Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

UNIVERSITÄT HAMBURG

Fakultät für Mathematik, Informatik und Naturwissenschaften

KLAUSUR (OOPM)

(Deckblatt)

im Rahmen der Lehrveranstaltung: Softwareentwicklung 2

bei (Prüferin/Prüfer): Dr. Axel Schmolitzky
Prof. Dr.-Ing. Heinz Züllighoven

am: 01.08.2011 (SoSe 2011)

NAME: [redacted]

VORNAME:

MATR.-NR.:

STUDIENGANG:

HINWEIS

Die Klausur ist auf der Seite 14 zu unterschreiben!

Rechtsmittelbelehrung:
Gegen die Bewertung dieser Prüfungsleistung kann innerhalb eines Monats nach ihrer Bekanntgabe Widerspruch erhoben werden. In diesem Zeitraum kann die Bewertung der Klausur eingesehen werden.
Der Widerspruch ist schriftlich oder zur Niederschrift bei der bzw. dem Vorsitzenden des für das Hauptfach zuständigen Prüfungsausschusses einzulegen. Es wird darauf hingewiesen, dass ein erfolgloses Widerspruchsverfahren kostenpflichtig ist.

Erläuterungen für das Beantworten der Fragen:

Für Multiple-Choice-Fragen gilt:

Kennzeichnen Sie *richtige* Antworten zu den gestellten Aufgaben deutlich sichtbar mit einem Kreuz:

☒ <Richtige Antwort>

Wenn eine bereits angekreuzte Antwort doch nicht angekreuzt sein soll, dann füllen Sie das Antwortkästchen vollständig aus und zeichnen ein leeres Kästchen links daneben:

☐ ■ <Doch nicht richtige Antwort>

Angekreuzte falsche Antworten werden *innerhalb einer Frage* von angekreuzten richtigen Antworten abgezogen; *minimal* können pro Frage jedoch nur 0 Punkte erzielt werden. Es empfiehlt sich daher, nur Antworten anzukreuzen, bei denen Sie sich *sicher* sind, dass sie richtig sind.

Jede angekreuzte richtige Antwort erzielt mindestens einen Punkt. Bei Fragen ist vermerkt, wenn eine richtige Antwort mehr als einen Punkt erzielt.

Für Lückentexte und Beschriftungen gilt:

Schreiben Sie beim Ausfüllen von Lückentexten, beim Beschriften von Grafiken oder Aufzählen von Begriffen deutlich. Schreiben Sie – wenn möglich – in Blockschrift. Streichen Sie falsche Worte klar durch. Schreiben Sie ggf. auf einer selbst gezogenen Linie auf derselben Seite, wenn der Platz nicht reicht.

RICHTIGE BESCHRIFTUNG

~~FALSCHES WORT~~ RICHTIGES WORT

Jede richtige Beschriftung zählt mindestens einen Punkt. Falsche Begriffe werden von den richtigen Begriffen abgezogen.

Für Zuordnungen gilt:

Schreiben Sie beim Ausfüllen von Zuordnungen jeweils eine Nummer auf den vorgesehenen Strich.

Kategorie		**Begriff**
Kategorie1	2	1. Begriff A
Kategorie2	1	2. Begriff B
Kategorie3	■ 3	3. Begriff C
Kategorie4	4	4. Begriff D

Punkte für falsche Zuordnungen werden von den Punkten für richtige Zuordnungen abgezogen.

Hilfsmittel:

Es sind keine weiteren Hilfsmittel als ein permanent schreibender Stift zugelassen. Insbesondere dürfen weder Taschenrechner, eigenes Papier noch sonstige Unterlagen verwendet werden.

1. ***Welche der folgenden Charakteristika gelten für den in der Informatik häufig verwendeten Modellbegriff nach Stachowiak? (Mehrere richtige Antworten möglich)***

   - ☐ Ein Modell ist niemals formal.
   - ☐ Ein Modell dient immer einem Zweck.
   - ☐ Ein Modell muss immer interpretiert werden.
   - ☐ Ein Modell ist immer graphisch.

2. ***Welche der folgenden Charakteristika gelten für das Verhältnis von Original und Modell? (Mehrere richtige Antworten möglich)***

   - ☐ Ein Modell kann keine zukünftigen Situationen voraussagen.
   - ☐ Ein Modell ist immer eine Abstraktion seines natürlichen oder künstlichen Originals.
   - ☐ Ein Modell ist objektiv und selbsterklärend.
   - ☐ Ein Modell kann Probehandeln ermöglichen.
   - ☐ Ein Modell kann nicht als Original für ein weiteres Modell dienen.
   - ☐ Ein Grundriss ist ein Modell eines realen Hauses.

3. ***Welche der folgenden Aussagen zu Verifikation und Validation sind richtig? (Mehrere richtige Antworten möglich)***

   - ☐ Verifikation überprüft die Korrektheit und Konsistenz eines Modells.
   - ☐ Ein Compiler validiert einen Programmtext.
   - ☐ Validation ist eine Bewertung eines Modells durch Personen oder Messungen.
   - ☐ Validation ist eine Form eines formalen Beweises.

4. ***Welche der folgenden Aussagen zur Zuverlässigkeit von Software nach Meyer sind richtig? (Mehrere richtige Antworten möglich)***

   - ☐ Zuverlässige Software ist korrekt und robust.
   - ☐ Korrekte Software ist immer robust.
   - ☐ Korrektheit von Software bezieht sich auf das spezifizierte Verhalten; Robustheit bezieht sich auf nicht-spezifizierte Fälle.
   - ☐ Robuste Software ist immer korrekt.

5. ***Welche der folgenden Aussagen zur formalen Semantik von Programmen sind richtig? (Mehrere richtige Antworten möglich)***

   - ☐ Für jede Programmiersprache gibt es eine formale Semantik.
   - ☐ Die bekannten Ansätze der formalen Semantik heißen denotational, motivational, und axiomatisch.
   - ☐ Die denotationale Semantik betrachtet die Abbildung von Speicherzuständen durch Anweisungen.
   - ☐ Die axiomatische Semantik legt für Programmelemente Vor- und Nachbedingungen fest.

6. *Welche der folgenden Aussagen gelten im Zusammenhang mit Abstrakten Datentypen (ADT)? (Mehrere richtige Antworten möglich)*

- ☐ Ein abstrakter Datentyp wird nur durch seine Strukturelemente und ihre Verbindungen definiert.
- ☐ Es soll von der Implementation eines Datentyps abstrahiert werden.
- ☐ Ein ADT definiert Daten über die darauf zulässigen Operationen.
- ☐ Ein abstrakter Datentyp wird durch das Vertragsmodell implementiert.
- ☐ Im Sinne von Meyer besteht ein ADT aus: Typen, Funktionen, Axiomen, Vorbedingungen.
- ☐ Ein abstrakter Datentyp wird nur durch Aufzählung seiner gültigen Werte definiert.

7. *Vervollständigen Sie die Axiome eines Stacks als ADT.*

```
For any s: STACK[T], x: T

top (push (s, x)) = _ _ _ _ _ ?

pop (push (s, x)) = _ _ _ _ _ ?
is_empty (empty) = true

is_empty (push (s, x)) =  _ _ _ _ _?
```

8. *Welche der folgenden Aussagen zum Vertragsmodell sind richtig? (Mehrere richtige Antworten möglich)*

- ☐ Vertragsbedingungen müssen vor und nach der Ausführung einer Operation gelten.
- ☐ Während der Ausführung einer Operation darf die Invariante nicht verletzt sein.
- ☐ Nachbedingungen sind Exceptions.
- ☐ Die Leistungen einer Klasse werden vertraglich geregelt. Beide Vertragsseiten (Klient und Dienstleister) müssen sich an den Vertrag halten.
- ☐ Der Vertrag wird in der Klienten-Klasse festgelegt.
- ☐ Der Dienstleister hat die Pflicht, die versprochene Leistung zu erbringen, wenn die entsprechenden Vorbedingungen erfüllt sind.
- ☐ Beim Aufruf wird der Vertrag an den Dienstleister übergeben (Call by Contract).
- ☐ Der Dienstleister hat die Pflicht, die Operation nur auszuführen, wenn der Zustand des Klienten es erlaubt.
- ☐ Damit ein Vertrag sinnvoll ist, muss der Klient die Einhaltung der Vorbedingung einer zu rufenden Operation prüfen können.
- ☐ Zusicherungen liefern kein Ergebnis.
- ☐ Wenn ein Klient sich nicht an die Vorbedingungen hält, ist der Dienstleister nicht verpflichtet, die Nachbedingungen zu garantieren.
- ☐ Eine Klassen-Invariante gilt für alle Exemplare einer Klasse und muss von allen Operationen berücksichtigt werden. Sie beschreibt (semantische) Randbedingungen einer Klasse insgesamt.

$\sum$

9. *Welche der folgenden Aussagen zu Subtyping sind richtig? (Mehrere richtige Antworten möglich)*

- ☐ Jeder Subtyp bietet alle Operationen seiner Supertypen an.
- ☐ Ein Exemplar eines Subtypen kann überall dort eingesetzt werden, wo ein Exemplar seiner Supertypen erwartet wird.
- ☐ Jede ist-ein-Beziehung ist auch eine Subtyp-Beziehung.
- ☐ Subtyping ist eine Beziehung zwischen zwei Implementationen.
- ☐ Die Verschärfung einer Vorbedingung einer Operation verhindert ein Subtyp-Verhältnis.
- ☐ Redeklarationen von Operationen verhindern generell Subtyp-Beziehungen.
- ☐ Eine Redeklaration durch eine kovariante Anpassung eines Parametertyps verhindert ein Subtyp-Verhältnis.
- ☐ Bei einer typsicheren Anpassung einer Operation in einem Subtyp können auch weitere Parameter definiert werden.
- ☐ Subtyping ist eine Beziehung, die von den Exemplarvariablen der beteiligten Klassen abhängt.
- ☐ Die Nachbedingungen einer Operation dürfen in einem Subtyp verschärft werden.

10. *Wie viele Interfaces kann ein Interface in Java erweitern (mit `extends`)?*

- ☐ Keines.
- ☐ Genau eines.
- ☐ Entweder keins oder eines.
- ☐ Beliebig viele.

11. *Welche der folgenden Aussagen sind für Java korrekt? (Mehrere richtige Antworten möglich)*

- ☐ Über die Variable eines beliebigen Interface-Typs kann immer die Operation `toString` aufgerufen werden.
- ☐ Eine Klasse definiert nur dann einen Typ, wenn sie selbst öffentliche Methoden anbietet.
- ☐ Java erlaubt multiples Subtyping, aber nur einfache Implementationsvererbung.
- ☐ Das Schlüsselwort `static` bei einer Methode bedeutet auch, dass ein Aufruf dieser Methode nicht dynamisch gebunden wird.
- ☐ Ein Interface kann auch eine Klasse erweitern (mit `extends`).
- ☐ Eine Klasse kann maximal ein Interface implementieren.
- ☐ Jeder Subtyp von `Object` verfügt über die Operation `name`, die den Namen eines Objektes liefert.
- ☐ Eine benutzerdefinierte Klasse erbt immer von der Klasse `Object`, entweder unmittelbar oder transitiv.

12. *Welche Aussagen sind für den folgenden Java-Quelltext richtig? (Mehrere richtige Antworten möglich)*

```
public interface A {
    void m(int a);
    void m2(String s);
}

public interface B {
    void m(boolean b);
    void m2(String s);
}

public interface C extends A,B { }
```

- ☐ Das Interface `C` bietet zwei Operationen mit dem Namen `m` an.
- ☐ Das Interface `C` bietet genau eine Operation mit dem Namen `m` an.
- ☐ Das Interface `C` bietet keine Operationen an.
- ☐ Das Interface `C` definiert einen Subtyp des Interfaces `A`.
- ☐ Das Interface `B` definiert einen Subtyp des Interfaces `A`.
- ☐ Das Interface `C` bietet genau eine Operation mit dem Namen `m2` an.

13. *Welche der folgenden Aussagen zur Klassifikation von Programmierfehlern nach Zeller sind richtig? (Mehrere richtige Antworten möglich)*

- ☐ Ein *defect* ist ein Fehler im Programmtext, der beliebig lange unentdeckt bleiben kann.
- ☐ Voraussetzung für einen *defect* ist eine *infection*.
- ☐ Eine *infection* ist eine Verfälschung von Variablenwerten aufgrund der Ausführung eines *defects*.
- ☐ Eine *infection* führt zwangsläufig zu einem *failure*.

14. *Welche der folgenden Aussagen zu Programmierfehlern und Umgebungsfehlern sind richtig? (Mehrere richtige Antworten möglich)*

- ☐ Für einen Programmierfehler kann es innerhalb des Programms selbst keinen korrekten Umgang geben.
- ☐ Je näher die Verfälschung des Speicherzustandes durch einen Programmierfehler und die Entdeckung dieser Verfälschung zeitlich beieinander liegen, desto schwieriger ist die Fehlersuche.
- ☐ Ein Programmierfehler kann, sobald er entdeckt wurde, durch einen Programmierer so behoben werden, dass er nicht wieder auftritt.
- ☐ Nicht jeder Programmierfehler führt zu einer Verfälschung des Speicherzustandes zur Ausführungszeit.
- ☐ Der einzig mögliche Umgang mit einem Umgebungsfehler ist der Abbruch des Programms.
- ☐ Ein Umgebungsfehler kann, sobald er entdeckt wurde, durch einen Programmierer so behoben werden, dass er nicht wieder auftritt.

15. ***Kennzeichnen Sie bei den folgenden Beispielen jeweils mit einem großen P oder einem großen U, ob es sich um einen Programmierfehler (P) oder um einen Umgebungsfehler (U) handelt.***

___ In einem Subtyp wird eine Vorbedingung verschärft.

___ Ein Mail-Programm bekommt keine Verbindung zu einem gültigen Mail-Host.

___ Beim Zugriff auf eine Liste wird ein ungültiger Index verwendet.

___ Bei einer Typzusicherung (Down-Cast) kommt es zu einer Exception.

___ Ein Benutzer macht in einem Formularfeld eine ungültige Eingabe.

___ Ein Programm versucht eine Datei zu lesen, die nicht existiert.

16. ***Welche der folgenden Ziele sollen durch eine Sprachunterstützung für Exceptions erreicht werden? (Mehrere richtige Antworten möglich)***

☐ Fehlerbehandlung soll den Normalfall nicht „verschütten".

☐ Programmierer sollen weniger Fehler machen.

☐ Fehler sollen nicht ignoriert werden können.

☐ Fehlerbehandlung soll effizienter werden.

☐ Der Benutzer einer Anwendung soll schneller korrekte Fehlermeldungen erhalten.

☐ Durch typisierte Exceptions soll je nach Fehlerart unterschiedlich reagiert werden können.

17. ***Welche der folgenden Aussagen zu Exceptions in Java sind richtig? (Mehrere richtige Antworten möglich)***

☐ Wenn eine assert-Anweisung zur Laufzeit ihren Ausdruck zu `false` auswertet, dann wird eine ungeprüfte Exception geworfen.

☐ Runtime-Exceptions wie `NullPointerException` oder `ArrayIndexOutOfBoundsException` sind ungeprüfte Exceptions.

☐ Ein Quelltextabschnitt, in dem eine geprüfte Exception geworfen werden könnte, muss immer in einem try-catch-Block stehen.

☐ Nur für ungeprüfte Exceptions besteht nach dem Auslösen die Möglichkeit, dass der normale Kontrollfluss zur Laufzeit nicht unterbrochen wird.

☐ Eine geprüfte Exception darf im Rumpf einer Methode `m` nur dann außerhalb eines try-Blocks geworfen werden, wenn dies im Kopf von `m` deklariert ist.

☐ Die Exception-Klassen in Java sollen immer zustandslose Exception-Exemplare definieren.

☐ Es können keine anwendungsspezifischen Exception-Klassen definiert werden.

☐ Eine Exception ist in Java ein Exemplar einer Exception-Klasse.

18. ***Welche der folgenden Aussagen gelten für Module in imperativen Programmiersprachen? (Mehrere richtige Antworten möglich)***

- ☐ Ein Modul definiert einen Namensraum.
- ☐ Jedes Modul ist gleichzeitig ein Typ.
- ☐ Die Importschnittstelle beschreibt, welche deklarierten und exportierten Merkmale ein Modul von seiner Umgebung benötigt.
- ☐ Ein Modul kapselt die Details seiner Implementation.
- ☐ Module und Klassen haben die gleichen Eigenschaften.
- ☐ Die Module imperativer Sprachen sind durch Klassen in objektorientierten Sprachen vollständig ersetzt worden.

19. ***Welche der folgenden Aussagen gelten für den Zugriffsschutz in Java? (Mehrere richtige Antworten möglich)***

- ☐ Ein Paket legt einen gemeinsamen Namensraum für die enthaltenen Klassen und Interfaces fest.
- ☐ Die Zugriffsrechte auf Top-Level Klassen und Interfaces werden vor dem Schlüsselwort `class` bzw. `interface` definiert: über die Modifikatoren `public`, `private`, `protected` oder ohne Modifikator (default-Zugriff).
- ☐ Ein Paket kann den Zugriff seiner Klienten auf die in ihm enthaltenen Pakete regeln.
- ☐ Ein als `protected` deklariertes Element (Variable oder Methode) einer als `public` deklarierten Klasse ist zugreifbar für Klassen im selben Paket und für Subklassen in anderen Paketen.
- ☐ Pakete können ineinander geschachtelt werden.
- ☐ Jede Klasse (jedes Interface) kann zu beliebig vielen Paketen gehören.

20. ***Wie viele unmittelbare Oberklassen kann eine benutzerdefinierte Klasse in Java haben? (1 richtige Antwort)***

- ☐ Keine außer `Object`.
- ☐ Entweder eine oder keine.
- ☐ Genau eine.
- ☐ Beliebig viele.

21. ***Wie nennt man das Ändern der Implementation einer Operation in einer Unterklasse, durch die eine andere Methode für dieselbe Operation eingesetzt wird? (1 richtige Antwort)***

- ☐ Überladen
- ☐ Redeklarieren
- ☐ Verifizieren
- ☐ Redefinieren

22. *Welche der folgenden Aussagen sind für Java korrekt? (Mehrere richtige Antworten möglich)*

- ☐ Eine abstrakte Methode darf keinen Rumpf definieren.
- ☐ Von einer Klasse, die eine als `final` deklarierte Methode enthält, darf keine Unterklasse gebildet werden.
- ☐ In einer abstrakten Klasse dürfen auch Exemplarvariablen als abstrakt deklariert werden.
- ☐ Eine Klasse, die eine abstrakte Methode enthält, muss selbst als abstrakt deklariert werden.
- ☐ Eine benutzerdefinierte Klasse erbt immer von der Klasse `Object`, entweder direkt oder indirekt.
- ☐ Aus dem Rumpf einer abstrakten Methode dürfen nur abstrakte Methoden aufgerufen werden.
- ☐ Eine Klasse darf nur als abstrakt deklariert werden, wenn sie mindestens eine abstrakte Methode enthält.
- ☐ Ein Exemplar einer Klasse A enthält unter anderem alle Exemplarvariablen, die in den Oberklassen von A definiert sind.

23. *Welche Ausgabe liefert das folgende Java-Programm? (1 richtige Antwort)*

```
public class A {
    A() { System.out.print("Alle wissen, was Qualität ist. "); }
}

public class B extends A {
    B() { System.out.print("Qualität ist undefinierbar. "); }
}

public class Test {
    public static void main(String[] args) {
        B b = new B();
        System.out.println();
    }
}
```

- ☐ "Qualität ist undefinierbar. "
- ☐ "Alle wissen, was Qualität ist. Qualität ist undefinierbar. "
- ☐ "Qualität ist undefinierbar. Alle wissen, was Qualität ist. "
- ☐ Es liefert lediglich einen Zeilenumbruch als Ausgabe.

24. Welche Aussagen zum folgenden Java-Programm sind richtig? (Mehrere richtige Antworten möglich)

```
public class A {
    public void m() { System.out.println("m in A"); }
    public static void s() { System.out.println("s in A"); }
}

public class B extends A {
    public void m() { System.out.println("m in B"); }
    public static void s() { System.out.println("s in B"); }
}

public class Test {
    public static void main(String[] args) {
        B b = new B();
        A a = b;
        System.out.println();  // letzte Anweisung
    }
}
```

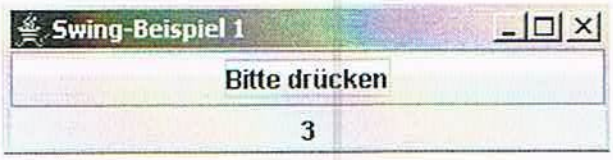
- ☐ Die Methode **m** ist in der Klasse **B** überladen.
- ☐ Die Methode **m** in der Klasse **B** redefiniert die aus **A** geerbte Operation.
- ☐ Die Methode **s** in der Klasse **B** redeklariert die aus **A** geerbte Operation.
- ☐ Die Anweisung **b.m();** anstelle der letzten Anweisung in der Methode **main** würde bei ihrer Ausführung die Ausgabe "m in B" bewirken.
- ☐ Die Anweisung **a.m();** anstelle der letzten Anweisung in der Methode **main** würde bei ihrer Ausführung die Ausgabe "m in A" bewirken.
- ☐ Die Anweisung **A.s();** anstelle der letzten Anweisung in der Methode **main** würde bei ihrer Ausführung die Ausgabe "s in A" bewirken.

25. Welche der folgenden Aussagen sind für die GUI-Programmierung mit AWT/Swing in Java korrekt? (Mehrere richtige Antworten möglich)

- ☐ Der Layout-Manager `BorderLayout` legt Layouts fest, bei denen maximal fünf GUI-Elemente als unmittelbare Kinder in der Komponentenhierarchie eingefügt werden können.
- ☐ Ein `JPanel` kann auch als Top-Level Container benutzt werden.
- ☐ Das Einbinden von anwendungsspezifischen Anweisungen in eine GUI erfolgt über das Anmelden von Implementationen so genannter Listener-Interfaces.
- ☐ Layouts in AWT/Swing (definiert durch Layout-Manager) können nicht ineinander geschachtelt werden.
- ☐ Ein `JFrame` definiert einen Top-Level Container.
- ☐ An einer GUI-Komponente wie beispielsweise einem `JButton` darf maximal ein Listener angemeldet werden.
- ☐ Die Implementation von Listener-Interfaces erfolgt in Java ausschließlich über anonyme innere Klassen.
- ☐ Die Swing-Komponenten setzen teilweise auf den AWT-Komponenten auf.
- ☐ Ein `JFrame` kann eine Menüleiste anbieten.
- ☐ Die Klasse `JButton` implementiert das Interface `ActionListener`.

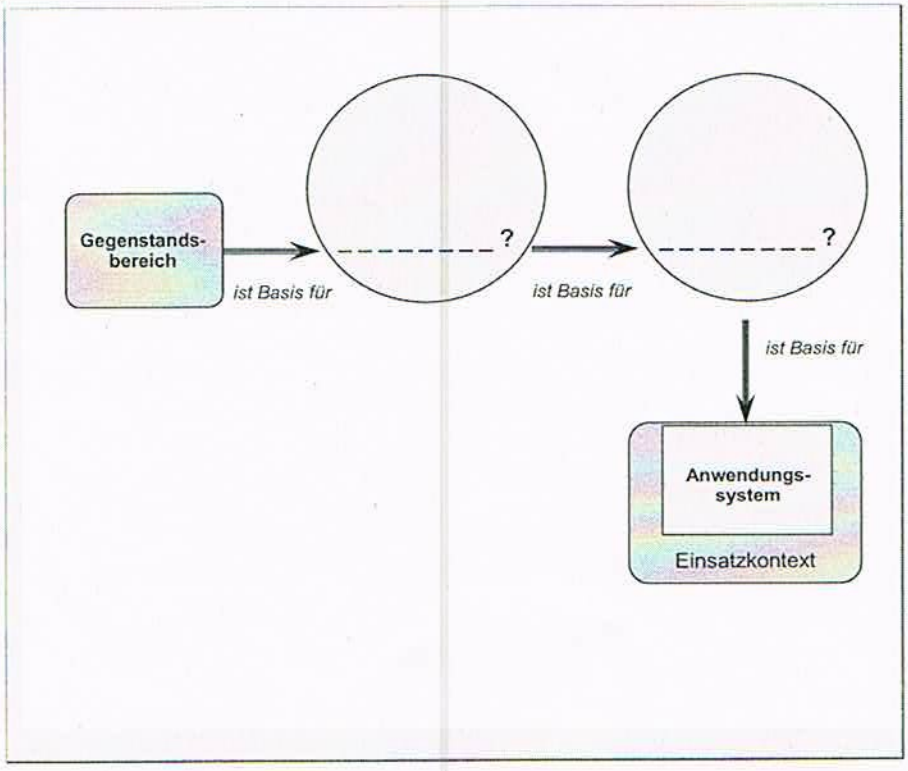
∑

26. *Der Einsatz welcher drei der genannten Klassen/Typen aus der Swing/AWT-Bibliothek ist wahrscheinlicher als der der anderen drei, um die folgende Swing-Oberfläche zu implementieren? (Drei richtige Antworten)*



- ☐ BoxLayout
- ☐ JFrame
- ☐ ActionListener
- ☐ JLabel
- ☐ ActionEvent
- ☐ JButton

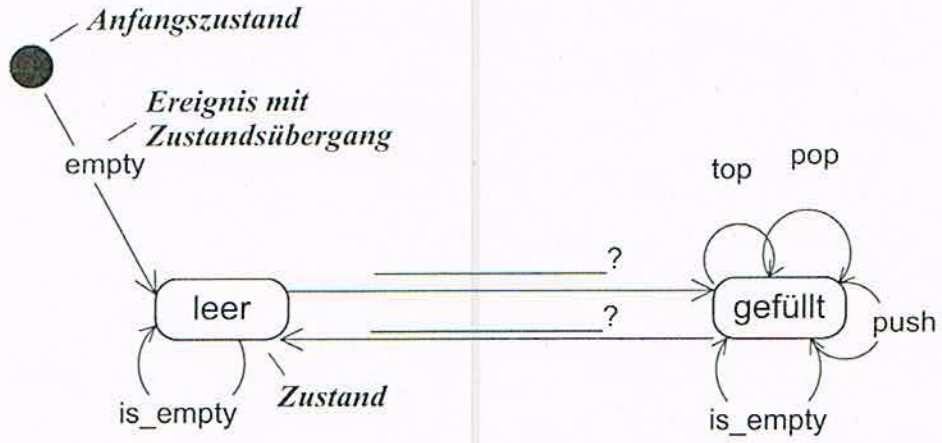
27. *Welche allgemeinen Modelle sind in der Softwareentwicklung wichtig? Beschriften Sie in der folgenden Abbildung die beiden kreisrunden Elemente mit den passenden Modellbezeichnungen auf den gestrichelten Linien.*



28. ***Welche der folgenden Aussagen zum Gegenstandsbereich sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Um die fachlichen Aufgaben zu identifizieren und zu verstehen, erstellen die Entwickler ein Klassenmodell des Gegenstandsbereichs.
- ☐ Bei der Modellierung des Gegenstandsbereichs ist der Einsatzkontext der Software irrelevant.
- ☐ Der Gegenstandsbereich kann eine Organisation, ein Bereich innerhalb einer Organisation oder ein Arbeitsplatz sein.
- ☐ Durch einen Gegenstandsbereich ist typischerweise auch eine entsprechende Anwendungsfachsprache festgelegt.

29. ***Vervollständigen Sie die Kantenbeschriftung des folgenden Zustandsdiagramms eines Stacks:***



30. ***Welche der folgenden Aussagen zu Begriffen des Werkzeug & Material-Ansatzes sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ *Werkzeuge* sind Gegenstände, mit denen Menschen im Rahmen einer Aufgabe *Materialien* verändern oder sondieren. Dies lässt sich auch gut für Anwendungssoftware umsetzen.
- ☐ *Materialien* sind die aktiven Einheiten in einem Softwaresystem.
- ☐ Umgangsformen kennzeichnen die Art und Weise, wie mit *Gegenständen* im Rahmen der verschiedenen *Aufgaben* gearbeitet wird.
- ☐ Eine direkte Abbildung von manuellen Werkzeugen in Software ist immer sinnvoll.
- ☐ *Materialien* sind Gegenstände, die im Rahmen einer Aufgabe Teil des Arbeitsergebnisses werden.
- ☐ Die eindeutige Trennung in *Werkzeug* und *Material* ist nur im Handwerk sinnvoll; bei Software unterscheiden wir *Oberfläche* und *Funktionen*.

Σ

31. ***Welche der folgenden Aussagen zu den in SE2 eingesetzten Entwurfsregeln für einfache interaktive Systeme sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Werkzeuge werden in eine Werkzeug- und eine UI-Klasse zerlegt.
- ☐ Die Werkzeug-Klasse eines Werkzeugs vermittelt zwischen den Services und den Materialien.
- ☐ Ein Werkzeug kann auf Services, Materialien und Fachwerte zugreifen.
- ☐ Nicht in jedem interaktiven System muss es Services geben.
- ☐ Materialien bilden die Grundkonstanten in einem interaktiven System.
- ☐ Ein Service sollte nicht auf andere Services zugreifen.
- ☐ In der Werkzeug-Klasse eines Werkzeugs sollte entschieden werden, wie mit einer fehlerhaften Benutzereingabe umzugehen ist.
- ☐ Werkzeuge sind die aktiven Einheiten in einem interaktiven System.
- ☐ Ein Material sollte die Werkzeuge kennen, die zu seiner Bearbeitung vorgesehen sind.
- ☐ Die UI-Klasse eines Werkzeugs sollte zur Schnittstelle ihres Pakets gehören.
- ☐ In der Schnittstelle einer Fachwertklasse sollten keine Materialien, Services oder Werkzeuge als Parameter auftauchen.
- ☐ Die Werkzeug-Klasse eines Werkzeugs ist zuständig für das Layout von GUI-Komponenten.

32. ***Welche der folgenden Aussagen zu Werten (im Vergleich zu Objekten) sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Es gibt genau eine Repräsentation für einen Wert.
- ☐ Ein Wert kann nicht kopiert werden.
- ☐ Eine Wertklasse kann nicht garantieren, dass es zu jedem ihrer modellierten Werte nur ein Exemplar gibt.
- ☐ Die primitiven Typen in Java sind Werttypen.
- ☐ Bei einer Wertklasse sollte verhindert werden, dass Unterklassen definiert werden können.
- ☐ Zu jedem Wert eines Werttyps darf es nur ein Exemplar der modellierenden Wertklasse geben.

33. ***Welche der folgenden Aussagen zu Objekten (im Vergleich zu Werten) sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Durch gemeinsame Verwendung von Objekten können Alias-Effekte auftreten.
- ☐ Objekte behalten auch bei Veränderungen ihre Identität.
- ☐ Objekte können erzeugt, aber nicht zerstört werden.
- ☐ Zwei gleiche Objekte sind automatisch auch identisch.
- ☐ Objekte können unveränderlich sein.
- ☐ Objekte können weder erzeugt noch zerstört werden.

34. *In Java sind die beiden unten als Quelltext aufgeführten Enumerationen* `Wochentag` *und* `Werktag` *implizit final, so dass sie nicht in einer Subtyp-Beziehung zueinander stehen können. Angenommen, eine Sprache würde Subtyp-Beziehungen zwischen solchen einfachen Enumerationen zulassen – welche der auf den Quelltext folgenden Aussagen zu den möglichen Typ-Beziehungen zwischen* `Wochentag` *und* `Werktag` *wäre richtig? (Eine richtige Antwort, 2 Punkte)*

```
enum Wochentag { MONTAG, DIENSTAG, MITTWOCH, DONNERSTAG, FREITAG,
                 SAMSTAG, SONNTAG }

enum Werktag { MONTAG, DIENSTAG, MITTWOCH, DONNERSTAG, FREITAG }
```

- ☐ `Wochentag` kann statisch typsicher als Subtyp von `Werktag` angesehen werden.
- ☐ `Werktag` kann statisch typsicher als Subtyp von `Wochentag` angesehen werden.
- ☐ Beide Subtyp-Beziehungen wären statisch typsicher.
- ☐ Beide Subtyp-Beziehungen wären nicht statisch typsicher.

35. *Welche der folgenden Aussagen zu Programmiersprachen sind richtig? (Mehrere richtige Antworten möglich)*

- ☐ Funktionale Programmiersprachen sind wertorientierte Programmiersprachen.
- ☐ Modula-2 ist eine objektorientierte Programmiersprache mit Modulen.
- ☐ Bei der Definition von Java wurde viel von C# abgeschaut.
- ☐ Jede objektorientierte Programmiersprache ist auch eine imperative Programmiersprache.

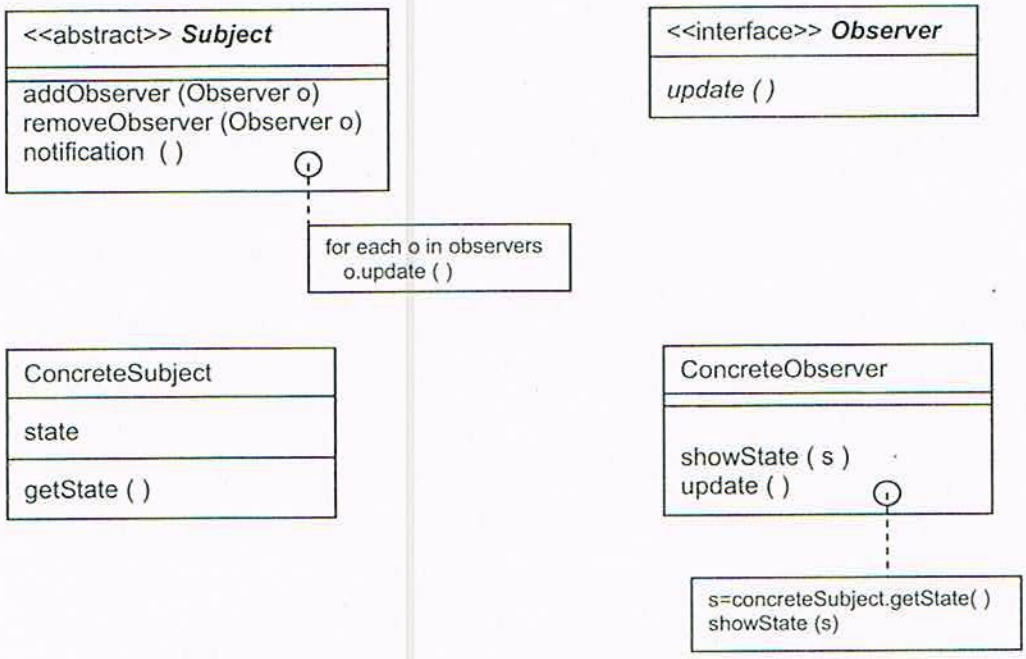
36. *Welche der folgenden Aussagen gelten für den objektorientierten Klassenentwurf? (Mehrere richtige Antworten möglich)*

- ☐ Wenn eine Klasse A die öffentlichen Exemplarvariablen einer anderen Klasse B benutzt, dann sind A und B eng gekoppelt.
- ☐ Je weniger Methoden eine Klasse enthält, desto besser ist sie strukturiert.
- ☐ Zwei Klassen, die sich gegenseitig benutzen, stehen in einer zyklischen Benutzt-Beziehung; manchmal kann dieser Zyklus statisch durch die Einführung eines Interfaces aufgelöst werden.
- ☐ Implizite Kopplung ist nicht formal nachweisbar; sie ist deshalb besonders unangenehm.
- ☐ Fünf Klassen, die sich alle gegenseitig benutzen, sind sehr lose gekoppelt.
- ☐ Wenn zwei Methoden in unterschiedlichen Klassen denselben Namen haben, sprechen wir von Code-Duplizierung.

37. *Welche der folgenden Aussagen gelten für Entwurfsmuster? (Mehrere richtige Antworten möglich)*

- ☐ Entwurfsmuster werden nicht erfunden, sondern entdeckt.
- ☐ Das Entwurfsmuster Proxy ist nach Gamma et al. ein Erzeugungsmuster.
- ☐ Entwurfsmuster sollten immer zusammen mit dem Problem beschrieben werden, das sie lösen sollen.
- ☐ Entwurfsmuster werden nach Gamma et al. unterteilt in Erzeugungsmuster, Klassenmuster und Objektmuster.

38. Zeichnen Sie in das folgende UML-Klassendiagramm des Beobachtermusters die gerichteten Beziehungen zwischen den Modellelementen mit den richtigen Pfeilspitzen und Linienarten. (4 mögliche Punkte)



39. Bei welchem Entwurfsmuster wird in der Implementation eine Sammlung benötigt? (1 richtige Antwort)

- ☐ Singleton
- ☐ Beobachter
- ☐ Abstrakte Fabrik
- ☐ Proxy

40. Welche der folgenden Aussagen gelten für Refactoring? (Mehrere richtige Antworten möglich)

- ☐ Beim Refactoring "Methode umbenennen" muss darauf geachtet werden, dass lokale Variablen, die im extrahierten Quelltext verändert werden, als Parameter übergeben werden.
- ☐ Eine Änderung der grafischen Benutzungsschnittstelle ist kein Refactoring.
- ☐ Eine gute Metapher für Refactoring ist „den Müll rausbringen", im Sinne von „sonst ersticken wir irgendwann an unseren Altlasten".
- ☐ Jede Änderung an einer Software, die nicht für den Benutzer sichtbar wird, ist ein Refactoring.
- ☐ Die Mechanik eines Refactorings beschreibt in kleinen und möglichst exakten Schritten, wie eine Restrukturierung durchzuführen ist.
- ☐ Die Mechanik von Refactorings muss von einem modernen Programmierer nicht mehr verstanden werden, weil gute IDEs seit einigen Jahren automatisierte Unterstützung für Refactorings anbieten.

41. Nennen Sie ein Refactoring, mit dem eine zu große Methode zergliedert werden kann.

Antwort: _____

Σ

42. ***Welche der folgenden Begriffserklärungen zum Thema Testen sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Der Klassentest besteht aus dem Test der einzelnen Operationen und ihrer Interaktionen im Kontext einer einzelnen Klasse.
- ☐ Funktionale Tests gibt es nur in funktionalen Programmiersprachen (z.B. Miranda).
- ☐ Im Integrationstest werden einzelne Komponenten im Zusammenspiel getestet.
- ☐ Lasttests prüfen das Verhalten technischer Systeme bei geladenen Treibern.

43. ***Welche der folgenden Ziele gelten für das Testen allgemein? (Mehrere Antworten möglich)***

- ☐ Testen soll Fehlerfreiheit nachweisen.
- ☐ Testen soll Aussagen zur Qualität ermöglichen.
- ☐ Testen ist eine Art eines formalen Korrektheitsbeweises.
- ☐ Testen soll Vertrauen und Verständnis unter den Entwicklern schaffen.

44. ***Welche der folgenden Aussagen zu metasprachlichen Eigenschaften einer Programmiersprache sind richtig? (Mehrere richtige Antworten möglich)***

- ☐ Ein objektorientiertes Programm lässt sich auch zum Gegenstandsbereich einer Modellierung machen.
- ☐ Die Menge der Schnittstellen aller Metaklassen werden als das Metaobjekt-Protokoll (MOP) einer Sprache bezeichnet.
- ☐ Das Metaobjekt-Protokoll (MOP) enthält alle Methodenaufrufe eines ablaufenden objektorientierten Programms.
- ☐ In einem Metaobjekt-Protokoll (MOP) werden die Methoden einer Klasse automatisch zu Parametern von Meta-Methoden.

Hamburg, den _____ _____

Unterschrift

∑

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

KLAUSUR
(Schlussblatt)

im Rahmen der Lehrveranstaltung: Softwareentwicklung 2

bei (Prüferin/Prüfer): Dr. Axel Schmolitzky
Prof. Dr.-Ing. Heinz Züllighoven

am: 01.08.2011 (SoSe 2011)

BEGRÜNDUNG DES LEHRENDEN UND NOTE:

Σ