Ludwig-Maximilians-Universität München
Institut für Informatik
PD. Dr. Peer Kröger, Dr. Andreas Züfle
Daniel Kaltenthaler, Klaus Schmid

München, 25.11.2014

Einführung in die Programmierung

WS 2014/15

Übungsblatt 7: Rekursion und Variablen

Besprechung: 03./05./08.12.2014

Ende der Abgabefrist: Dienstag, 02.12.2014 14:00 Uhr.

Hinweise zur Abgabe:
Geben Sie bitte Ihre gesammelten Lösungen zu diesem Übungsblatt in einer Datei `loesung07.zip` unter `https://uniworx.ifi.lmu.de` ab.

Aufgabe 7-1 *Typkonversion* **0 Punkte**

Ausdrücke in Java, die aus mehr als nur einem elementaren Datentyp zusammengesetzt sind, werden je nach Verwendung durch implizite oder explizite Typkonversion ausgewertet. Gegeben sind nun folgende Variablendeklarationen:

```
long l1 = 5L;
long l2 = 123L;
int i1 = 9;
int i2 = 5877;
short s1 = 6;
byte b1 = 99;
byte b2 = 2;
float f1 = 2.0f;
double d1 = 0.222;
double d2 = 17.0;
char c1 = '!';
char c2 = 'a';
String str1 = "123";
```

(a) Welchen Typ und welchen Wert haben die folgenden Ausdrücke? Begründen Sie Ihre Entscheidung kurz.

- (b1 * i1) / (f1 * 3.0f)
- ”1 + 2 + 3 = ”+ (i1 - 3)
- d1 / f1 + i1
- c1 * c2
- i1 + str1 + l2

(b) Begründen Sie, warum nachfolgender Ausdruck einen Kompilierfehler ergibt. Wie könnte man ihn beheben?

```
byte b3 = b1 + b2;
```

Ergänzen Sie die Textdatei `typkonversion.txt`, die Sie auf der Homepage finden, mit Ihren Lösungen und geben Sie diese ab.

Hinweise:

- *Zeichen* vom Typ **`char`** werden in zusammengesetzten arithmetischen Ausdrücken automatisch in den Typ **`int`** konvertiert (ensprechend ihrem ASCII-Code).
- In Java werden *Zeichenketten* mit doppelten Anführungszeichen umrahmt und sind vom Typ `String`. Die Konkatenation von Zeichenketten erfolgt durch den +-Operator (der Operator ist also *überladen*). Ist wenigstens einer der beiden Operanden in `a + b` ein String, so wird der gesamte Ausdruck als String-Konkatenation ausgeführt.
- Zur Überprüfung Ihrer Lösung ist es hilfreich, ein Java-Programm zu schreiben, das die angegebenen Ausdrücke ausgibt (Programm nicht abgeben!).

Aufgabe 7-2 *Rekursion in Java* **1+1+2+2+2 Punkte**

Implementieren Sie die folgenden Berechnungen **auf rekursive Weise** in der Datei `Rekursion.java` und erstellen Sie für jede Unteraufgabe eine Methode. Eine iterative Berechnung oder die Verwendung von Bibliotheksfunktionen wie beispielsweise `Math.pow()` oder `Math.sqrt()` wird nicht akzeptiert. Eigene Hilfsmethoden dürfen verwendet werden.

(a) Berechnen Sie für ein $x \in \mathbb{R}$ und ein $n \in \mathbb{N}$ das Ergebnis der Exponentialfunktion x^n. Ein Beispiel:

```
exp (4.2, 3) = 74.088
```

(b) Berechnen Sie für ein $n \in \mathbb{N}$ das Ergebnis der Fakultätsfunktion $n! = 1 \times 2 \times 3 \times ... \times n$. Ein Beispiel:

```
fakultaet (4) = 24
```

(c) Überprüfen Sie, ob es sich bei einem $n \in \mathbb{N}$ um eine Primzahl handelt, indem Sie alle potenziellen Teiler von 2 bis $n - 1$ testen. Ein Beispiel:

```
istPrim (893) = false
```

(d) Setzen Sie den euklidischen Algorithmus zur Berechnung des größten gemeinsamen Teilers zweier Zahlen $m, n \in \mathbb{N}$ rekursiv um. Die rekursive Version des euklidischen Algorithmus ist wie folgt definiert:

```
    ggt(m,n) := m, falls m == n
ggt(m,n) := ggt(m-n,n), falls m > n
ggt(m,n) := ggt(m,n-m), falls n > m
```

Ein Beispiel-Aufruf:

```
ggt (42,154) = 14
```

(e) Überführen Sie ein $n \in \mathbb{N}$ in p-adische Darstellung. Sie können einen Gültigkeitsbereich für p von $[2, 16]$ annehmen. Die Methode soll das Ergebnis in einem `String` zurück geben. Beispiele:

```
rebase (5,2) = "101"
rebase (237,16) = "ED"
```

Aufgabe 7-3 *Globale und lokale Größen* **1+1+1 Punkte**

Implementieren Sie folgende Methoden im zur Verfügung gestellten Modul `GlobalLokal.java`.

(a) Implementieren Sie die Methode `swap()`, die die Werte von `g1` und `g2` gegeneinander austauscht.

(b) Erstellen Sie die Methode `sort(a,b,c)`, die die Parameter der Größe nach aufsteigend sortiert und in dieser Reihenfolge in die Variablen `smallest`, `median`, `biggest` abspeichert.

(c) Ausgehend von folgender Notentabelle, erstellen Sie die Methode `grade(n)`, die für eine ganzzahlige Anzahl Punkte die resultierende Note als `double`-Wert zurück gibt.

Punkte	Note
0 – 59,5	5,0
60 – 64,5	4,0
65 – 69,5	3,7
70 – 74,5	3,3
75 – 79,5	3,0
80 – 84,5	2,7
85 – 89,5	2,3
90 – 94,5	2,0
95 – 99,5	1,7
100 – 104,5	1,3
105 – 120	1,0