Name: _____ Matrikelnummer: _____

Test 2 in Programmierparadigmen am 26. 1. 2026

Beispiele für **markierte (= ausgewählte)** Auswahlfelder: ⊗ ☑ ☒⊗
Beispiele für **nicht markierte (= nicht ausgewählte)** Auswahlfelder: ○ ☒ ☒○

Beurteilung: 1 Punkt pro Zeile mit gänzlich richtig markierten Auswahlfeldern

Aufgabe 1 (10 Punkte)

`A` und `B` seien definiert durch `interface A<T> {}` und `interface B<R, S> extends A<S> {}`.
Bitte markieren Sie jedes Auswahlfeld, bei dem der links stehende Typ ein Untertyp des darüber stehenden Typs ist. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	`A<String>`	`A<? super String>`	`B<? extends String, ?>`
`A<Object>`	○	○	○
`A<String>`	○	○	○
`B<String, Object>`	○	○	○
`B<Object, String>`	○	○	○
`B<?, ? extends String>`	○	○	○
`B<? extends String, ?>`	○	○	○
`B<? super String, ? extends String>`	○	○	○
`B<? extends String, ? super String>`	○	○	○
`B<A<String>, Object>`	○	○	○
`B<A<Object>, String>`	○	○	○

Aufgabe 2 (10 Punkte)

Bitte markieren Sie in jeder Zeile das eine Auswahlfeld, bei dem die links stehende Aussage am ehesten eine Eigenschaft der darüber stehenden Parametrisierungsform in Java oder AspectJ ist.

	Generizität	Annotationen	Aspekte
Join-Points sind dynamisch identifizierbare Programmstellen	○	○	○
kann die Werkzeuge zur Programmerzeugung beeinflussen	○	○	○
„`*`“ steht für beliebige Zeichen außer „`.`“ (Punkt)	○	○	○
für binäre Methoden wird Rekursion benötigt	○	○	○
zur Laufzeit sind Daten über Reflexion zugreifbar	○	○	○
wird für Kernfunktionalität meist gemieden	○	○	○
Lücken werden durch spezielle Parameter dargestellt	○	○	○
`around()` ersetzt den spezifizierten Porgammtext	○	○	○
mit `@Retention` wird die Lebensdauer festgelegt	○	○	○
Typschranken legen vorausgesetzte Funktionalität fest	○	○	○

Aufgabe 3 (10 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem der links stehende Typausdruck (mit Typnamen aus den Paketen `java.util.function` und `java.lang`) ein Typ des darüber stehenden Lambdas ist. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	`x->y->x+y`	`(x,y)->x+y`	`x->x+2`	`x->{}`	`Long::sum`
`BinaryOperator<String>`	○	○	○	○	○
`LongBinaryOperator`	○	○	○	○	○
`BiFunction<Long,Long,Long>`	○	○	○	○	○
`Function<Long,Function<Long,Long>>`	○	○	○	○	○
`Function<Long,LongUnaryOperator>`	○	○	○	○	○
`LongUnaryOperator`	○	○	○	○	○
`Consumer<String>`	○	○	○	○	○
`Function<Long,Function<String,String>>`	○	○	○	○	○
`BiFunction<Long,String,Long>`	○	○	○	○	○
`BiFunction<String,Long,String>`	○	○	○	○	○

Aufgabe 4 (10 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem die links stehende Kommandozeile (in `bash` ausgeführt) die darüber stehende Auswirkung hat. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	mehrere Prozesse laufen gleichzeitig	Prozesse laufen im Hintergrund	Prozesse über Pipeline verbunden	Standardeingabe von `cat` umgeleitet	Standardausgabe von `cat` umgeleitet	Fehlerausgabe von `cat` umgeleitet
`cat < a | wc > b`	○	○	○	○	○	○
`cat a |& wc &>> b`	○	○	○	○	○	○
`cat a &> b && wc b`	○	○	○	○	○	○
`cat a ; wc a`	○	○	○	○	○	○
`for i in * ; do cat $i \| wc > ../wc/$i ; done`	○	○	○	○	○	○
`for i in * ; do ( cat $i > ../bak/$i & ) ; done`	○	○	○	○	○	○
`if test 'cat a' = "a b c" ; then wc b ; else wc c ; fi`	○	○	○	○	○	○
`cat a || wc b`	○	○	○	○	○	○
`cat < a |& wc &> b &`	○	○	○	○	○	○
`cat | wc`	○	○	○	○	○	○

Aufgabe 5 (10 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem die links stehende Eigenschaft auf den darüber stehenden Methodenaufruf zutrifft. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	`Thread.sleep(5)`	`wait()`	`notify()`	`notifyAll()`
muss mit `InterruptedException` umgehen	○	○	○	○
darf nur in einem Monitor verwendet werden	○	○	○	○
soll nicht in einem Monitor verwendet werden	○	○	○	○
gehört nicht zu Synchronisationsmechanismen	○	○	○	○
suspendiert Ausführung des aktuellen Threads	○	○	○	○
suspendiert Ausführung auf bestimmte Zeit	○	○	○	○
weckt systemweit alle wartenden Threads auf	○	○	○	○
weckt höchstens einen wartenden Threads auf	○	○	○	○
weckt mindestens einen wartenden Thread auf	○	○	○	○
gibt den Lock vorübergehend frei	○	○	○	○

Aufgabe 6 (10 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem die links stehende Aussage eine Eigenschaft des darüber stehenden Entwurfsmusters ist. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	Iterator	Decorator	Proxy	Prototype	Factory-Method
unterstützt Umkehrung der Abhängigkeiten	○	○	○	○	○
führt zu vielen kleinen Objekten	○	○	○	○	○
für oberflächliche Erweiterungen geeignet	○	○	○	○	○
oft große Anzahl an Unterklassen nötig	○	○	○	○	○
ist Entwurfsmuster für Struktur	○	○	○	○	○
ist erzeugendes Entwurfsmuster	○	○	○	○	○
hilft große Zahl an Klassen zu vermeiden	○	○	○	○	○
beruht auf Delegation	○	○	○	○	○
häufig als innere Klasse implementiert	○	○	○	○	○
Objektidentität ist damit unzuverlässig	○	○	○	○	○

Aufgabe 7 (10 Punkte)

Bitte markieren Sie in jeder Zeile das eine Auswahlfeld, bei dem die links stehende Aussage am ehesten eine Eigenschaft des darüber stehenden Entwurfsmusters ist.

	Iterator	Decorator	Proxy	Prototype	Factory-Method
zyklische Strukturen bereiten Probleme	○	○	○	○	○
robuste Varianten werden bevorzugt	○	○	○	○	○
Verantwortlichkeiten wieder entziehbar	○	○	○	○	○
schlecht geeignet für umfangreiche Objekte	○	○	○	○	○
mehrere gleichzeitige Abarbeitungen möglich	○	○	○	○	○
kein Decorator, aber gleiche Struktur möglich	○	○	○	○	○
kann mit kovarianten Problemen umgehen	○	○	○	○	○
es gibt externe und interne Varianten	○	○	○	○	○
flache von tiefen Kopien unterschieden	○	○	○	○	○
führt zu parallelen Klassenhierarchien	○	○	○	○	○

Aufgabe 8 (10 Punkte)

Bitte markieren Sie in jeder Zeile das eine Auswahlfeld, bei dem die links stehende Aussage am ehesten eine Eigenschaft des darüber stehenden Entwurfsmusters ist.

	Iterator	Decorator	Proxy	Prototype	Factory-Method
Component ist ein Bestandteil	○	○	○	○	○
Product ist ein Bestandteil	○	○	○	○	○
Subject ist ein Bestandteil	○	○	○	○	○
Creator ist ein Bestandteil	○	○	○	○	○
Aggregate ist ein Bestandteil	○	○	○	○	○
Smart-Reference ist eine Variante davon	○	○	○	○	○
wird auch *Wrapper* genannt	○	○	○	○	○
wird auch *Surrogate* genannt	○	○	○	○	○
wird auch *Virtual-Constructor* genannt	○	○	○	○	○
wird auch *Cursor* genannt	○	○	○	○	○

Aufgabe 9 (10 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem die links stehende Aussage eine Eigenschaft des darüber stehenden Entwurfsmusters ist. Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	Visitor	Singleton	Template-Method	Factory-Method
verwaltet verwandte Operationen zentral	○	○	○	○
für Lazy-Initialization geeignet	○	○	○	○
gibt Grundgerüst von Algorithmus vor	○	○	○	○
Anzahl erzeugter Objekte kontrollierbar	○	○	○	○
kann zu vielen Klassen führen	○	○	○	○
kann zu vielen Methoden führen	○	○	○	○
verwendet häufig Hooks	○	○	○	○
kann zu vielen Objekten führen	○	○	○	○
mehrere Arten primitiver Operationen	○	○	○	○
beruht auf dem Hollywood-Prinzip	○	○	○	○

Aufgabe 10 (10 Punkte)

Bitte markieren Sie in jeder Zeile das eine Auswahlfeld, bei dem die links stehende Aussage am ehesten eine Eigenschaft des darüber stehenden Entwurfsmusters ist.

	Visitor	Singleton	Template-Method	Factory-Method
accept hat einen Parameter	○	○	○	○
instance ist meist `static`	○	○	○	○
Element ist ein Bestandteil	○	○	○	○
hilft Duplikate im Code zu vermeiden	○	○	○	○
kann als Anti-Pattern gesehen werden	○	○	○	○
Lambdas können Schreibaufwand reduzieren	○	○	○	○
eine Methode ist häufig `final`	○	○	○	○
globale Variable ist Alternative dazu	○	○	○	○
unterstützt direkte Code-Wiederverwendung	○	○	○	○
Umgang mit Unterklassen problematisch	○	○	○	○