

Lösungsvorschlag Klausur OOSE WS 05/06

Aufgabe 1

(6 Punkte)

- (a) Aus welchen 6 Teilen besteht eine Anwendungsfall-Beschreibung (gemäß des in der Vorlesung angegebenen Formats)? (3 Punkte)

Eine Anwendungsfall-Beschreibung besteht aus:

- Name des Anwendungsfalls
- Kurzbeschreibung
- Vorbedingung
(Voraussetzung für eine erfolgreiche Ausführung des Anwendungsfalls)
- Nachbedingung
(Zustand nach erfolgreicher Ausführung)
- einem Standardablauf (Primärszenario)
(Schritte bzw. Interaktionen, die im Normalfall bei Ausführung des Anwendungsfalls durchlaufen werden)
- mehreren Alternativabläufen (Sekundärszenarien)
(bei Fehlerfällen und Optionen)

(Zusätzlich kann ein Aktivitätsdiagramm für den Anwendungsfall angegeben werden.)

- (b) Aus welchen 5 Schritten besteht die in der Vorlesung angegebene Vorgehensweise zur Entwicklung eines statischen Modells in der objektorientierten Analyse? (3 Punkte)

1. Klassen identifizieren
2. Assoziationen bestimmen
3. Attribute identifizieren
4. Vererbung einführen
5. Modell überarbeiten

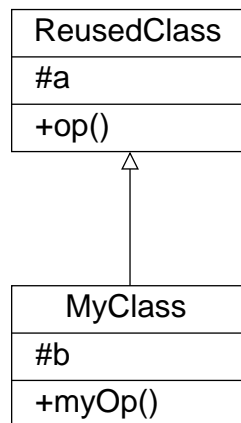
Aufgabe 2

(6 Punkte)

Beschreiben Sie die beiden in der Vorlesung angegebenen Techniken zur Wiederverwendung von Klassen im objektorientierten Entwurf (jeweils mit kurzer Erläuterung anhand eines einfachen Klassendiagramms).

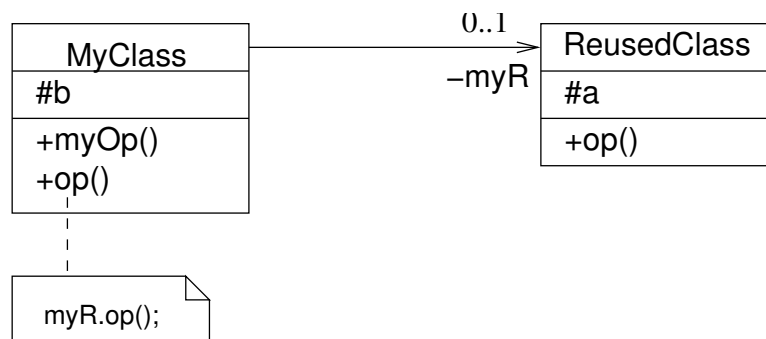
Wiederverwendung durch *Spezialisierung*:

Die Klasse *MyClass* erbt von der Klasse *ReusedClass* die Operation *op*.



Wiederverwendung durch *Delegation*:

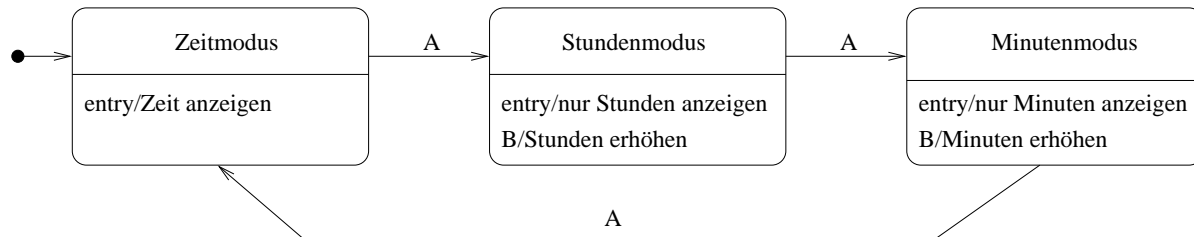
Die Klasse *MyClass* delegiert den Aufruf von *op* an die Klasse *ReusedClass*.



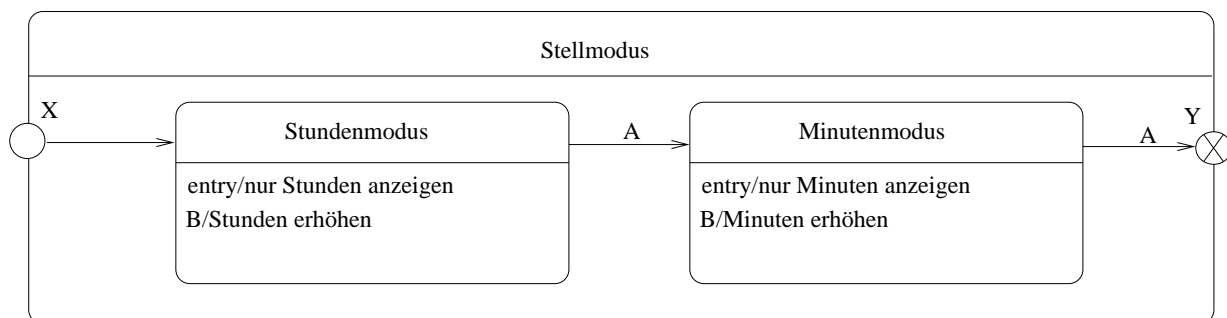
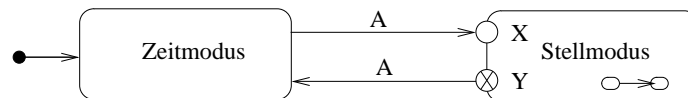
Aufgabe 3

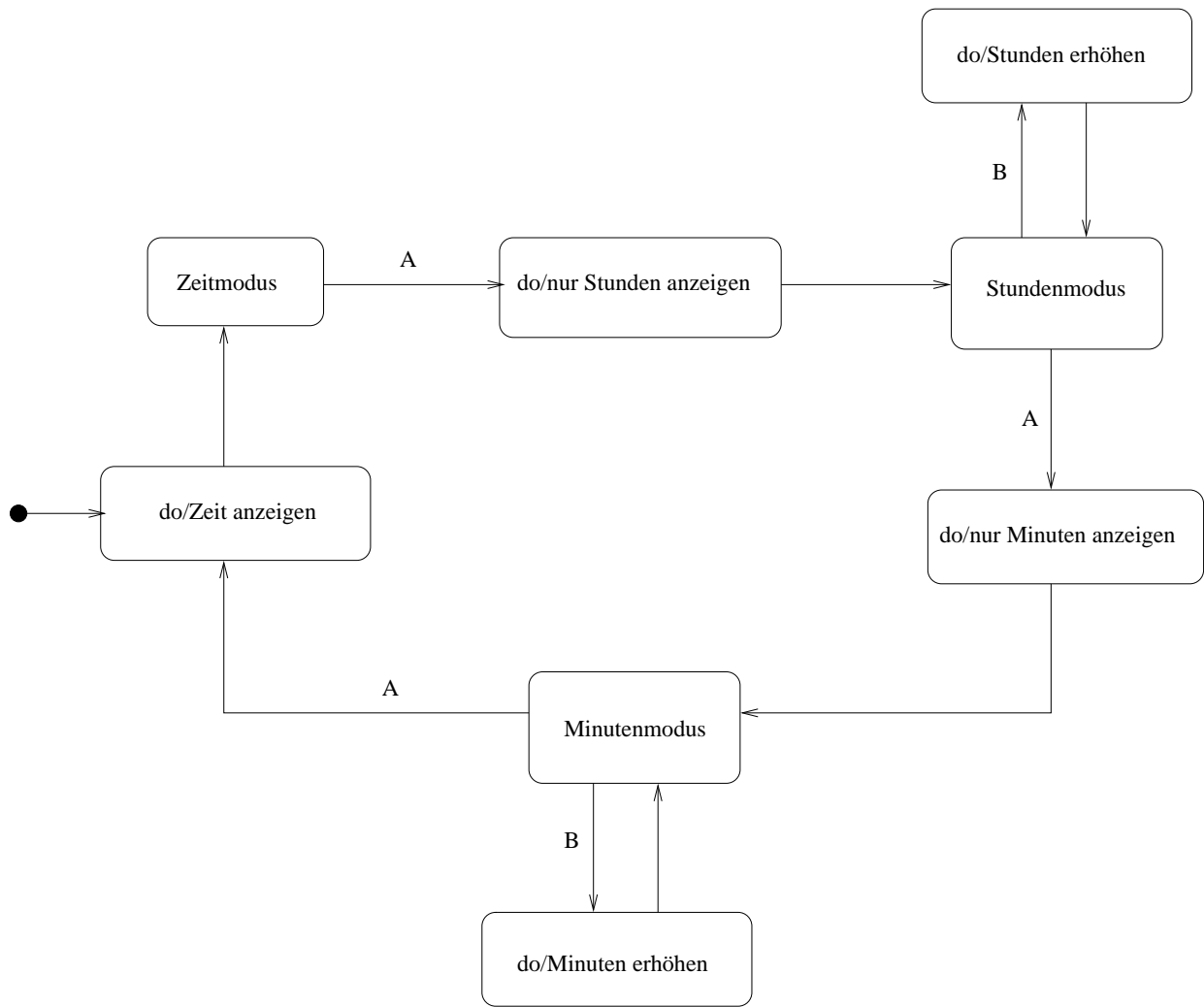
(10 Punkte)

Das folgende Zustandsdiagramm beschreibt das Verhalten einer einfachen Digitaluhr mit zwei Druckknöpfen A und B zum Einstellen der Stunden bzw. Minuten.



- (a) Zeichnen Sie auf der Rückseite dieses Blatts ein äquivalentes Zustandsdiagramm, in dem nur stabile Zustände und Aktivitätszustände vorkommen. (7 Punkte)
- (b) Wir betrachten wieder das oben angegebene Zustandsdiagramm. Die beiden Zustände *Stundenmodus* und *Minutenmodus* sollen als sequentielle Unterzustände in einem komplexen Oberzustand namens *Stellmodus* zusammengefasst werden. Zeichnen Sie auf der unteren Hälfte dieses Blatts zwei Diagramme: Eine abstrakte Darstellung des Zustandsdiagramms ohne Detaillierung des Oberzustands; eine gesonderte detaillierte Darstellung des Oberzustands mit seinen Unterzuständen und Transitionen. (Hinweis: Verwenden Sie einen *entry* und einen *exit point*.) (3 Punkte)





Aufgabe 4

(14 Punkte)

Gegeben sei das folgende Java-Programm:

```
class M {
    private boolean b;
    private A a;
    public void op() {
        F f = new F();
        a = new A(f);
    }
}

class A {
    private R r;
    public A(I i) {
        r = i.createX();
    }
}

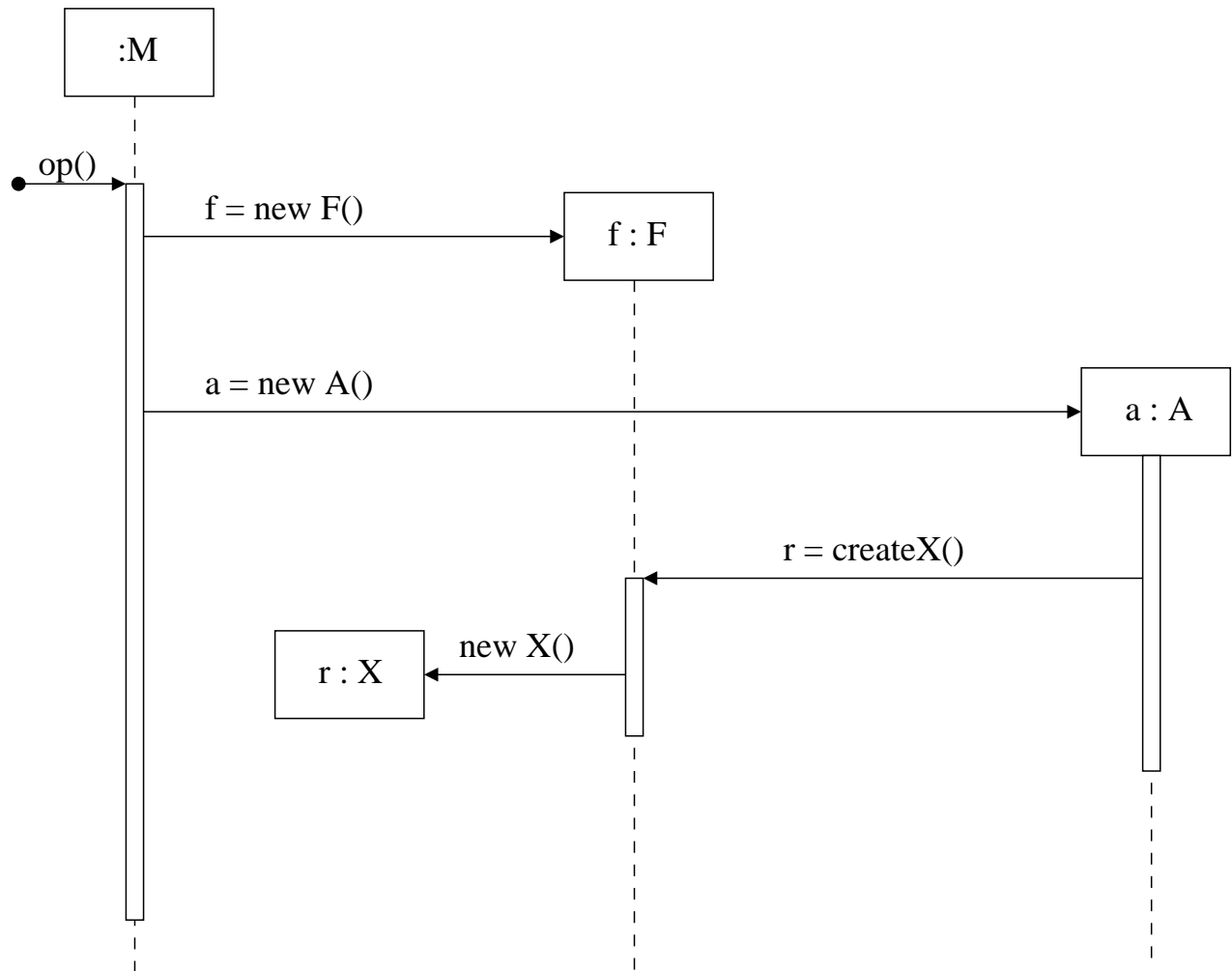
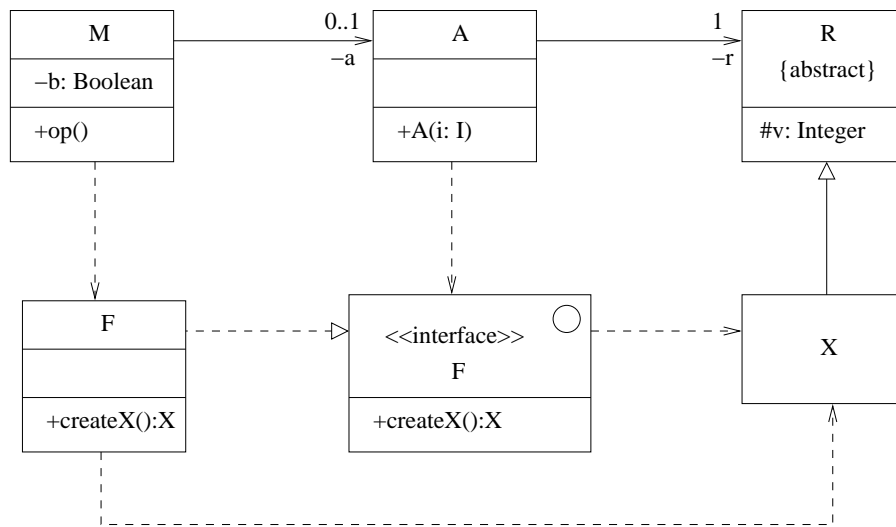
interface I {
    public X createX();
}

class F implements I {
    public X createX() {
        return new X();
    }
}

abstract class R {
    protected int v;
}

class X extends R {
}
```

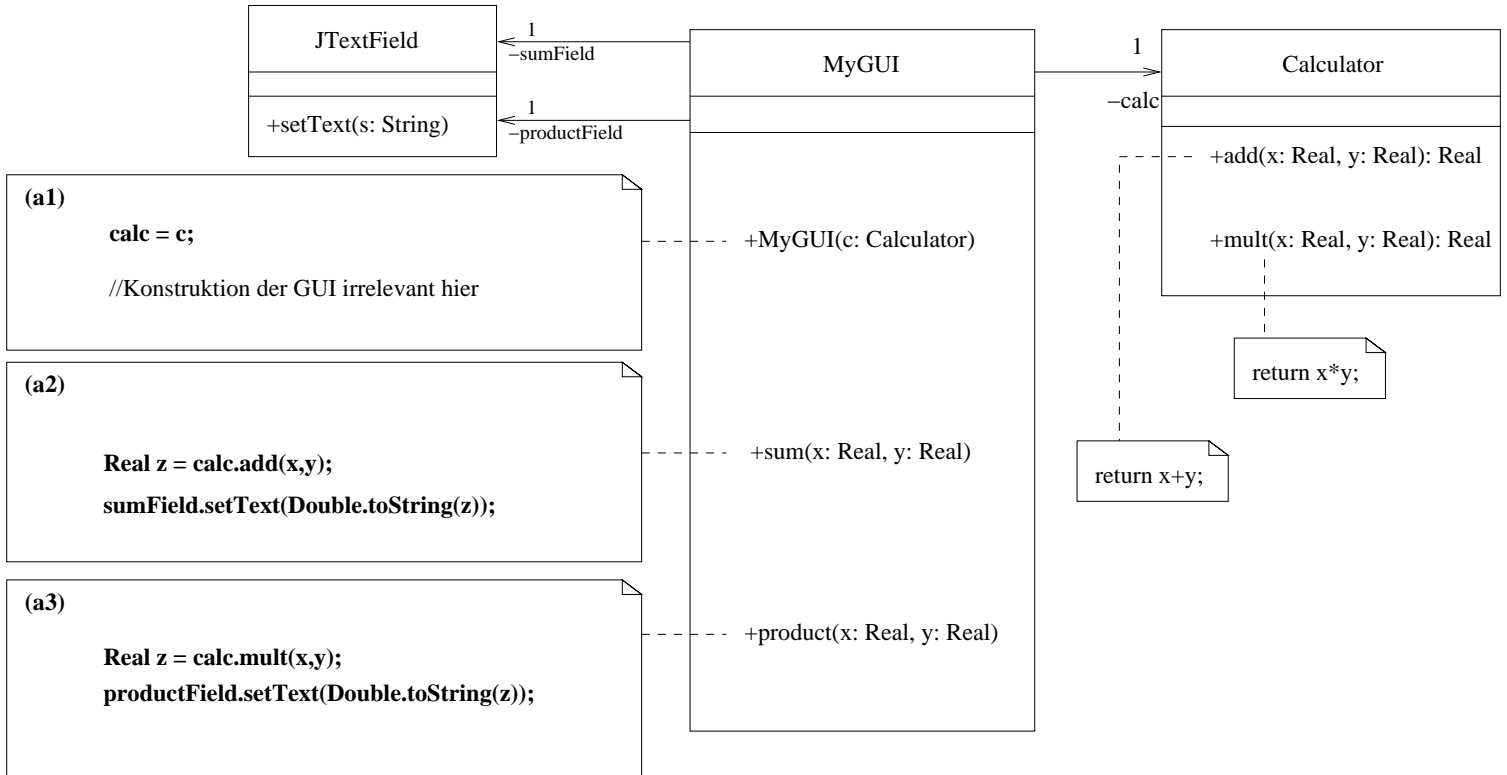
- (a) Zeichnen Sie neben den oben angegebenen Programmcode ein UML-Klassendiagramm, das die statische Struktur des Programms modelliert. Als Typen von Attributen sollen nur Standarddatentypen verwendet werden. Geben Sie auch Multiplizitäten und Rollennamen von (evtl. gerichteten) Assoziationen an sowie alle aus dem Programm ersichtlichen Abhängigkeiten. Zugriffsrechte sind ebenfalls zu berücksichtigen. (8 Punkte)
- (b) Vervollständigen Sie den auf der Rückseite dieses Blatts angegebenen Ansatz für ein Sequenzdiagramm, so dass alle auf den Aufruf der Operation *op* folgenden Interaktionen (gemäß der oben angegebenen Methodenrumpfe) modelliert werden. (6 Punkte)



Aufgabe 5

(14 Punkte)

- (a) Gegeben sind die folgenden UML-Modelle (ein Klassendiagramm und zwei Sequenzdiagramme) zur Modellierung eines einfachen Systems zur Berechnung der Summe und des Produkts zweier Zahlen. Bei der Konstruktion eines *MyGUI*-Objekts soll dieses mit einem gegebenen *Calculator*-Objekt verbunden werden. Tragen Sie an der mit (a1) markierten Stelle den dazu benötigten Java-Code ein. Das Sequenzdiagramm *Sum* zeigt die Interaktion, die beim Aufruf der Operation *sum* durchgeführt wird. Tragen Sie den zugehörigen Java-Code an der Stelle (a2) ein. Analog soll an der Stelle (a3) der Java-Code zur Realisierung des Sequenzdiagramms *Product* eingetragen werden. (4 Punkte)



- (b) Die Kommunikation zwischen *MyGUI* und *Calculator* zur Berechnung von Summe und Produkt soll nun nicht mehr über Rückgabewerte erfolgen sondern mittels indirekter Kommunikation unter Verwendung der Observertechnik. Dazu sind in dem auf der Rückseite dieses Angabenblatts angegebenen Klassendiagramm einerseits die mit (b1) bis (b5) markierten Stellen durch den benötigten Java-Code zu ergänzen, andererseits sind noch

