Otto-von-Guericke Universität Magdeburg
Fakultät für Informatik
Prof. Dr. R. Dumke, 23. Juli 2009

Prüfungsklausur Programmierung

Bewertung

Aufgabe 1 (4 Punkte): *Was versteht man unter Programmierungstechnik. Nennen Sie zwei Beispiele und jeweils einen dafür benötigten Software-Prozessor.*

Lösung:

Emulation: Crosscompiler **1 P, 1 P**

Entwickluerdokumentation: JavaDoc **1 P, 1 P**

Aufgabe 2 (6 Punkte): *Beschreiben Sie den folgenden Java-Ausdruck in der BNF und als Syntaxgraph beginnend mit dem Nichtterminal <ausdruck> bzw. <expression>.*

```
var1/2 * var2 - var3
```

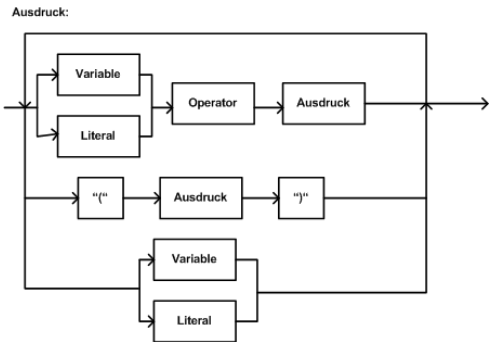
Lösung:

BNF: E==>*Field* | *Literal* | E *Infix* E | *Prefix* E | E *Postfix* | *Conditional_E* | *Other_E*.

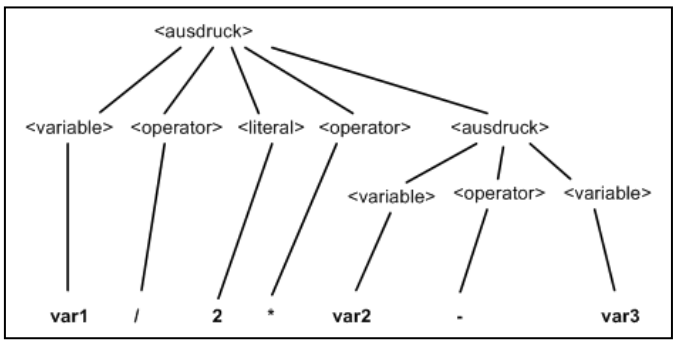
3 P

oder (ausgewählt): <ausdruck> ::= <literal> | <variable> |
(<ausdruck> <operator> <ausdruck>)|(<operator><ausdruck>)
und speziell: <ausdruck>::= <variable><op> <literal> <op> <variable> <op><variable>

Syntaxgraph: z. B. allgemein



oder ganz speziell; oder auch als Syntaxbaum: **3 P**



Aufgabe 3 (5 Punkte): *Welches sind korrekte Zahlenkonvertierungen in Java?*

a) `double` → `short` b) `float` → `char` c) `byte` → `short`
d) `boolean` → `int` e) `long` → `float`

Lösung: korrekt nur c, bis Java 1.3 auch e (deshalb hier zugelassen)
andere falsch bzw. fehlerhaft **je 1 P**

Aufgabe 4 (4 Punkte): *Was ist der Unterschied zwischen Algorithmenkorrektheit und Programmkorrektheit? Beschreiben Sie kurz zwei statische Programmtestmethoden.*

Lösung:

Algorithmenkorrektheit: (formal) beweisbar **1 P**

Programmkorrektheit: nicht beweisbar; folgt: (systematisches) Testen **1 P**

Programmverifikation: formaler Korrektheitsbeweis **1 P**

Review: Durchsicht von Programmen von Kollegen nach allg. zumeist firmenspezfischen Prinzipien **1 P**

(aber auch: Checkliste oder symbolische Abarbeitung)

Aufgabe 5 (6 Punkte): *Was ist der Unterschied zwischen Überladen und Überschreiben von Methoden in Java? Ordnen Sie die jeweilige Form der Verwendung folgender Java-Elemente zu:*

a) abstrakte Klasse, b) polymorphe Methode einer Klasse,
c) Methoden-Signaturerweiterung, d) Java-Interface.

Lösung:

a) Überschreiben **1 P**

b) Überladen **1 P**

c) Überladen **1 P**

d) Überschreiben **1 P**

Überschreiben: „Wirkung der Vererbung“ **1 P**

Überladen: unterschiedliche Signaturen für dieselbe Methode **1 P**

Aufgabe 6 (5 Punkte): *Erklären Sie kurz drei vordefinierte Annotationen. Erläutern Sie zwei Javadoc-Annotationen, die im Java-Quelltext die Programmdokumentation entsprechend erweitern.*

Lösung:

@Retention: definiert Annotationslebensdauer **1 P**

@Deprecated: kennzeichnet (u.U.) veraltetes Java **1 P**

@Documented: definierter JavaDoc-Bereich **1 P**
usw.

@author: Autorenangabe **1 P**

@param: explizite Parameterkennzeichnung **1 P**
usw.

Aufgabe 7 (4 Punkte): *Beschreiben Sie kurz die drei Grundkomponenten der MVC-GUI-Technologie. Wozu dient der Look-And-Feel-Ansatz?*

Lösung:

Model: Applikation, Daten	**1 P**
View: Präsentation, GUI	**1 P**
Controller: Interaktionssteuerung	**1 P**
Look-And-Feel: Plattformzuschneidung des View	**1 P**

Aufgabe 8 (8 Punkte): *Ordnen Sie die folgenden Methoden den jeweiligen Java-Klassen zu:* `getAdler()`, `setText()`, `matches()`, `getPort()`, `mkdir()`, `isAlive()`, `isEmpty()`, `nextInt()`.

a) `File`	b) `TextField`	c) `Pattern`	d) `Thread`
e) `Hashtable`	f) `Deflater`	g) `URL`	h) `Scanner`

Lösung:

a) File:	mkdir()	**1 P**
b) TextField:	setText()	**1 P**
c) Pattern:	matches()	**1 P**
d) Thread:	isAlive()	**1 P**
e) Hashtable:	isEmpty()	**1 P**
f) Deflater:	getAdler()	**1 P**
g) URL:	getPort()	**1 P**
h) Scanner:	nextInt()	**1 P**

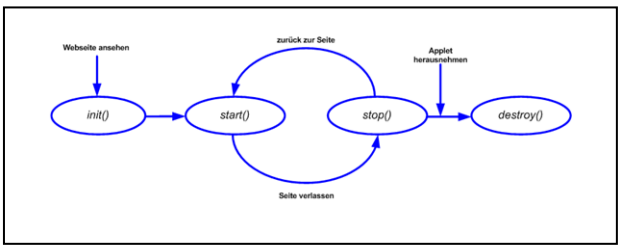
Aufgabe 9 (4 Punkte): *Erläutern Sie kurz die OO-Qualitätsindikatoren „Generationskonflikt“ und „Gottklasse“. Beschreiben Sie kurz zwei Formen der Java-Programmoptimierung.*

Lösung:

Generationskonflikt: direkte Unterklassen der Oberklasse definieren mehr als 50% neu	**1 P**
Gottklasse: *eine Klasse deklariert mehr als 50 Attribute (unabhängig von Sichtbarkeit und Typ)*	**1 P**
*das **Systemverhalten** des möglicherweise vorhandenen Systems, dessen Teil das Programm darstellt,*	**1 P**
*das **Leistungsverhalten** des Programms selbst und*	**1 P**
*das **Ressourcenverhalten** des jeweiligen Rechners bzw. des Betriebssystems*	usw.

Aufgabe 10 (7 Punkte): *Erklären Sie vier unterschiedliche Zustände eines Java-Applets. Erläutern Sie kurz die Bedeutung der Interfaces Comparable, Iterable und Readable für eine Datentyperweiterung.*

Lösung:



sowie ganz kurze Erläuterung!

4 P

Vergleichen: *Die Werte zwischen den Wrapper-Klassen können miteinander verglichen werden (Mehode compareTo), da sie das Interface* ***Comparable*** *implementeren. Diese Vergleichsform geht von einer „natürlichen“ Ordnung der Objekte aus.*

1 P

Folgestruktur: *Durch den Bezug zum Interface* ***Iterable*** *kann die jeweilige Datenstruktur als eine Folgestruktur verwendet werden, deren Werte der Reihe nach verwendet (Methode iterator) bzw. geordnet werden können.*

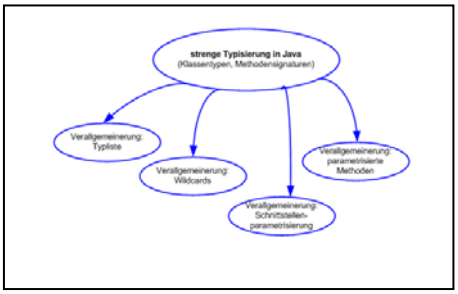
1 P

Lesbarkeit: *Um die Werte einer Datenstruktur durch so genannte Puffer-Reader lesbar zu Machen (Methode read), kann das Interface Readable benutzt werden.*

1 P

Aufgabe 11 (5 Punkte): *Charakterisieren Sie kurz die vier Formen generischer Typbildung und -anwendung. Welche Aufgabe hat der Garbage Collector in Java?*

Lösung:



4 P

Die 6 formalen Darstellungsformen sind Anwendungen dieser Grundformen und werden natürlich auch angerechnet!

Garbage Collector: bereinigt Objektsituation im laufenden Java-Programm **1 P**

Aufgabe 12 (2 Punkte): *Nennen Sie jeweils einen Grund für die Einbettung eines Java-Programms in das Programm einer anderen Programmiersprache und umgekehrt.*

Lösung:

Java in anderes: Web-Eigenschaften **1 P**

Anderes in Java: Programmeffizienz **1 P**