64-041 Übung Rechnerstrukturen

Aufgabenblatt 5 Ausgabe: 21.11., Abgabe: 28.11. 24:00

Gruppe	
Name(n)	Matrikelnummer(n)

Aufgabe 5.1 (Punkte 10)

Größenvergleich von Gleitkommazahlen: Für den Vergleich von Gleitkommazahlen bietet Java alle sechs Vergleichsoperatoren:

```
a == b    a != b    a > b    a >= b    a < b    a <= b
```

Aufgrund der unvermeidlichen Rundungsfehler bei Gleitkommarechnung ist jedoch Vorsicht bei Verwendung dieser Operatoren geboten. Zum Beispiel liefert

```
double a = 0.1;
double b = 0.3;
System.out.println( (3*a) == b );
```

den Wert `false`.

Ein naheliegender Ansatz ist daher, zwei Zahlen als „gleich“ anzusehen, wenn der Absolutwert ihrer Differenz kleiner als eine (vom Benutzer) vorgegebene Konstante ist:

```
final double eps = 1.0E-12;

if (Math.abs( a - b ) <= eps) { // Zahlen fast gleich
  ...
}
```

Welchen offensichtlichen Nachteil hat dieses Verfahren? Skizzieren Sie außerdem, wie man ihn beheben könnte.

Aufgabe 5.2 (Punkte 8+2+5)

UTF-8: Die ISO-8859-1 Codierung benutzt 8-bit für jedes enthaltene Zeichen. Die direkte Codierung der basic-multilingual Plane von Unicode (Java Datentyp `char`) verwendet pro Zeichen 16-bit, während die UTF-8 Codierung Vielfache von 8-bit benutzt.

(a) Wir betrachten einen deutschsprachigen Text mit insgesamt 500 000 Zeichen. Wir nehmen die folgenden Wahrscheinlichkeiten für die Umlaute an, andere Sonderzeichen kommen nicht vor: **Ä/ä** 0,56 %, **Ö/ö** 0,287 %, **Ü/ü** 0,616 % und **ß** 0,308 %.

Wie viele Bytes belegt dieser Text bei Codierung nach ISO-8859-1, in direkter Unicode Darstellung und in UTF-8?

(b) Im aktuellen Unicode-Standard sind für die CJK-Symbole (chinesisch, japanisch, koreanisch) neben einem Basis-Zeichensatz noch mehrere Erweiterungen definiert. Wir betrachten einen chinesischen Text mit insgesamt 500 000 Schriftzeichen, der aus den Basis-Zeichen (von `U+4E00` bis `U+9FEA`) und Symbolen der Extension-A (von `U+3400` bis `U+4DB5`) besteht.

Wie viele Symbole sind das? Berechnen Sie das Ergebnis direkt im Hexadezimalsystem aus und verwenden Sie dabei das 16-Komplement für die Subtraktion. Die Lösung muss mit Zwischenschritten abgeben werden.

(c) Wie viele Bytes belegt der chinesische Text bei direkter Unicode Darstellung und bei Codierung als UTF-8?

Aufgabe 5.3 (Punkte 5+5+5+5)

Shift-Operationen statt Multiplikation: Ersetzen Sie die folgenden Berechnungen *möglichst effizient* durch eine Folge von Operationen: <<, +, -. Nehmen Sie für die Variablen x und y den Datentyp `int` (32-bit Zweierkomplementzahl) an.

(a) $y = 18 \cdot x$

(b) $y = 14 \cdot x$

(c) $y = -56 \cdot x$

(d) $y = 62 \cdot (x + 2)$

Aufgabe 5.4 (Punkte 5+10+10+15)

Logische- und Shift-Operationen: Realisieren Sie, die folgenden Funktionen als *straightline*-Code in Java, das heißt ohne Schleifen oder If-Else Abfragen, bzw. ternärer Operator `.. ? .. : ...` Außerdem dürfen nur einige der logischen und arithmetischen Operatoren benutzt werden:

```
! ~ & ^ | + << >> >>>
```

Alle Eingabeparameter und Rückgabewerte sind jeweils (32-bit) Integerwerte.

(a) `bitXnor(x,y)` Diese Funktion soll die XNOR-Verknüpfung (Äquivalenz) realisieren: $x_i \equiv y_i$. Als Operatoren dürfen nur `|` und `~` (OR, Negation) benutzt werden.

(b) `getByte(x,n)` Diese Funktion soll das, durch `n` angegebene Byte ($0 \leq \texttt{n} \leq 3$) aus dem Wert `x` extrahieren.

(c) `rotateRight(x,n)` Die Funktion soll den in Java nicht vorhandenen Rotate-Right Operator für `x` nachbilden. Für das zweite Argument `n` gilt: $0 \leq \texttt{n} \leq 31$.

(d) `abs(x)` Der Absolutwert (Betrag) von `x`. Welchen Wert liefert ihre Funktion für den Eingabewert -2^{31}? Beschreiben Sie, wie Ihre Lösung funktioniert.

Aufgabe 5.5 (Punkte 10+5)

Base-64 Codierung: Wie in der Vorlesung skizziert, werden bei der Base-64 Codierung jeweils drei 8-bit Eingangswerte durch vier 6-bit Ausgangswerte ersetzt, die dann zur Datenübertragung als (7-bit) ASCII-Zeichen codiert werden.

(a) Beschreiben Sie durch logische- und Schiebe-Operationen, wie bei der Decodierung aus den vier Eingabezeichen `a1`...`a4` (hier schon als Integer Zahlen), die drei 8-bit Ausgangswerte `b1`...`b3` berechnet werden. Vervollständigen Sie dazu die Ausdrücke `b`... im nachfolgenden Java-Code.

```
int a1, a2, a3, a4;          // vier Zeichen, Wertebereich je 0..63

int b1 = ?
int b2 = ?
int b3 = ?

...
```

(b) In dem ersten Aufgabenteil wurde angenommen, dass die Zeichen `a1`...`a4` schon in Zahlen von 0...63 umgesetzt worden sind. Skizzieren Sie, wie diese Konvertierung durchgeführt werden kann. Programmcode muss nicht geschrieben werden, es genügt wenn Sie textuell beschreiben, wie man das machen könnte.