

Fragen, Antworten, Kommentare zur aktuellen Vorlesung

Sie haben zwei E-Mails mit dem Wunsch bekommen, die Lehrevaluationen von Vorlesung und Praktikum durchzuführen. Bitte kommen Sie dem zügig nach.

(Falls Sie keine Mail erhalten haben, direkt bei mir melden)

Kommentar zur Aufgabe mit dem equals für die Linie:

Leider hat nicht jeder genau verstanden, dass man die Methode nicht alleine frei neu schreiben sollte. In der Vorlesung wurde ein klarer (Standard-)Weg vorgegeben, wie ein equals() entstehen soll. Zuerst bei jeder einzelnen Objektvariable, die als Typ eine echte Klasse hat, schrittweise einen Vergleich machen und nach Unterschieden suchen. Am Ende werden dann die Objektvariablen, die einen elementaren Typen (int, boolean, double, ...) direkt mit == auf Gleichheit geprüft [gibt es bei Linie nicht]. Natürlich dürfen erfahrenere Leute mit der Lösung experimentieren und sie umgestalten, da es durchaus sinnvolle Lösungsvarianten gibt. Dies gilt für Personen, die mit der Programmierung anfangen nur, wenn Sie die Standard-Lösung umgesetzt und genauso wichtig, das Konzept verstanden haben. Zum Verständnis des Konzepts tragen Aktivitätsdiagramme bei.

Frage: Ist der Nutzungsdialog so ok?

```
void dialog() {
    this.io.ausgeben("(0) Ende (1) Kreis (2) Dreieck: ");
    int eingabe = this.io leseInteger();
    if (eingabe == 1){
        this.kreisEingeben();
    }
    if (eingabe == 2){
        this.dreieckEingeben();
    }
    if (eingabe != 0){
        dialog();
    }
}
```

Nein. Am Ende ruft die Methode sich selbst auf, was im Beispiel kein Problem ist, generell aber dazu führen kann, dass innerhalb einer Methode die gleiche Methode beliebig oft aufgerufen wird. Erinnern Sie sich an den Debugger, da werden die Aufrufe nicht abgeschlossener Methoden auf einen Stapel gelegt, der nur endlich viel Speicher hat und es so irgendwann zu einem Speicherüberlauf führen kann. Der generelle Ansatz, dass eine Methode sich selbst direkt oder über andere Methoden aufruft wird Rekursion genannt und kann für bestimmte Aufgaben, aber nicht für einen Dialog, sinnvoll sein. Die sinnvolle Nutzung von Rekursion ist später ein Thema im Studium und spielt in Programmierung 1 keine Rolle.

Frage: Bei Aufgabe 25e, was ist genau mit Schnitt gemeint?

Es soll der Durchschnitt *aller* bisher eingegebenen Zahlen berechnet werden. Dazu wird keine Liste benötigt, in der alle Zahlen drinstehen. Überlegen Sie welche Informationen Sie brauchen, um einen Durchschnitt zu berechnen. (Mit einer ArrayList geht es natürlich auch, sollte aber hier ohne gehen.)

Frage: Ich habe die Aufgaben 6 und 13 zu den Ausdrücken mal wiederholt. Mir ist noch unklar ob Zuweisungen und Methodenaufrufe Ausdrücke sind.

Zuweisungen sind als Befehle kein Ausdruck, da sie keinen Wert zurück liefern.

Googelt man sich da rein, ist eine reine Zuweisung schon ein Ausdruck. Das haben wir in der Vorlesung nicht behandelt und spielt in der Klausur keine Rolle:

Zuweisung als Befehl: `c=a+1;`

reine Zuweisung: `c=a+1`

Also ist folgender Befehl syntaktisch korrekt:

`a=b=c+1;`

(der Wert vom Ausdruck `c+1` wird der Variablen `b` zugewiesen, dann wird der Wert von `b` zugewiesen) Wird seltenst benutzt, da es immer irritiert, findet man manchmal in einer Variante bei Schleifen:

```
int x;
while ((x = o.fkt(y)) != 4){
    // hier kann aktueller Wert von x genutzt werden
    erst fkt(y) ausrechnen, den Wert x zuweisen, dann mit 4 vergleichen.
```

Ein Methodenaufruf ist ein Ausdruck, wenn das Ergebnis nicht void ist, also ist `d.mathA(c)` in der Beispielaufgabe ein Ausdruck.

Nebenbei könnte bei A 13 jetzt aufgefallen sein, dass `methA(c)`; durchaus ein sinnvoller Befehl sein könnte, wenn `methA(.)` eine Klassenmethode wäre, was hier nicht der Fall ist.

Beim Blick auf die Probeklausur fällt auf, dass genau eine solche Aufgabe nicht drin vorkommt, das genau wegen der hier genannten Spezialfälle.