

Name: Matrikelnummer: 

## Test 2 in Programmierparadigmen am 22. 1. 2024

Beispiele für **markierte** (= ausgewählte) Auswahlfelder:Beispiele für **nicht markierte** (= nicht ausgewählte) Auswahlfelder:

Beurteilung: 1 Punkt pro Zeile mit gänzlich richtig markierten Auswahlfeldern,  
1 Punkt pro richtigem Begriff oder Ausdruck in einem Textfeld

### Aufgabe 1 (30 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
beruht auf <i>Delegation</i>	<input type="checkbox"/>				
verwendet häufig <i>Hooks</i>	<input type="checkbox"/>				
<i>Subject</i> ist ein Bestandteil	<input type="checkbox"/>				
<i>Creator</i> ist ein Bestandteil	<input type="checkbox"/>				
<i>Product</i> ist ein Bestandteil	<input type="checkbox"/>				
wird auch <i>Wrapper</i> genannt	<input type="checkbox"/>				
wird auch <i>Surrogate</i> genannt	<input type="checkbox"/>				
<i>Component</i> ist ein Bestandteil	<input type="checkbox"/>				
ist erzeugendes Entwurfsmuster	<input type="checkbox"/>				
ist Entwurfsmuster für Struktur	<input type="checkbox"/>				
führt zu vielen kleinen Objekten	<input type="checkbox"/>				
auch auf leeren <i>Aggregaten</i> sinnvoll	<input type="checkbox"/>				
<i>robuste</i> Varianten werden bevorzugt	<input type="checkbox"/>				
flache und tiefe Kopien unterschieden	<input type="checkbox"/>				
es gibt interne und externe Varianten	<input type="checkbox"/>				
führt zu parallelen Klassenhierarchien	<input type="checkbox"/>				
häufig als innere Klasse implementiert	<input type="checkbox"/>				
Objektidentität ist damit unzuverlässig	<input type="checkbox"/>				
oft große Anzahl an Unterklassen nötig	<input type="checkbox"/>				
wird auch <i>Virtual-Constructor</i> genannt	<input type="checkbox"/>				
zyklische Strukturen bereiten Probleme	<input type="checkbox"/>				
<i>Smart-Reference</i> ist eine Variante davon	<input type="checkbox"/>				
auch für kovariante Probleme einsetzbar	<input type="checkbox"/>				
hilft große Zahl an Klassen zu vermeiden	<input type="checkbox"/>				
für oberflächliche Erweiterungen geeignet	<input type="checkbox"/>				
auch für entziehbare Verantwortlichkeiten	<input type="checkbox"/>				
schlecht geeignet für umfangreiche Objekte	<input type="checkbox"/>				
kann gleiche Struktur wie <i>Decorator</i> haben	<input type="checkbox"/>				
unterstützt Umkehrung der Abhängigkeiten	<input type="checkbox"/>				
mehrere gleichzeitige Abarbeitungen möglich	<input type="checkbox"/>				

### Aufgabe 2 (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.

	Factory-Method	Template-Method	Visitor	Singleton
Methode <i>accept</i> kommt vor	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Anzahl der Objekte kontrollierbar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
häufig als Anti-Pattern angesehen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<i>Hollywood-Prinzip</i> spielt eine Rolle	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
sehr viele Klassen können entstehen	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
sehr viele Objekte können entstehen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
sehr viele Methoden können entstehen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
verwandte Operationen zentral verwaltet	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
kann mit kovarianten Problemen umgehen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<i>Hooks</i> von <i>abstrakten Methoden</i> unterschieden	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### Aufgabe 3 (10 Punkte)

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

	Annotationen	Aspekte	Generizität
T+ steht für jeden Untertyp eines Typs T	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
dynamisch nur über <i>Reflexion</i> verwendbar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
drei Arten von <i>Advice</i> werden unterschieden	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<i>gebundene Wildcards</i> erhöhen die Flexibilität	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ideal geeignet für <i>Querschnittsfunktionalitäten</i>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<code>args(...)</code> hilft beim Sammeln von Kontextinformation	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
implizite Untertypbeziehungen werden nicht unterstützt	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Typkonsistenz wird nur statisch durch <i>Typinferenz</i> überprüft	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
über eine <code>default</code> -Klausel lassen sich Default-Werte angeben	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
binäre Methoden sind über rekursive Deklarationen ausdrückbar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### Aufgabe 4 (10 Punkte)

Bitte markieren Sie pro Zeile das eine entsprechende Auswahlfeld, bei dem die links stehende Aussage im Zusammenhang mit Softwareentwurfsmustern (kurz *Muster* genannt) immer, manchmal oder nie zutrifft.

	immer	manchmal	nie
die <i>Programmstruktur</i> zeigt, um welches Muster es sich handelt	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Muster können Ideen hinter dem Softwareentwurf sehr gut ersetzen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Muster sollen zur Einschätzung von Konsequenzen eingesetzt werden	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Muster helfen dabei, persönliche Expertisen auf Konsistenz zu prüfen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
der Begriff <i>Anti-Pattern</i> ist nur bei <i>nicht-wertender Sichtweise</i> sinnvoll	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
der Einsatz von Mustern ist wichtiger als die Einhaltung von Konventionen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Software ist besser, wenn mehr Muster und weniger <i>Anti-Pattern</i> vorkommen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
um Konsequenzen zu gewichten, sind <i>wertende Sichtweisen</i> von Mustern nötig	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Konsequenzen eines Musters treffen auf alle Implementierungen des Musters zu	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
etablierte Muster sind zuverlässig, da lange Diskussionsprozesse dahinterstehen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### Aufgabe 5 (15 Punkte)

Bitte markieren Sie jedes Auswahlfeld, bei dem die links stehende Kommandozeile, in `bash` fehlerfrei ausgeführt, die darüber stehende Auswirkung hat. Dabei sind die Auswirkungen folgendermaßen beschrieben:

- `back` mindestens ein Prozess wird im Hintergrund gestartet
- `pipe` mindestens eine Pipeline wird angelegt
- `args` mindestens ein Kommandozeilenargument wird übergeben
- `in` mindestens eine Standardeingabe wird zu einer Datei (nicht Pipeline) umgeleitet
- `out` mindestens eine Standardausgabe wird zu einer Datei (nicht Pipeline) umgeleitet
- `err` mindestens eine Fehlerausgabe wird zu einer Datei (nicht Pipeline) umgeleitet

Es können keines, eines oder mehrere Felder pro Zeile auszuwählen sein.

	<u>back</u>	<u>pipe</u>	<u>args</u>	<u>in</u>	<u>out</u>	<u>err</u>
cat a	<input type="checkbox"/>					
cat a >> b	<input type="checkbox"/>					
cat < a   wc > b	<input type="checkbox"/>					
cat *.java    cat b	<input type="checkbox"/>					
cat a &> b	<input type="checkbox"/>					
cat   wc &	<input type="checkbox"/>					
cat   wc &> a	<input type="checkbox"/>					
cat a &>> b &	<input type="checkbox"/>					
cat < a &>> b	<input type="checkbox"/>					
cat a && cat b	<input type="checkbox"/>					
cat a   wc > b &	<input type="checkbox"/>					
cat < a &  wc > b &	<input type="checkbox"/>					
if test a = a ; then cat < a ; fi	<input type="checkbox"/>					
for i in a b c ; do cat \$i ; done	<input type="checkbox"/>					
for i in a b c ; do ( cat \$i > \$i.bak & ) ; done	<input type="checkbox"/>					

### Aufgabe 6 (13 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.

	<u>()-&gt;?L</u>	<u>s-&gt;s+1</u>	<u>(s,t)-&gt;s+t</u>	<u>s-&gt;t-&gt;s+t</u>	<u>Long::sum</u>
BiFunction<Long,Long,Long>	<input type="checkbox"/>				
BiFunction<Long,String,Long>	<input type="checkbox"/>				
BiFunction<Long,String,String>	<input type="checkbox"/>				
BiFunction<String,Long,String>	<input type="checkbox"/>				
BiFunction<String,String,String>	<input type="checkbox"/>				
Function<Long,Function<Long,Long>>	<input type="checkbox"/>				
Function<String,Function<Long,String>>	<input type="checkbox"/>				
LongSupplier	<input type="checkbox"/>				
LongBinaryOperator	<input type="checkbox"/>				
LongUnaryOperator	<input type="checkbox"/>				
Supplier<String>	<input type="checkbox"/>				
BinaryOperator<String>	<input type="checkbox"/>				
UnaryOperator<String>	<input type="checkbox"/>				

### Aufgabe 7 (12 Punkte)

Beantworten Sie folgende Fragen bitte mit je drei Ausdrücken oder Begriffen. Vermeiden Sie Umschreibungen, Beschreibungen oder Erklärungen. Es werden jeweils nur die ersten drei Ausdrücke oder Begriffe gewertet.

Geben Sie bitte drei *rekursive Typparameterdeklarationen* an, also Ausdrücke, mit denen in Java rekursive Typparameter deklariert werden können. Sie sollen verschiedene Arten *gebundener Wildcards* enthalten:

Geben Sie bitte drei verschiedene Namen von in Java standardmäßig vordefinierten Annotationen an. Achten Sie dabei bitte auf Java-konforme Groß- und Kleinschreibung:

Geben Sie bitte drei verschiedene Namen von Methoden an, die in `java.util.stream.Stream<T>` deklariert sind und zu den *Stream-abschließenden Operationen* zählen:

Geben Sie bitte drei verschiedene Namen von Methoden an, die in `java.util.stream.Stream<T>` deklariert sind und zu den *Stream-modifizierenden Operationen* zählen: