Klausur Softwaretechnik

08.09.2006

Prof. Dr. Walter F. Tichy
M. Sc. A. Jannesari
Dipl.-Inform. G. Malpohl

Hier das Namensschild aufkleben.

Zur Klausur sind keine Hilfsmittel und kein eigenes Papier zugelassen. Die Bearbeitungszeit beträgt 60 Minuten. Die Klausur ist vollständig und geheftet abzugeben.

Aufgabe	**1**	**2**	**3**	**4**	**5**	**∑**
Maximal	15	11	11	12	11	**60**
K1						
K2						
K3						

Aufgabe 1: Aufwärmen (4+2+2,5+2+3+1,5= 15P)

a.) Kreuzen Sie an, ob die Aussage wahr oder falsch ist. (4 P)
Hinweis: Jedes korrekte Kreuz zählt 0,5 Punkte, jedes falsche Kreuz bewirkt 0,5 Punkte Abzug! Die Teilaufgabe wird mindestens mit 0 Punkten bewertet.

wahr	falsch	**Aussage**
		Die Durchführbarkeitsuntersuchung und Erzeugung des Pflichtenhefts werden beide in der Definitionsphase durchgeführt.
		Wenn eine Klasse eine abstrakte Methode besitzt, dann ist sie auch selbst abstrakt.
		Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer Schnittstelle, die sich bei einer Änderung der Entscheidung mitändert.
		Die Zuordnung von Modulen zu Schichten in der Schichtenarchitektur ist immer eindeutig.
		Bei Rekursionseliminierung ist eine Transformation in die iterative Form nicht mehr schwierig, sobald eine rechtsrekursive Form vorliegt.
		Kontrollflussorientierte Tests und datenflussorientierte Tests gehören zu der statischen Analyse von Programmen.
		Die einfache Bedingungsüberdeckung fordert, dass die atomaren Bedingungen mit allen möglichen Kombination der Wahrheitswerte W und F belegt werden.
		Im "Synchronisiere und Stabilisiere-Model" ist die Priorisierung nach Funktionen nicht möglich.

b.) Welche vier Arten von Anforderungen gibt es bei der Anforderungsermittlung? (2 P)

c.) Sie entwerfen ein Modul nach dem Geheimnisprinzip. Nennen Sie fünf Aspekte, die Sie dabei möglicherweise verbergen wollen. (2,5 P)

d.) Nennen Sie vier Techniken, die bei der Optimierung auf Ebene der „Algorithmen und Datenstrukturen“ zum Einsatz kommen. (2 P)

e.) Definieren Sie die Begriffe Testobjekt, Testfall und Testtreiber. (3 P)

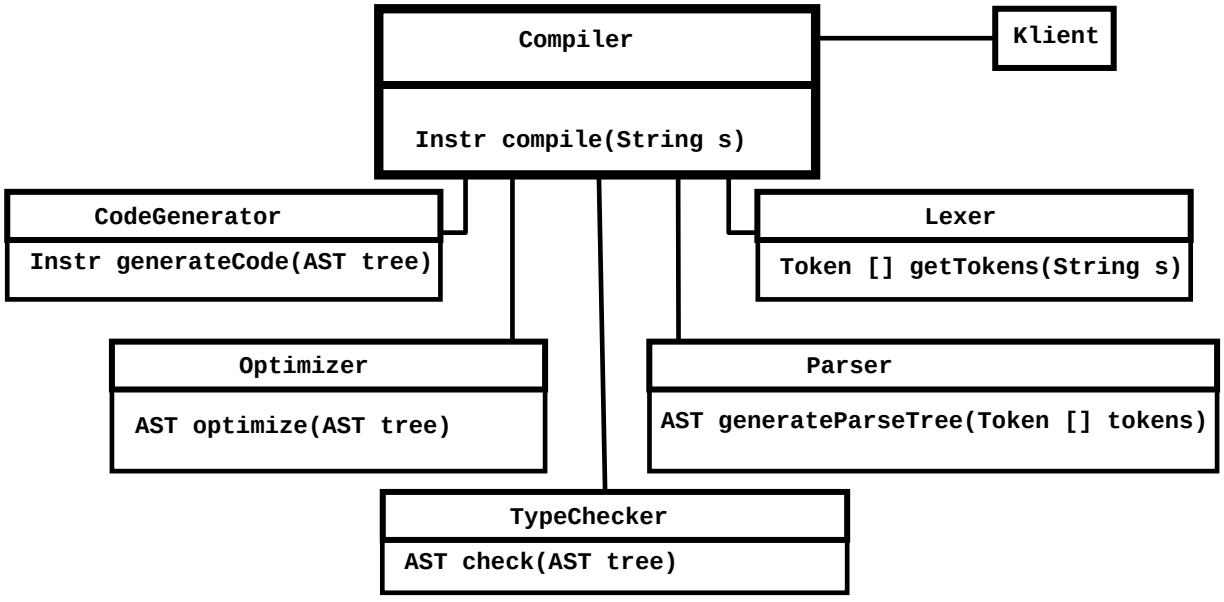
f.) Nennen Sie drei Kategorien von Tätigkeiten in der Wartungs- und Pflegephase. (1,5 P)

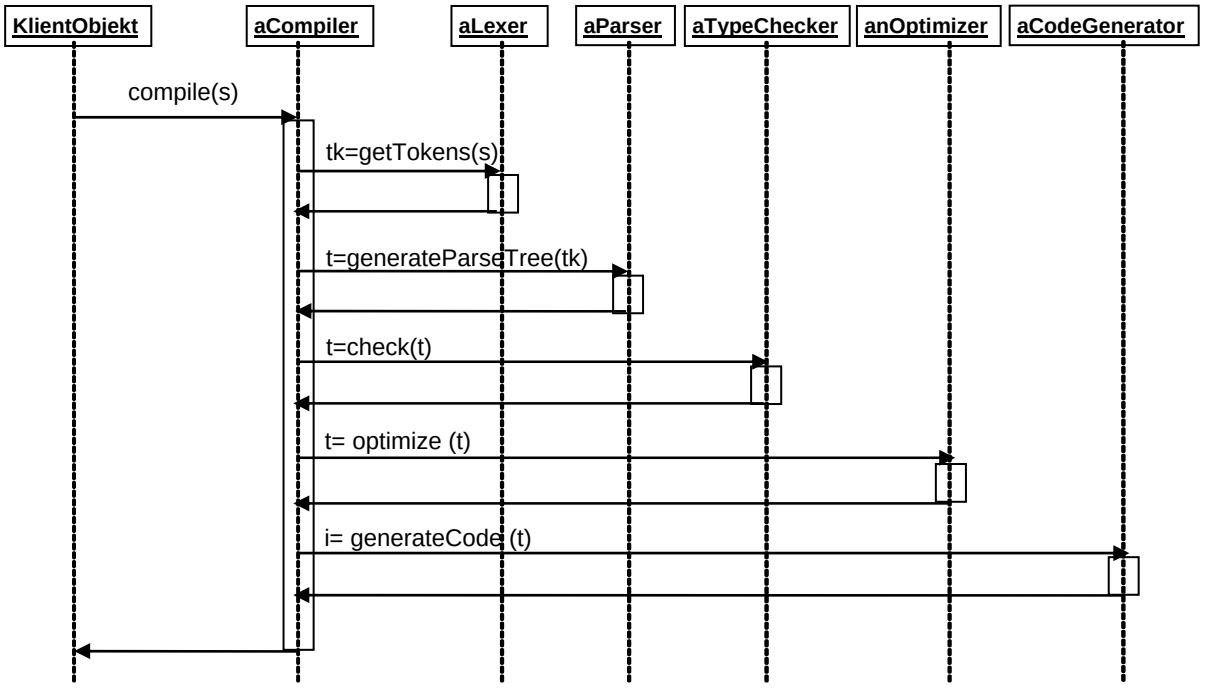
Aufgabe 2: Entwurfsmuster (5+3+3= 11P)

a.) Ein binärer arithmetischer Ausdruck enthält einen Operanden, einen Operator [+ - * /] und einen anderen Operanden. Die Operanden können entweder eine Zahl sein oder selbst wieder ein anderer binärer arithmetischer Ausdruck (z.B. 2 + 3 und (2 + 3) + (4 * 6)). Entwerfen Sie mit Hilfe eines Entwurfsmusters ein Klassendiagramm, um die Bestandteile eines binären arithmetischen Ausdrucks zu einer Baumstruktur zusammenzufügen und seine Bestands-Hierarchien zu repräsentieren. Nennen Sie das verwendete Entwurfsmuster. (5 P)
Hinweis: Die Klammern sind im Klassendiagramm nicht zu berücksichtigen.

Verwendetes Entwurfsmuster: _____

b.) Gegeben sind das folgende Klassendiagramm und Sequenzdiagramm:





Welches Entwurfsmuster wäre für den oben stehenden Entwurf geeignet? Zu welcher Kategorie gehört dieses Muster und welchen Zweck erfüllt es im Allgemeinen? (3 P)

c.) Implementieren Sie nun die Methode c*ompile()* der Compiler-Klasse in Java. Achten Sie auf die Rolle der Compiler-Klasse in dem Klassendiagramm und dem Sequenzdiagramm. (3 P)

```
public class Compiler  {
      private Lexer aLexer = new Lexer();
      private Parser aParser = new Parser();
      private TypeChecker aTypeChecker = new TypeChecker ();
      private CodeGenerator aCodeGenerator = new CodeGenerator();
      private Optimizer anOptimizer = new Optimizer();
      public Instr compile(String s)  {

      }
}
```

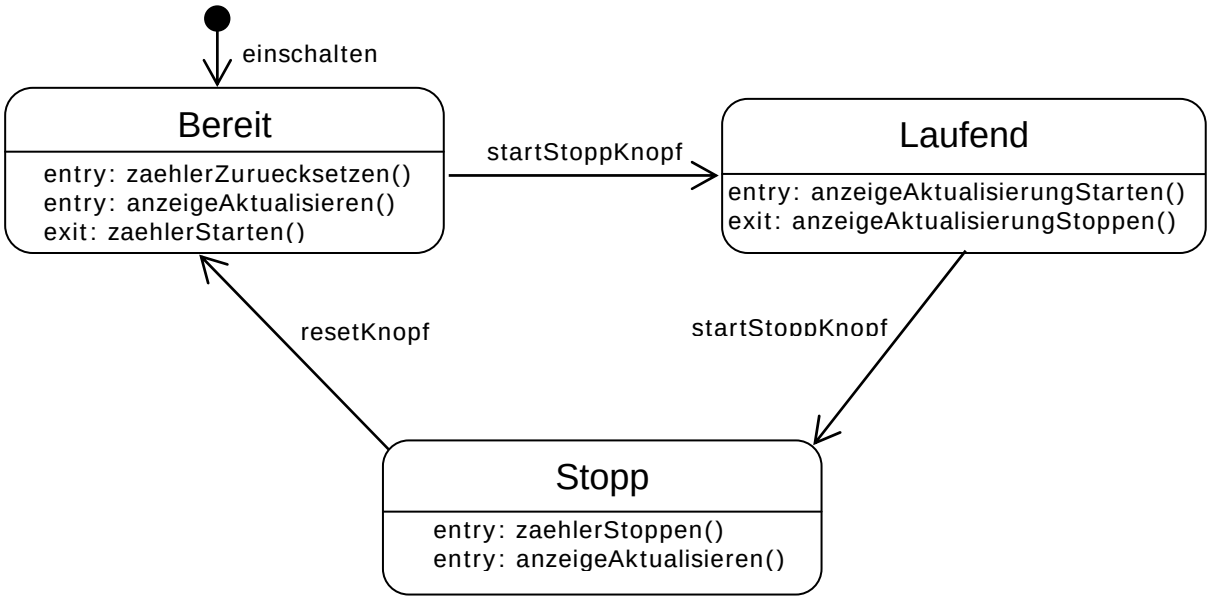
Aufgabe 3: Aktivitätsdiagramm (11P)

Entwerfen Sie ein Aktivitätsdiagramm, das die Durchführung einer Klausur beschreibt. Halten Sie sich dabei so eng wie möglich an die nachfolgende Beschreibung und achten Sie darauf, parallele Aktivitäten korrekt zu synchronisieren. Beginnen Sie mit der Modellierung der Aktivitäten nach *Betreten des Hörsaals*.
Hinweis: Aktivitäten der Studenten sind nicht zu modellieren.

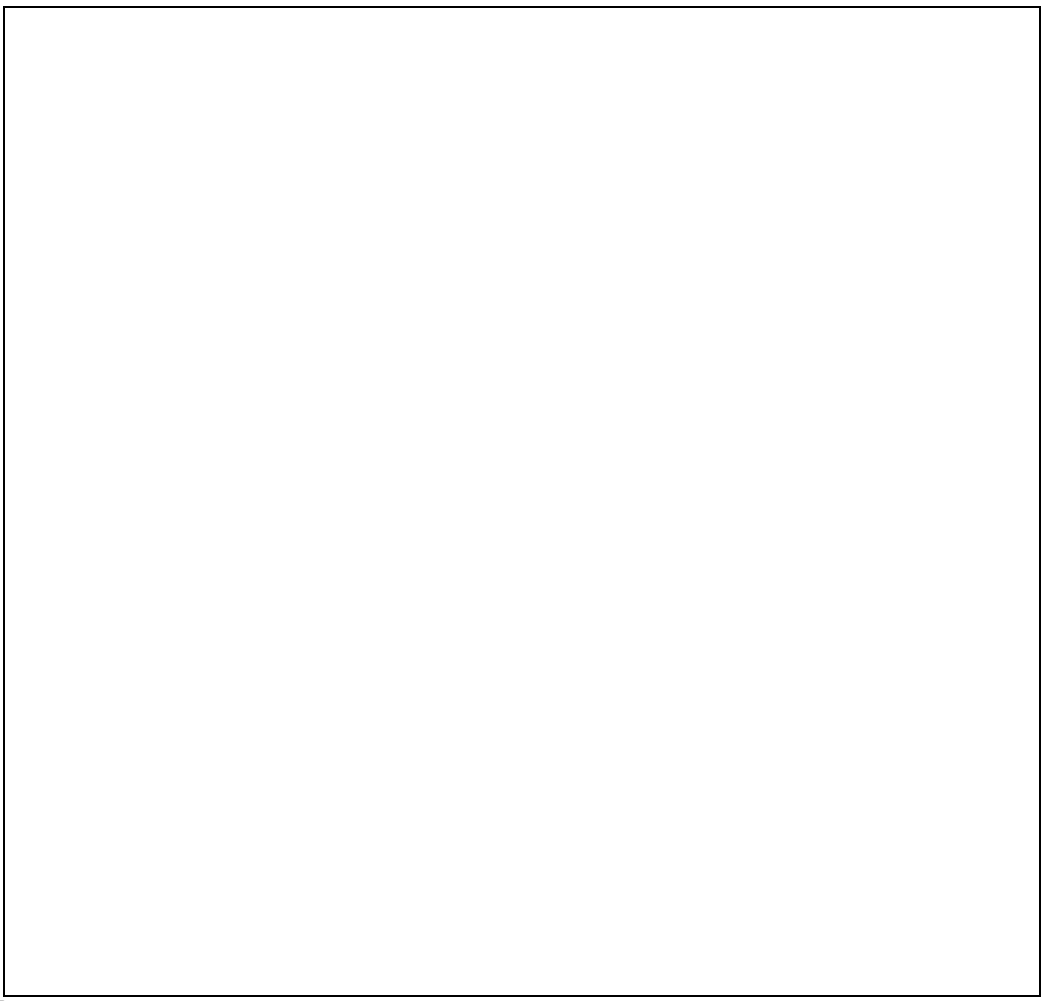
> Die Klausuraufsicht wird von drei Personen durchgeführt, die alle Mitarbeiter der Universität sein müssen: Der Hauptverantwortliche (A1) wird von zwei Helfern (A2 und A3) unterstützt, alle drei tragen Uhren.
> *Nach Betreten des Hörsaals beginnt A1 die Tafel zu beschriften, während einer seiner Helfer die Etiketten auslegt und der andere die mitgebrachten Türschilder („Bitte nicht stören! Klausur!“) an den Türen anbringt. Auf den Etiketten stehen der Vor- und Nachname jedes Studenten, seine Matrikelnummer und eine fortlaufende Nummer, die später die Verwaltung der Klausur vereinfacht.*
> *Sind diese Aufgaben erledigt, kann der Einlass beginnen. Dazu öffnen die Helfer die Türen des Hörsaals und A1 beaufsichtigt den Einlass. Solange noch nicht alle Studenten ihren Platz gefunden haben, helfen A2 und A3 bei der Platzsuche.*
> *Der Hauptverantwortliche A1 wartet ab, bis alle ihre Plätze gefunden haben. Dann erklärt er den Ablauf, beaufsichtigt das Austeilen der Klausurblätter und verkündet den Beginn der Klausur. Anschließend beaufsichtigt er 60 Minuten lang die Klausur und sagt das Ende der Klausur an.*
> *Nachdem A1 den Ablauf der Klausur angesagt hat, sind A2 und A3 für das Austeilen der Klausur, die anschließende Aufsicht zuständig. Während der Bearbeitungszeit geht zusätzlich einer von beiden herum und kontrolliert die Studentenausweise, während der andere Aufsicht führt.*
> *Während der Klausur verlässt keine Aufsicht den Hörsaal.*

Aufgabe 4: Abbildung von UML auf Code (12 P)

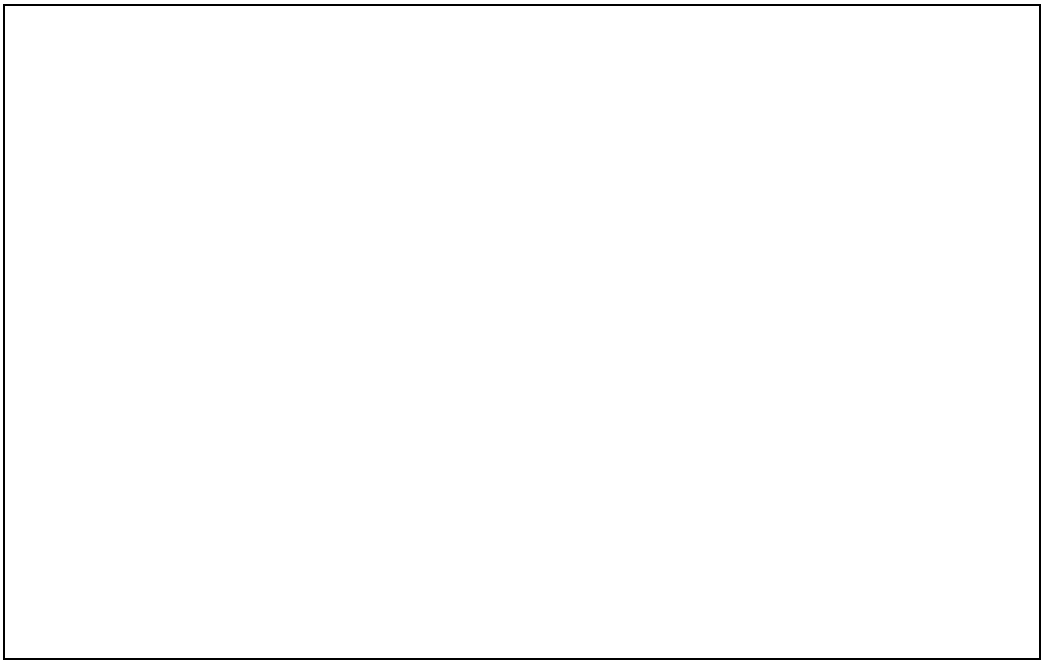
Das folgende Zustandsdiagramm veranschaulicht die Zustände einer Stoppuhr. Sie kann sich in einem der folgenden Zustände befinden: *Bereit*, *Laufend* oder *Stopp*. Vervollständigen Sie den Java-Code für das gegebene UML-Diagramm. Implementieren Sie den Code so, dass sie möglichst genau das durch das Zustandsdiagramm spezifizierte Verhalten an den Tag legen. Verwenden Sie hierfür die Lücken der Vorlage auf dieser und der folgenden Seite. Achten Sie auf korrekte Java-Syntax!
Hinweis: Sobald ein Knopf gedrückt wird, wird die zuständige Methode (z.B. startStop-Knopf()) durch den Event-Handler ausgeführt.




```
public class StoppUhr {

      private enum Zustand {
            BEREIT, LAUFEND, STOPP
      }
      private Zustand aktuellerZustand;

      public StoppUhr() {
            aktuellerZustand = Zustand.BEREIT;
      }
      private void zaehlerZuruecksetzen() {…}
      private void anzeigeAktualisieren() {…}
      private void zaehlerStarten() {…}
      private void anzeigeAktualisierungStarten () {…}
      private void anzeigeAktualisierungStoppen () {…}
      private void zaehlerStoppen() {…}
      public void einschalten() {…}

      public void startStoppKnopf() {

            switch (aktuellerZustand) {
```



```
        }
    }

    public void resetKnopf() {
```



```
    }
}
```

Aufgabe 5: Testen (7+3+1= 11 P)

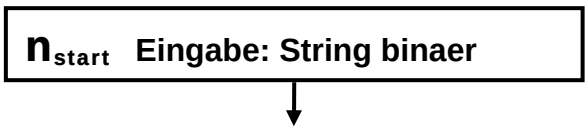
Gegeben ist die folgende Java-Funktion. Es wandelt eine Binärzahl in eine Dezimalzahl um.

```
public static long wandleDezimalZahl(String binaer) {
    long zahl = 0;
    if (binaer != null) {
      for (int i = 0; i < binaer.length(); i++) {
        char zeichen = binaer.charAt(i);
        zahl *= 2;
        if (zeichen == '1') {
          zahl++;
        } else if (zeichen != '0') {
          zahl = -1;
          break;
        }
      }
    }
    return zahl;
}
```

a.) Erstellen Sie auf der folgenden Seite den **Kontrollflussgraphen** der Funktion *wandleDezimalZahl()*. Wenden Sie dabei das aus der Vorlesung bekannte Verfahren an. (7 P) *Hinweis*: Die Zwischensparche wird nicht bewertet.

b.) Erstellen Sie zwei minimale Testfall-Mengen: Eine, für die Anweisungsüberdeckung und eine andere für die minimale Zweigüberdeckung des obigen Programms. Begründen Sie Ihre Antworten. (3 P)

c.) Gibt es für dieses Programm eine Testfallmenge, die das Kriterium der Pfadüberdeckung erfüllt? Begründen Sie Ihre Antwort. (1 P)


n_{start} Eingabe: String binaer


n_{stopp} return zahl;