrift Erstprüfer: ……

Ggfs. Unterschrift Zweitprüfer: ……

Teil 1 – Theorie

Aufgabe 1 Kommunikation - Konflikte

Punkte: ____

3 Punkte

Beschreiben Sie die **Grundprinzipien** des Schichtenmodells zur Bearbeitung von Konflikten nach Berg.

Aufgabe 2 Software Qualität

Punkte: ____

5 Punkte

a) Nennen Sie genau drei Software Qualitätsmerkmale nach ISO/IEC 9126-1
3 Punkte

b) Welches der Qualitätsmerkmale nach ISO/IEC 9126-1 muss man mit besonderem Bedacht optimieren und warum?
2 Punkte

Aufgabe 3 Software Qualität

Punkte: ____

3 Punkte

In der Programmiersprache C können Ausdrücke wie dieser: `c = a--b;` zu Fehlinterpretationen führen. Erklären Sie die Greedy Strategie, die hilft, den Ausdruck richtig zu interpretieren.

Aufgabe 4 Software Qualität

Punkte: ____

5 Punkte

Zum Thema Optimierungsfehler:

a) Zu welchem Zweck werden typischerweise Optimierungen an funktional korrektem Code vorgenommen? Nennen Sie genau zwei Gründe.

b) Welche besondere Gefahr besteht bei Code Optimierungen und warum?

c) Wie können Sie dieser Gefahr entgegenwirken?

Aufgabe 5 Code Kommentare

Punkte: ____

4 Punkte

Betrachten Sie folgenden Code Ausschnitt hinsichtlich der Dokumentation.

```
/*
Funktion int ack(int n, int m)
Autor: Irgendwer
Datum: 15.3.2013
Revision History:
    12/15/2006: While Iteration eingefügt
    13/01/2008: Funktionsname geändert

*/
int ack(int n, int m)
{
    /*solange die erste Variable nicht null ist ...*/
    while(n!=0)
    {
        /*teste zweite Variable auf 0 */
        if(m==0)
        {
            m=1;
        }
        else
        {
            /*hier erfolgt der rekursive Aufruf*/
            m=ack(m,n-1);
        }
        n--
    }
    /*Liefere Ergebnis zurueck*/
    return m+1;
}
```

a. Was ist an dieser Dokumentation schlecht?
2 Punkte

b. Wie können Sie die Dokumentation verbessern?
2 Punkte

Aufgabe 6 Exception Handling

Punkte: ____

6 Punkte

Perspektiven des Exception Handlings: Beim Design der Exceptions, die eine Klasse werfen kann, sind verschiedene Perspektiven zu berücksichtigen.

a) Wer ist an der Information interessiert, die die geworfene Exception gibt?
3 Punkte

b) Welche Informationen muss daher aus eine Exception ersichtlich sein?
3 Punkte

Aufgabe 7 Continuous Integration

Punkte: ____

8 Punkte

Sie setzen eine Continuous Integration Infrastruktur mit den Produkten Subversion, Maven, Nexus, Hudson auf.

a. Warum benötigen Sie Nexus oder ein ähnliches Produkt?
3 Punkte

b. Wie arbeiten Nexus und Maven zusammen?
2 Punkte

c. Anhand welcher Koordinaten lokalisiert Maven Artefakte anderer Projekte?
3 Punkte

Aufgabe 8 JUnit

Punkte: ____

5 Punkte

Erklären Sie die folgenden Begriffe im Zusammenhang mit JUnit:

a. Testklasse
1 Punkt

b. TestSuite
2 Punkte

c. TestFixture
2 Punkte

Aufgabe 9 JUnit

Punkte: ____

3 Punkte

Was sind parametrisierbare Tests mit JUnit und wofür werden sie verwendet?

Aufgabe 10 Scrum

Punkte: ____

5 Punkte

In einem großen Scrum Projekt mit **mehr als einem** Scrum Team müssen sich die Teams abstimmen. An welchen Stellen in Ihrem Sprint können Sie das tun? Nennen Sie 5 verschiedene Möglichkeiten, die sich nicht ausschließen, sondern ergänzen.

Teil 2 Praxis

Aufgabe 1 Testen

Punkte: ____

1. Erstellen Sie einen Kontrollflussgraphen für folgende C-Funktion:
   8 Punkte

Funktion **Kontrollflussgraph**

```
void insertionSort(int *arrayToSort, int arrayLength){
    int index;
    int innerIndex;
    int resortIndex;
    int temp;

    for(index =1; index< arrayLength;index ++){

        for(innerIndex = 0; innerIndex <index;innerIndex ++){

            if (arrayToSort[index] < arrayToSort[innerIndex]){

                temp = arrayToSort[index];

                    for(resortIndex = index; resortIndex >= innerIndex; resortIndex--){

                        arrayToSort[resortIndex] = arrayToSort[resortIndex-1];

                    }
                    arrayToSort[innerIndex] = temp;

            }
        }
    }
}
```

2. Definieren Sie möglichst wenige Testfälle, so dass eine vollständige Anweisungsüberdeckung gegeben ist.
   2 Punkte

3. Definieren Sie möglichst wenige Testfälle, so dass eine vollständige Zweigüberdeckung gegeben ist.
   2 Punkte

4. Erklären Sie das Konzept der vollständigen Pfadüberdeckung.
   2 Punkte

5. Wie viele Testfälle benötigen Sie für das Programm oben für eine vollständige Pfadüberdeckung?
   1 Punkt

6. Wie können Sie die Anzahl der Testfälle gegenüber der vollständigen Pfadüberdeckung reduzieren?
   Nennen Sie den Namen des Konzepts und beschreiben Sie es.
   4 Punkte

Aufgabe 2 Anomalienanalyse

Punkte: ____

12 Punkte

Erstellen Sie einen Kontrollflussgraphen und eine Datenflussanalyse des folgenden Programms und markieren Sie gegebenenfalls Anomalien.

```
double sqrt (double x){
    double returnValue =0;
    double w = 0;

    if (x > 0.0){
        while (abs(w*w-x > 0.01)){
            w=w-((w*w-x)/(2.0*w));
        }

    }
    return returnValue;
}
```

Aufgabe 3 Anforderungen

4 Punkte

Betrachten Sie folgende Anforderung an ein Softwaresystem:

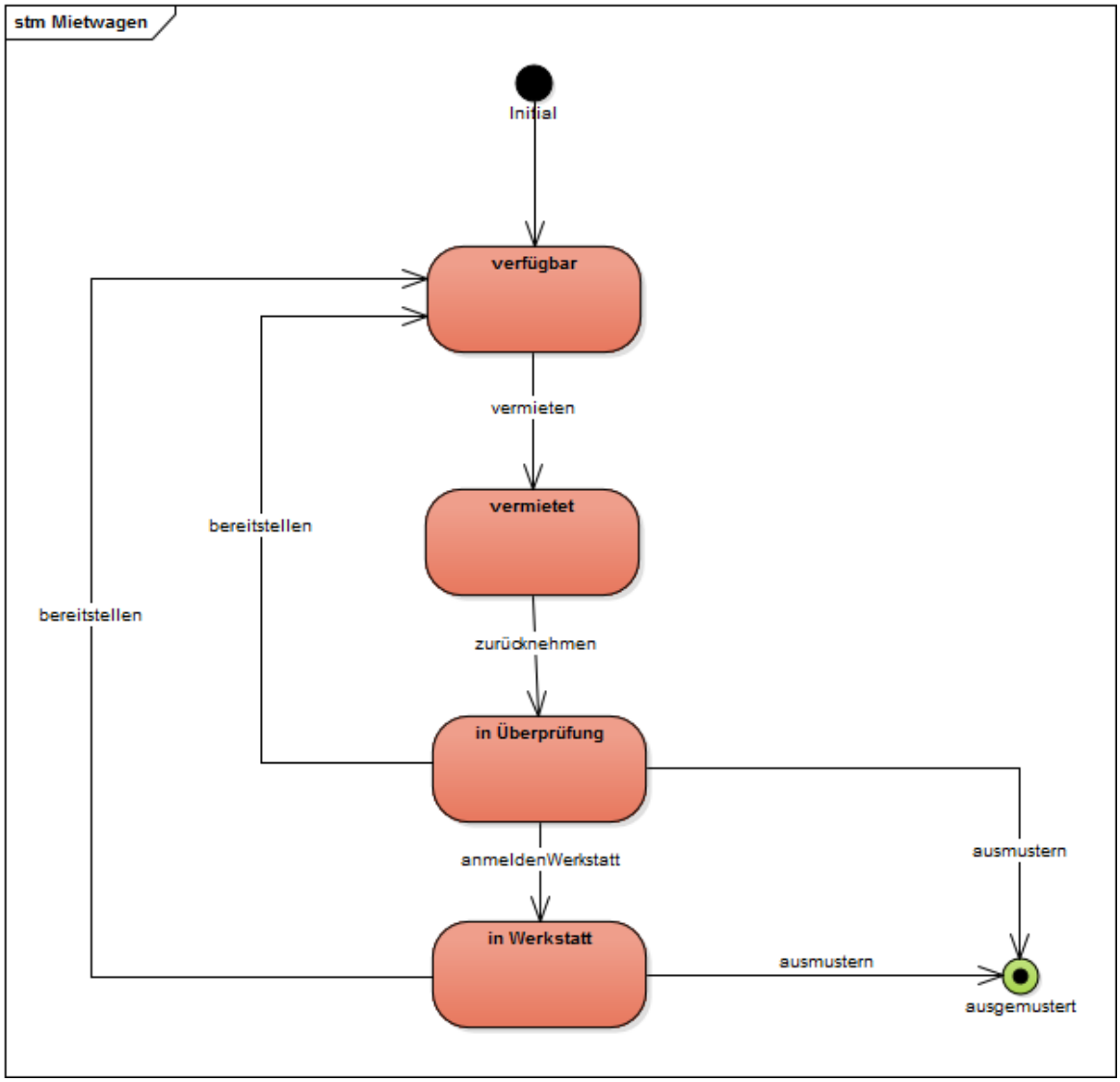
„Das System soll bei Störung gestoppt werden.“

a. Benennen Sie zwei typische Fehler, die der Autor der Anforderung bei der Formulierung gemacht hat.
2 Punkte

b. Welche Information fehlt bei der Anforderung dadurch dass der Autor die unter a) aufgeführten Fehler begangen hat?
2 Punkte

Aufgabe 4 Zustandsbehafteter Software Test.

a. Betrachten Sie folgendes Zustandsdiagramm eines Mietwagens und erstellen Sie einen Übergangsbaum.
6 Punkte



Geben Sie Ihre Antwort auf der nächsten Seite.

Übergangsbaum:

b. Nach welchem **Prinzip** erstellen Sie mithilfe des Übergangsbaums Testfälle?
2 Punkte