Kontrollfragen A

1. Nennen Sie Unterschiede zwischen konkreten (Basis-) Klassen und abstrakten Klassen.
   Von abstrakten Klassen können keine Objekte instanziiert werden.

2. Nennen Sie Unterschiede zwischen abstrakten Klassen und Interfaces.
   Bei Interfaces sind alle darin enthaltenen Methoden abstrakt.
   Bei abstrakten Klassen können nur einzelne Methoden abstrakt sein und andere implementiert.

3. Können Sie in Interfaces Instanzvariablen definieren?
   Nein!
   Klassen-Konstanten sind möglich (public static final)

4. Können Sie in Interfaces private Methoden definieren?
   Nein, alle Methoden sind public.

5. In Interfaces können Sie keine Klassenmethoden (static) definieren. Warum nicht?
   Eine statische Methode muss zwingend implementiert sein.
   Bei einem Interface sind jedoch keine implementationen (Code-Blöcke) erlaubt, somit kann dies bei einem Interface nicht gemacht werden.

Kontrollfragen b

Was ergeben die folgenden Ausdrücke:

1. false && true ^ true ?
   false

   Begründung:
   false && (true ^ true)

2. false && true == false ^ false ?
   false

   Begründung:
   == hat eine höhere Priorität als ^ resp. &&
   false && (true == false) ^ false
   Richtig wäre: (false && true) == (false ^ false) (dann würde es true geben)

3. int a = 13, b = 114;
   System.out.println ("Ergebnis: " + a < b);
   ERROR: operator < cannot be applied to java-lang.String, int

   Begründung:
   + hat eine höhere Priorität als <
   System.out.println((“Ergebnis: “ +a) < b)
   String < Integer gibt einen Error!
   Richtig wäre: System.out.println ("Ergebnis: " + (a < b));

➔ **Prioritäten der Operanden siehe nächste Seite!**

Operator Precedence in Java / Rangfolge von Operatoren in Java

Level	Operator	Description	Associativity
1	[] . () ++ --	access array element access object member invoke a method / parentheses post-increment post-decrement	left to right
2	++ -- + - ! ~	pre-increment pre-decrement unary plus unary minus logical NOT / logical negation bitwise NOT / bitwise complement	*right to left*
3	(type) new	type cast Object creation	*right to left*
4	* / %	Multiplication Division Modulus	left to right
5	+ - +	Addition Subtraction String concatenation	left to right
6	<< >> >>>	Bitwise left shift Bitwise right shift with sign extension Bitwise right shift with zero extension	left to right
7	< <= > >= instanceof	less than less than or equal greater than greater than or equal Type comparison (objects only)	left to right
8	== !=	is equal to is not equal to	left to right
9	&	Bitwise AND	left to right
10	^	Bitwise exclusive OR	left to right
11	|	Bitwise OR	left to right
12	&&	Logical AND	left to right
13	||	Logical OR	left to right
14	?:	conditional	*right to left*
15	= += -= *= /= %=	Assignment Addition assignment Subtraction assignment Multiplication assignment Division assignment Modulus assignment	*right to left*