Do., 13.12.2012 Informatik Q1 M (Kr)

2. Klausur 12/I (Q1.1)

Dauer: 4*45 min (9:15 bis 12:15 Uhr) Name: ___www.r-krell.de__

Hilfsmittel: Taschenrechner, Geodreieck, Blatt mit 1-AMOR-Befehlssatz (ohne eigene Zusätze)

* *Achte auf sorgfältige Darstellung mit vollständigem, nachvollziehbarem Lösungsweg!* *
* *Kommentiere deine Programme!* *

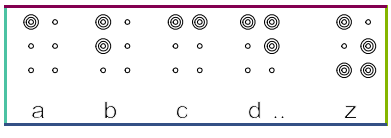
1 Dualarithmetik I

a) Addiere die 12-Bit-Zahlen dual (ohne Umwandlung): 0101 1001 0011 + 1001 0101 1111
b) Bilde das Zweierkomplement (d.h. wechsele das Vorzeichen) von 1001 0110 1110
c) Verwandle in 8-Bit-Dualzahlen, berechne dort nachvollziehbar von Hand und verwandle das Ergebnis zurück ins Dezimalsystem: c1) 39 + 27 c2) 27 – 39
d) Stelle die Ziffernfolge im BCD-System dar und addiere dual: 39 + 27 .
Erläutere außerdem, was man unter der Dezimalanpassung versteht und warum und in welchen beiden Fällen sie grundsätzlich durchgeführt werden muss.
e) Verwandle näherungsweise in duale Kommazahlen mit maximal5 Nachkommastellen (und den Stellenwerten $^1/_2$, $^1/_4$, ..., $^1/_{32}$), addiere dual und verwandle zurück: 3,9 + 2,7
f) Erläutere, dass die Rechnung 3,9 + 2,7 im BCD-System genau wie in Aufgabe d) abläuft. Vergleiche jeweils den Aufwand dual bzw. BCD für ganze Zahlen (c1) und d)) sowie für Kommazahlen (e) und f)) und erläutere, wann und wo das BCD-System Vorteile bietet.
g) Der Java-Datentyp *double* verwendet das Dualsystem, nicht das BCD-System. Seien a und b zwei *double*-Variablen, jeweils gefüllt mit dem Ergebnis einer längeren Berechnung. Jetzt soll kontrolliert werden, ob bei beiden Rechnungen auf mindestens 3 dezimale Nachkommastellen genau das gleiche heraus gekommen ist. Erläutere für alle Abfragen, ob sie bei jeder solchen näherungsweisen Übereinstimmung und nur in diesem Fall „gleich“ melden, und wähle zum Schluss begründet die beste Kontrolle aus (oder gib eine eigene beste Möglichkeit an):
(1) if (a == b) { System.out.println ("gleich"); }
(2) if (a + 0.0001 == b) { System.out.println ("gleich"); }
(3) if (a + 0.0001 >= b) { System.out.println ("gleich"); }
(4) if (a – b == 0) { System.out.println ("gleich"); }
(5) if (a – b < 0.0001) { System.out.println ("gleich"); }
(6) if (a – b < 0.0001 || b – a < 0.0001) { System.out.println ("gleich"); }
(7) if (a – b < 0.0001 && b – a < 0.0001) { System.out.println ("gleich"); }

2 Dualarithmetik II

a) Verwandle in Dualzahlen, berechne dort nachvollziehbar von Hand und verwandle das Ergebnis zurück ins Dezimalsystem: a1) 9 * 5 a2) 13 * 6
b) Fertige für das Verfahren der dualen Multiplikation ein grobes Struktogramm mit deutschem Text an (Beschriftung etwa „Gehe im 1. Faktor alle Dualstellen von hinten nach vorne durch“ [oder auch „im 2. Faktor“ oder „..von vorne nach hinten“ -- was eben gebraucht wird]) und schreibe dann eine kommentierte Java-Methode *public String multipliziere (String dual1, String dual2)*. Darin soll zum Summieren die schon fertige Methode *private String addiere (String dual1, String dual2)* benutzt werden, die je zwei rechtsbündig übereinander stehende Dualzahlen addiert.
c) Verwandle in Dualzahlen, berechne dort nachvollziehbar von Hand und verwandle das Ergebnis zurück ins Dezimalsystem: 30 : 6
d) Gib 2 Vor- und 2 Nachteile der Dualarithmetik gegenüber dem Dezimalrechnen an!

3 Kodierungen



a) Bei der Blindenschrift bestehen alle Zeichen aus bis zu 6 erhabenen Punkten in einer 2x3-Matrix.
a1) Können in der Blindenschrift außer den 26 Buchstaben *a* bis *z* auch die Umlaute *äöüß*, die Ziffern *0* bis *9* und die üblichen Satzzeichen *.,;:?!'"*-+*=

dargestellt werden?

a2) In der Blindenschrift gibt es zwar einige Kurzschriftzeichen (z.B. ein Extra-Zeichen für *sch*), aber keine Großbuchstaben. Statt dessen zeigt ein vorangestelltes Großschriftzeichen an, dass der nächste Buchstabe eigentlich groß geschrieben werden müsste. Warum?

a3) Entscheide begründet, ob die Blindenschrift als 6-Bit-Binärcode aufgefasst werden kann.

b) In Java belegt der String "Hi" drei Byte. Notiere alle Bits (lt. Ansi-Tabelle)!

c) Warum werden Dualzahlen im Computer mit führenden Nullen geschrieben, d.h. 00000010 statt einfach nur 10 für 2?

d) Gib dezimal den Ganzzahlbereich an, der mit (1) 7 oder (2) n Bit dargestellt werden kann (Beisp.: 3 Bit für -4..+3)

```
Ansi-Tabelle (Auszug)   1 1 1 1 1 1
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

64 @ A B C D E F G H I J K L M N O
80 P Q R S T U V W X Y Z [ \ ] ^ _
96 ' a b c d e f g h i j k l m n o

   (Beispiel: ^ = 80 + 14 = 94)
```

e) Warum nimmt man in Kauf, dass eine negative Zahl mehr dargestellt werden kann als eine positive, d.h. bei 3-Bit-Zahlen -4..+3 statt z.B. -3..+4 ?

f) Beim 1-AMOR-Modellrechner besteht ein Maschinenbefehl aus 10 Bit, nämlich 4 Bit für den Op-Code (Befehlsnummer) und 6 Bit für die Adresse. Ein anderer Rechner verfügt über 50 verschiedene Befehle und hat 4096 Speicherzellen (Adressen 0..4095). Gib das Befehlsformat an und notiere, wie viele Speicherzellen von je 8 Bit ein Maschinenbefehl belegt.

4 Maschinensprache

a) Zeichne für nebenstehendes Maschinenprogramm einen PAP (Programmablaufplan) und entscheide, was auf dem Bildschirm erscheint, wenn (1) x = 3 bzw. wenn (2) x = 7 eingegeben wurde. Beschreibe prägnant (in einem Satz), was das Programm für/mit x macht!

b) Nachdem z.B. der Befehl „1.) SUB 9“ ausgeführt wurde, holt der Modellrechner den nächsten Befehl „2.) SP< 5“ und erhöht, noch bevor dieser Befehl ausgeführt wird, den Befehlszähler um 1 (auf 3). Erläutere, warum dies am Ende der Holphase (statt erst nach der Ausführung von „2.) SP< 5“) geschieht, zumal es ja möglicherweise überflüssig ist.

```
 0.) LAD 63
 1.) SUB 9
 2.) SP< 5
 3.) LAD 63
 4.) SPR 6
 5.) LAD 9
 6.) TRF 62
 7.) STOP

 9.) 5
62.)  Bildschirm
63.) x von Tastatur
```

5 Java und Maschinensprache

Gegeben sind zwei ganze Zahlen x und y. Als Hilfsvariable wird z genutzt, das mit z=0 startet. Solange x > 0 ist, wird x immer um 1 vermindert und wird z um y erhöht. Am Ende wird der Wert von z ausgegeben. (Beispiel: (x,y,z) = (3,4,0) -> (2,4,4) -> (1,4,8) -> (0,4,12) -- Endergebnis 12)

a) Zeichne ein Struktogramm

b) Schreibe eine Java-Methode *public void aufgabe5 (int x, int y)*, die am Schluss den Wert von z auf den Bildschirm schreibt

c) Zeichne einen Programmablaufplan (PAP), der sich an 1-AMOR orientiert (Schleifenende bei x==0)

d) Schreibe ein 1-AMOR-Maschinenprogramm, das die Rechnung durchführt (Werte von x und y sollen anfangs in den Zellen 20 und 21 stehen; für z soll der Bildschirmspeicher [Zelle 62]) benutzt werden.

e) Begründe, dass das beschriebene Verfahren immer das Produkt z = x*y der anfänglichen Werte von x und y liefert. Fasst man die Verringerung von x um 1 als Addition x = x + (-1) auf, dann wird bei diesem Verfahren nur addiert – bei der schriftlichen Multiplikation (vgl. 2a)/b)) musste ebenfalls addiert, aber auch verschoben werden. Vergleiche den Aufwand beider Multiplikationen und begründe, welches Verfahren insbesondere bei großem x besser ist.