Institut für Informatik
der Universität München
Prof. Dr. M. Hofmann
Dr. J. Johannsen, Dr. R. Matthes

SS 2004

13.5.04

Übungen zur Vorlesung
Informatik II

Blatt 4

Abgabe der Hausaufgaben bis spätestens am 24.5.04, 14:00 Uhr über
`http://lehre.tcs.ifi.lmu.de/info2/Abgabe/abgabe.php`,
Bearbeitung in Gruppen zu max. 3 Personen ist zulässig.

Bitte halten Sie sich an folgende Grundsätze zur Bearbeitung der Aufgaben. Die Bewertung Ihrer Abgaben kann andernfalls deutlich herabgesetzt werden.

Verwenden Sie genau die Signatur der Aufgabenstellung, insbesondere die Namen der Klassen und Methoden in der vorgegebenen Klein- und Großschreibung.

Verwenden Sie keine „magischen" Größen in Ihren Programmen, d. h., weisen Sie Konstanten (außer den Zahlen 0 und 1 und dem leeren String) immer einer mit `final` deklarierten und in Großbuchstaben geschriebenen „Variablen" zu und verwenden Sie anschließend nur diese. Wer gute Gründe hat für eine Abweichung von dieser Regel, der kommentiere seinen Code entsprechend.

Zu jeder Klasse xxx, für die keine `main`-Methode verlangt ist, muß eine Testklasse xxx`Test` geschrieben werden, die sinnvolle Tests der Klasse xxx enthält (in einer `main`-Methode). In den Tests dürfen „magische" Größen natürlich beliebig vorkommen!

Jede Klasse und `public`-Methode muß angemessen mit `javadoc`-Kommentaren dokumentiert werden.

Schriftliche Aufgaben müssen als ASCII-Datei (Umlaute etc. sind aber erlaubt) mit dem angegebenen Namen abgegeben werden.

Programmieraufgabe P-10 (`WurzelRechner.java`): **5 Punkte**

Die *Methode von Heron* ist ein schon aus dem antiken Griechenland bekanntes Näherungsverfahren für Quadratwurzeln. Ist x ein Näherungswert für $\sqrt{a}$, so ist das arithmetische Mittel aus x und a/x ein besserer Näherungswert.

Programmieren Sie eine Klasse `WurzelRechner`, die mit der ersten Näherung 1 beginnt, und bei Aufruf der Methode `double naechsteNaeherung()` den nächsten Näherungswert liefert. Die Methode `boolean nochUngenau()` soll `false` zurückgeben, falls zwei aufeinanderfolgende Näherungen hinreichend nah beieinander liegen.

Die Testklasse soll zu einem Wert a, der über die Konsole eingegeben wird, die Folge der Näherungswerte für $\sqrt{a}$ ausgeben, die solange fortgesetzt werden soll, bis `nochUngenau()` irgendwann `false` wird.

– bitte wenden –

Programmieraufgabe P-11 (`Tabelle.java, Tabelle.html`): **5 Punkte**
Programmieren Sie ein Applet, das eine Multiplikationstabelle für die Zahlen von 1 bis 9 erzeugt. Die Tabelle soll einen Rahmen haben, und die Eingabewerte an den Rändern sollen durch Linien von der eigentlichen Tabelle getrennt sein.

Wegen der Kommutativität der Multiplikation soll nur die rechte obere Hälfte der Tabelle bis einschließlich der Hauptdiagonale gefüllt werden.

Ein Beispiel (als Screenshot) finden Sie auf der Vorlesungshomepage.

Schriftliche Aufgabe S-12 (`Fakultaet.txt`): **10 Punkte**
Betrachten Sie folgendes Programm-Fragment zur Berechnung der Fakultätsfunktion:

```
public static int fakultaet(int n) {
    // Anfang
    int a = 1;
    int k = n;
    while (k > 0) {
        a = a * k;
        k--;
    }
    // Ende
    return a;
}
```

Sei c der Programmabschnitt zwischen den Kommentaren `Anfang` und `Ende`. Leiten Sie im Hoare-Kalkül das Hoare-Tripel $\{n \geq 0\}\, c\, \{a = n!\}$ – mithin die Korrektheitsaussage für die Methode `fakultaet` – ab. Geben Sie dazu immer die benutzte Regel an sowie die Ableitungen für die Prämissen. Sie können "top-down" oder "bottom-up" vorgehen, also von dem gewünschten Tripel her oder von den ersten Prämissen her (die Sie vermutlich in einem ersten Durchgang auf Papier herausfinden).

Hinweis: Die Schleifen-Invariante enthält unter Anderem die Aussage $a \cdot k! = n!$. Beachten Sie aber den Definitionsbereich von "!".