# Fragen zur Klausurvorbereitung

Sonntag, 21. Februar 2021

### 1. Zahlensysteme

a) Wie sind positive ganze Zahlen in Java realisiert?

Im Zweierkomplement?

Das Zweierkomplement definiert für positive und negative Ganzzahlen folgende Kodierung:

- Das Vorzeichen einer Zahl bestimmt ein Bit, das 1 bei negativen und 0 bei positiven
- Um eine 0 darzustellen, ist kein Bit gesetzt.

Java kodiert die Ganzzahldatentypen byte, short, int und long immer im Zweierkomplement (der Datentyp char definiert keine negativen Zahlen). Mit dieser Kodierung gibt es eine negative Zahl mehr als positive, da es im Zweierkomplement keine positive und negative 0 gibt, sondern nur eine »positive« mit der Bit-Maske 0000...0000.

Aus < http://openbook.rheinwerk-verlag.de/javainsel/22 001.html>

Durch den Datentyp int werden ganze Zahlen repräsentiert. Für eine int-Zahl werden 4 Byte zur Abspeicherung verwendet; durch sie werden alle ganzen Zahlen von -2<sup>31</sup> bis 2<sup>31</sup>-1 dargestellt. Kommt man mit diesem Zahlenbereich nicht aus, kann man den Grunddatentyp long verwenden; kleinere Zahlenbereiche werden durch die Grunddatentypen short, byte erfasst.

- b) Wie sind negative ganze Zahlen in Java realisiert?
- c) Wie sind Gleitkommazahlen in Java realisiert?

float, double

- d) Welchen Zahlenbereich decken die Zahltypen von Java ab?
- -1.7\*10^308 bis 1.7\*10^308

Der größere Fließkomma-Datentyp wird als double bezeichnet. Double besitzt eine Größe von 64 Bit und der Wertebereich erstreckt sich von -1,7 \* 10^308 bis 1,7 \* 10^308 - Kapselung: Daten (Klassen, Methoden, Variablen) können in Java mit Hilfe von https://www.java-tutorial.org/datentypenundvariablen.html Modifikatoren (modifier; private/public/protected) gekapselt werden, damit der Zugriff

Тур	Wertebereich	Länge
byte	-128127	8 Bit
short	-3276832767	16 Bit
int	-21474836482147483647	32 Bit
long	-92233720368547758089223372036854775807	64 Bit

### 2. Zeichencodes

a) Geben Sie die binäre Darstellung eines U als ASCII, Unicode, UTF-8 und UTF-16 an.

ASCII: 0101 0101 (binär) Unicode: U+0055 UTF-8: 0101 0101

UTF-16: 0000 0000 0101 0101

b) Geben Sie die binäre Darstellung eines Ü als Unicode, UTF-8 und UTF-16 an.

ASCII: 1101 1100 (binär) Unicode: U+00DC UTF-8: 1100

## 3. Datentypen

a) Nennen Sie alle primitiven Datentypen von Java.

new Integer(1234).intValue(); (Methode intValue ist auch in der Oberklasse Number definiert und es wird erst zur

new StringBuilder().append("Aufgabe ").append((i + 1)).toString();

Größe<sup>[1]</sup> Wrapper-Klasse Wertebereich java.lang.Boolear Boolescher Wahrheitswert, Boolescher Typ[3] java.lang.Character 0 ... 65.535 (z. B. 'A') Unicode-Zeichen (UTF-16) cha 8 bit -128 ... 127 byte java.lang.Byte Zweierkomplement-Wert -32.768 ... 32.767 short 16 bit java.lang.Short Zweierkomplement-Wert int 32 bit iava.lang.Integer -2.147.483.648 ... 2.147.483.647 Zweierkomplement-Wert long java.lang.Long  $-2^{83}$  bis  $2^{83}$ -1, ab Java 8 auch 0 bis  $2^{84}$  -1<sup>[4]</sup> 64 bit Zweierkomplement-Wert 32 bit +/-1,4E-45 ... +/-3,4E+38 32-bit IEEE 754, es wird empfohlen, diesen Wert nicht für Programme zu verwenden, die sehr genau rechnen müssen. java.lang.Float 64 bit +/-4,9E-324 ... +/-1,7E+308

b) Welche Referenztypen gibt es in Java?

Objekte, Strings und Arrays. Außerdem die vordefinierte Konstante null, die eine leere Referenz hezeichnet.

In Folie 2-11:

0-	01			π	Ψ	70	<u> </u>				
05	(	)	*	+	,	-		/			
06	0	1	2	3	4	5	6	7			
07	8	9	:	;	<	=	>	?			
10	@	Α	В	С	D	Е	F	G			
11	Н	ı	J	K	L	М	N	0			
12	Р	Q	R	S	Т	U	V	W			
13	Х	Υ	Z	[	\	]	۸	_			
14	`	а	b	С	d	е	f	g			
15	h	i	j	k	_	m	n	0			
16	р	q	r	S	t	u	٧	w			
17	Х	у	Z	{		}	~	DEL			
• 1-Byte-Zeichen: 0xxxxxxx											
• 2-B	• 2-Byte-Sequenz: 110xxxxx 10xxxxxx										

3

ETX

VT

DC3

**ESC** 

#

4

**EOT** 

NP

DC4

FS

\$

5

ENQ

CR

NAK

GS

%

6

**ACK** 

SO

SYN

RS

&

BEL

SI

**ETB** 

US

16 Bit

21 Bit

Beispiele:

0

NUL

BS

DLE

CAN

SOH

HT

DC1

ΕM

STX

NL

DC2

**SUB** 

00

02

Zeichen UTF-16 UTF-16 binär UTF-8 binär U+0041 00000000 01000001 01000001 11000011 10000100 00000000 11000100 Ä U+00C4 11100010 10000010 10101100 € U+20AC 00100000 10101100

• 3-Byte-Sequenz: 1110xxxx 10xxxxxx 10xxxxxx

• 4-Byte-Sequenz: 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

"this" verweis auf das aktuelle initialisierte Objekt indem sich die Methode befindern in der this aufgerufen wird. "this" kann nur in Instanzmethoden benutzt werden, nicht in statischen Methoden. "this" ist immer der erste Übergabeparameter bei einer Instanzmethode

(unsichtbar).

darauf genau geregelt ist (Geheimnisprinzip). z.B. private String titel;

public String titel;

Vererbung: erlaubt die Definition ähnlicher Klassen, indem man Gemeinsamkeiten in Oberklassen zusammenfasst und Eigenheiten in davon abgeleiteten Unterklassen ergänzi public final class Unterklasse extends Oberklasse {...}

Polymorphie: ein und dieselbe Variable kann zur Laufzeit Objekte unterschiedlicher Klassen referenzieren. Stichwort Subtyping

z.B. Object x; x = new Beispiel(); x = 1;

Dynamische Bindung: erst zur Laufzeit wird entschieden, welche Methodenimplementierung aufgerufen wird

Laufzeit erkannt, welches intValue() aufgerufen wird

b) Was ist bei Klassen für Wertobjekte (value objects) zu beachten?

- pro Obiekt ein unveränderlicher Wert
- Wert -> Instanzvariablen, final
- gleicher Wert -> Wertobjekte sind gleich

c) Erklären Sie, was eine Schnittstelle ist und wie sie verwendet wird.

- Zur Definition einheitlich benutzbarer Klassen
- mbstrakte Oberklasse mit ausschließlich öffentlichen Methoden, ohne Instanzvariablen, Unne Konstruktor

# Referenztypen Felder: Typ[] Klassen Fundamentalklassen: Zeichenketten Wrapperklassen Boolean, ... Wurzelklassen Object, Throwable Typinformation Class<T>, ... Anwendungsklassen: enum Name class Name interface Name

c) Erklären Sie den Unterschied zwischen Referenztypen und Werttypen.

Werttyp-Variablen speichern den unmittelbaren Wert des angegebenen Typs. Der Wert wird bei einer Zuweisung kopiert.

Variablen mit Referenztyp speichern nur die Information, wo der Wert des angegebenen Typs steht. Der eigentliche Wert wird bei einer Zuweisung nicht kopiert, nur die Referenz (d.h. die Adresse der Werte).

```
0,3 * 2 = 0,6 Übertrag 0

> 0,6 * 2 = 0,2 Übertrag 1

0,2 * 2 = 0,4 Übertrag 0

0,4 * 2 = 0,8 Übertrag 0

0,8 * 2 = 0,6 Übertrag 1

Binärzahl:

0.01001

123 / 2 = 61 Rest 1

61 / 2 = 30 Rest 1

30 / 2 = 15 Rest 0

15 / 2 = 7 Rest 1

7 / 2 = 3 Rest 1

7 / 2 = 3 Rest 1

3 Amp = 1 Rest 1
```

### 7. Methoden

a) Erklären Sie an einer Beispielklasse, was Klassenmethoden, Instanzmeth
Instanzmethoden sind. Wo und wie werden sie jeweils definiert und wo ur public final class WertKlasse {
aufgerufen werden?

b) Erklären Sie das Überladen (Overloading) von Methoden.

Beim Overloading können Methoden einer Klasse denselben Namen habei wenn sie sich in der Parameterliste unterscheiden. Der Compiler wählt die

c) Erklären Sie das Überschreiben (Overriding) von Methoden.

Unterklassen können (vor allem abstrakte) Instanzmethoden ihrer Oberklasse(n) überschreiben, d.h. anders implementieren.

```
8. Zeichenketten
```

```
1 / 2 = 0 \text{ Rest } 1
                                                  Binärzahl:
                                                                1111011
   public GermanComparator() {}
   private String normalize(String s) {
           switch (s.charAt(<u>i</u>)) {
package klas envorlagen;
    private final TYP NAME;
    hashCode und toString überschreiben. (Entityclass höchstens toString)
```