Software-Entwicklung in Großprojekten

Braindump vom 8.4.2019

Wintersemester 18/19

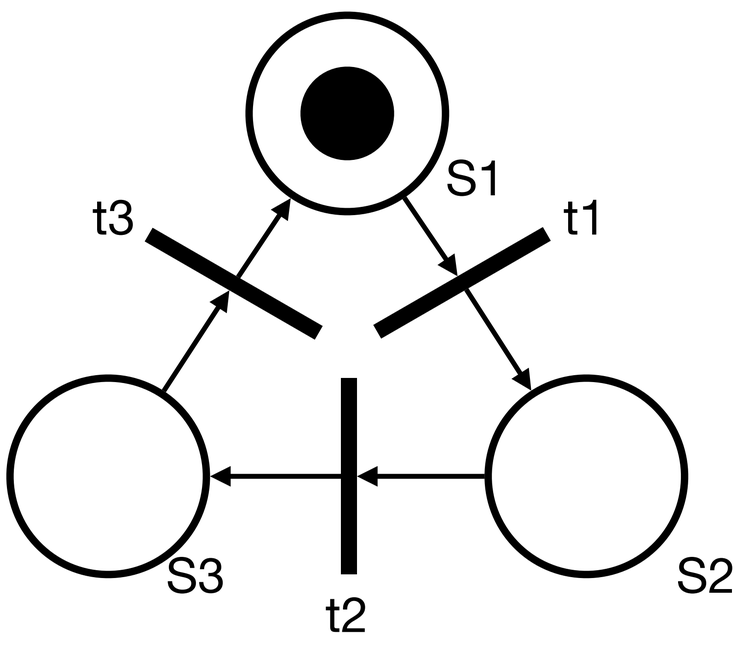
1) Wissensfragen

Falsche Aussagen müssen begründet werden

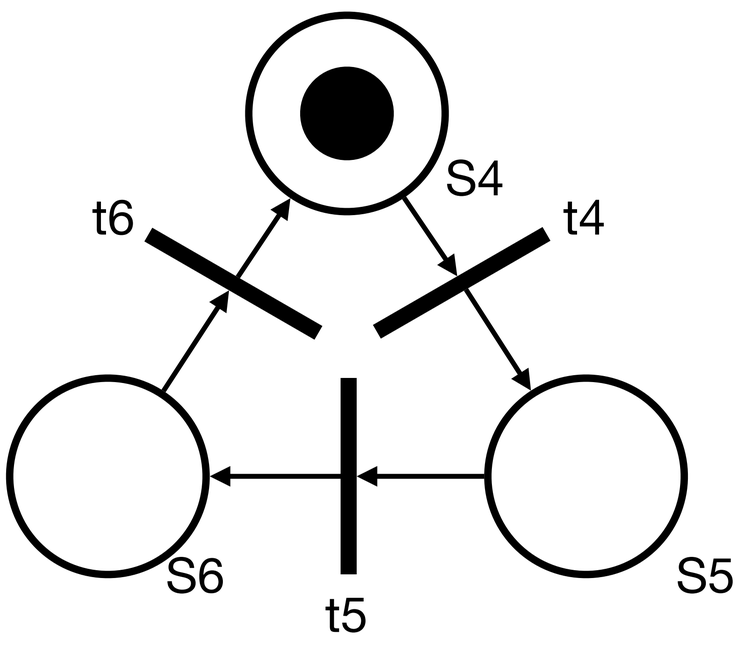
	WAHR	FALSCH
Eine System-Spezifikation ist korrekt, wenn es möglich ist sie unter den im Voraus bekannten Randbedingungen in ein Ablauffähiges System um zu setzen	☐	☐
Zwei Module p und q haben Inhaltskopplung, wenn Modul p eine Datenstruktur als Parameter an Modul q übergibt. Modul q aber nur Teile dieser Datenstruktur verwendet	☐	☐
Ein Modul hat prozedurale Kohäsion, wenn es eine Reihe von Aktionen auf gemeinsamen Daten ausführt. Die Reihenfolge ist irrelevant	☐	☐
Jedes Kommunikationsdiagramm lässt sich in ein semantsich äquivalentes Sequenzdiagramm transformieren	☐	☐
Für eine Bottom-Up Integration ist der Einsatz von Stubs erforderlich	☐	☐
Der Systemtest dient zur Erkennung von Unstimmigkeiten zwischen Modulen und Software-Feinentwurf	☐	☐

2) Petrinetz

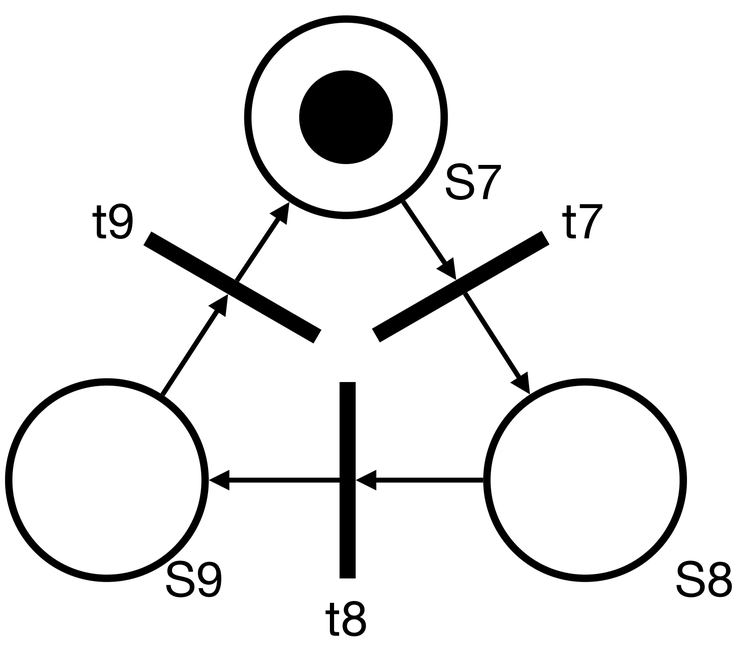
Prozess 1



Prozess 2



Prozess 3



a) Erweitern sie das Petri-Netz um Token, Plätze, nicht inhibitierende Kanten und Transitionen, sodass:

   1) Zu jedem Zeitpunkt maximal zwei Prozesse in Phase 3 sind

   2) Ein Prozess darf nur nach Phase drei übergehen, wenn sich mindestens ein weiterer Prozess in Phase 2 befindet

   3) Jeder Prozess muss alle Phasen immer wieder durchlaufen können

b) Gibt es eine minimale Anzahl an Token, die nie überschritten wird? Falls ja geben sie diese Anzahl, die zugehörige Markierung und eine Sequenz von Schaltungen an, damit diese Markierung erreicht wird an. Falls nein begründen sie dies anhand einer Sequenz von Schaltungen

3) Entwurfsmuster

Ein Webserver stellt im Internet Wetterdaten der letzten 10 Jahre für ausgewählten Städte zur Verfügung. Die Wetterdaten sind in einer Datenbank gespeichert. Die Klasse DBWeather stellt eine Schnittstelle zu dieser Datenbank bereit. Um an Wetterdaten zu kommen gibt es die Methode:

```
getWeather(date: DateTime, city: String)
```

Zur Performancesteigerung wird ein Caching-Mechanismus eingebaut.

a) Welches aus der Vorlesung bekannte Entwurfsmuster eignet sich zur Umsetzung des gewünschten Verhaltens

b) Welcher Klasse und welchem Gültigkeitsbereich lässt sich dieses Entwurfsmuster zuordnen

c) Zeichnen Sie ein Klassendiagramm der Hierachie nach Anwendung des Entwurfsmusters

d) Zeichnen sie je ein Sequenzdiagramm für einen Cache Hit und einen Cache Miss

4) Refactoring

```
public class Pizza {
        public String type = "";

        public double getPrice() {
                try {
                        switch(type) {
                                case "Haehnchen": return 12;
                                case "Salami": return 10.5;
                                case "Vegetarisch": return 9;
                        }
                }
                throw new RuntimeException("Unsupportet_Pizza_Type");
        }
}
```

```
public class NormalOrder {
        List<Pizza> pizzas;

        public NormalOrder(List<Pizza> pizzas) {
                this.pizzas = pizzas;
        }

        public double getTotalPrice() {
                double result = 0;

                for(Pizza p : pizzas) {
                        result += getChargeFor(p);
                }

                return result;
        }

        private double getChargeFor(Pizza p) {
                return p.getPrice();
        }
}
```

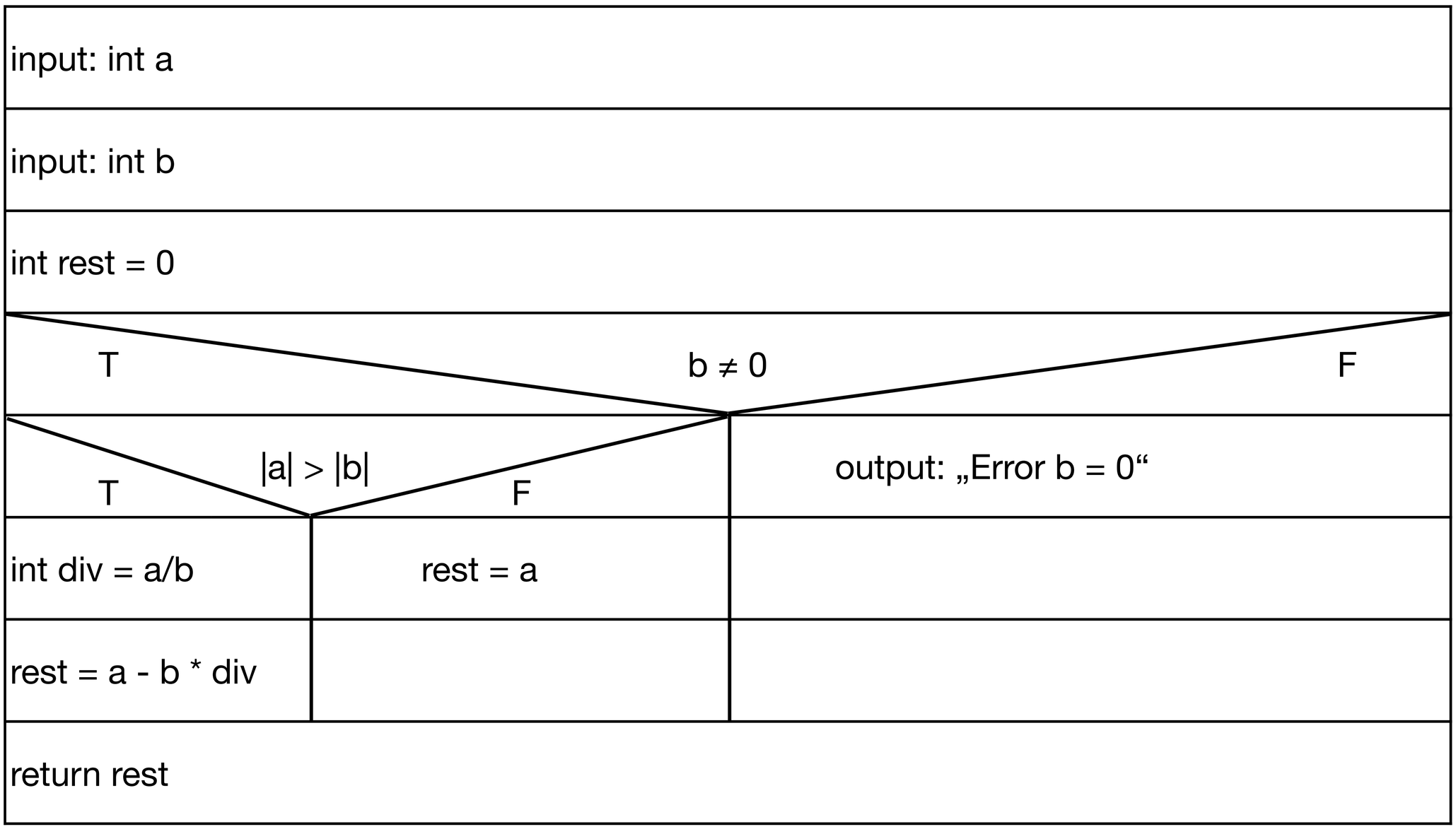
```
public class SpecialOrder {
        List<Pizza> pizzas;

        public SpecialOrder(List<Pizza> pizzas) {
                this.pizzas = pizzas;
        }

        public double getTotalPrice() {
                double result = 0;

                for(Pizza p : pizzas) {
                        result += getChargeFor(p);
                }

                return result;
        }

        private double getChargeFor(Pizza p) {
                return p.getPrice() * 0.9;
        }
}
```

a) Nennen Sie 2 Refactoring Technicken, die an diesem Beispiel anwendbar sind und begründen Sie jeweils kurz.

b) Erstellen Sie ein UML-Klassendiagramm, das den Code nach der Durchführung ihres Refactorings darstellt.

5) Strukturiertes Testen

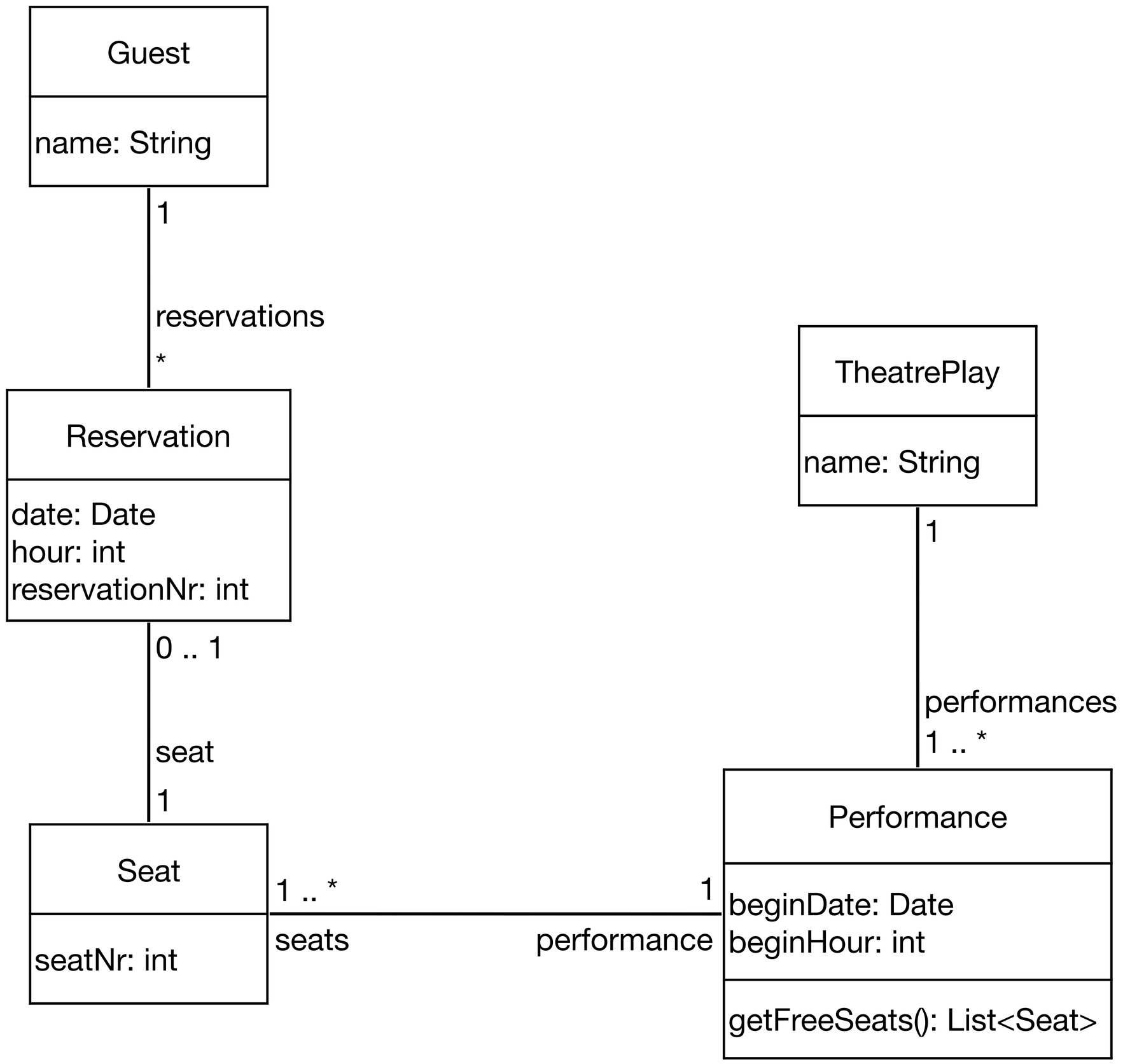
Das dargestellte Programm soll „modulo“ korrekt berechnen.



Testfall (a,b) Menge von Testfällen T = {(12, 5), (-12, 5), (12, -5), (-12, -5), (3, 0), (2, 5)}

a) Zeichnen Sie den zugehörigen Kontrollflussgraphen
b) Welches strengstes Kriterium zur Überdeckung des Graphen, das in der Vorlesung vorgestellt wurde, wird durch T erfüllt?
c) Geben Sie die minimale Teilmenge von T an, die das Kriterium noch erfüllt
d) Welcher Fehler kann mit T nicht gefunden werden
    1) Geben Sie einen Testfall an, der den Fehler findet
    2) Wie kann der Fehler behoben werden?
    3) Welches Testverfahren aus der Vorlesung hätte den Fehler auf jeden Fall entdeckt?

6) OCL



Formulieren sie folgende Anforderungen in OCL

a) Jeder Guest kann maximal 10 Reservierungen vornehmen

b) Jede Reservierungsnummer liegt zwischen 1 und 10000

c) Reservierungen am gleichen Tag wie die zugehörige Vorstellung müssen mindestens 1 Stunde vor Beginn der Vorstellung erfolgt sein

d) Platznummern sind eindeutig

e) Zwischen 12 Uhr und 14 Uhr beginnt keine Vorstellung

f) Der Methodenaufruf p.getFreeSeats() gibt eine Liste aller nicht reservierten Plätze der Performance p zurück