

- ① Gegeben sind die Variablen *char a* und *char b* mit Inhalten. Was passiert beim Befehl *a=b*; ?
 Die Variable *a* erhält den Inhalt 'b'. Der Buchstabe 'a' wird zum Buchstaben 'b'. Die Variable *b* erhält den Inhalt von *a*. Sind *a* und *b* gleich, so ist das Ergebnis *true*. Die Variable *a* erhält den Inhalt der Variablen *b* und *b* behält seinen alten Inhalt. Variable *b* wird zu *a*.
- ② Der Befehl *iks = üpsilon*; soll gültig sein. Dann müssen: beide Variablen den gleichen Typ haben. beide Variablen vorher den gleichen Inhalt haben. beide Variablen den gleichen Namen haben. beide Variablen nachher den gleichen Inhalt haben -- nämlich den, den vorher schon *iks* *üpsilon* keiner von beiden hatte. *üpsilon* musste schon Inhalt haben.
- ③ *a, b* sind definiert und bereits gefüllt. Betrachte die Anweisungen (1) *a = b + 4*; und (2) *a - 4 = b*;
 (1) geht nur, wenn *a* und *b* z.B. beide vom Typ *int* oder vom Typ *double* sind; (1) und (2) sind äquivalent und beide gültig. (2) wird *true*, wenn z.B. *a=9* und *b=5* ist. nur (2) ist richtig: *b* erhält den um 4 verminderten Wert von *a*. nur (1) ist gültig: *a* erhält den um 4 erhöhten Wert von *b*. Keine Anweisung ist gültig: es müsste statt dessen z.B. *a == b + 4*; heißen!
- ④ Gegeben sind *int z = 4*; *String zett = "4"*; und *String wort="vier"*; Dann sind folgende Zuweisungen möglich/richtig: *wort = zett*; *wort = "z"*; *wort = "4"*; *zett = "4" + '4'*; *z = zett*; *z = wort*; *zett = 'wort'*; *zett = "wort"*; *z = 'wort'*; *z = "wort"*; *zett = z*;
- ⑤ Weiterhin sei *String wort="vier"*; definiert. Nach Ausführung des Befehls *wort = wort + "wort"*; hat dann die Variable *wort* folgenden Wert (Inhalt): "4 + wort" 4 + "4" "vier+wort" "wort+vier" "4wort" "vierwort" "4+4" "44" "8" "wortwortwortwort"

Für die Aufgaben 6 bis 8 ist die Klasse *Test* wie folgt gegeben:

```
public class Test
{
    String farbe = "grau";
    int wert = 200;

    public void value (int w)
    { wert = w; }

    public void black()
    { farbe = "schwarz"; }

    public String white ()
    { return (farbe); }
}
```

- ⑥ Folgende Anweisungen (in einer anderen Klasse) sind richtig: *Test t = new t*; *Test t = new t()*; *Test t = new Test*; *Test t = new() Test*; *Test t = new Test()*; *Test() t = new Test()*; *new t = Test()*; *Test t = new Test(grau, 200)*; *Test t = new Test("grau",200)*;
- ⑦ Nach erfolgreichem Erzeugen von *t* als Objekt vom Typ *Test* in einer anderen Klasse ist richtig:
 t[farbe] = "rot"; *farbe.t = "rot"*; *t("rot") = farbe*; *farbe("rot") = t*; *t = farbe("rot")*; *t.farbe = new("rot")*; *t."rot" = new farbe*; *t."rot" = new farbe()*;
 t.value("rot"); *t.value()*; *int x = t.value()*; *int x = t.value(4)*; *int color = t.white()*;
 String colour = t.white(); *String colour = t.black()*; *t.black()*; *t.black("red")*;
 t.schwarz = black(); *t.farbe() = t.white()*; *t.farbe = t.white()*; *t.white() = "rot"*;
- ⑧ Jetzt sei zusätzlich in einer anderen Klasse *Test[] neu = new Test[5]*; definiert. Syntaktisch korrekt ist: *neu[2].farbe*; *neu["grau"] = farbe*; *neu["grau"].farbe*; *neu[farbe] = "grau"*; *neu[2]["grau"] = farbe*; *neu[farbe][2] = "grau"*; *farbe.neu[2]*; *2.farbe[neu]*;
- ⑨ Jetzt sei *int[] reihe = {7, 8, 10, 6, 5}*; definiert. Dann gilt: *reihe[7]* ist der Wert ganz links. in *reihe[2]* steht 10 an der Stelle 1 steht eine 8 an der Stelle 5 steht eine 5 *reihe[5]* gibt's nicht an der Stelle 10 steht eine 2 *reihe[4]* ist der Wert ganz rechts