64-041 Übung Rechnerstrukturen und Betriebssysteme

Aufgabenblatt 5 Ausgabe: 13.11., Abgabe: 20.11. 24:00

Gruppe	
Name(n)	Matrikelnummer(n)

Aufgabe 5.1 (Punkte 10+10+5)

ULP: Um zu untersuchen, wie sich Rundungsfehler bei der Gleitkomma-Arithmetik auswirken, betrachten wir folgendes Programm, das in einer Schleife immer wieder den Wert `0,1`, der sich in binärer Gleitkommadarstellung ja nicht exakt darstellen lässt, aufaddiert.

```
1  // Test floating point arithmetic...
2  public class aufg05_1 {
3    public static void main( String args[] ) {
4      // calculate the sum of 1E9 unprecise numbers
5      int n = 0;
6      float sum;
7      float limit = 1.0E8f;
8      for( sum = 0; sum < limit; sum += 0.1 ) {
9        n++;
10 //     if ((n % 100000) == 0) {
11 //       System.out.println("n="+ n + " sum="+ sum + " target="+ (n*0.1));
12 //     }
13     }
14     System.out.println("sum is " + sum + " compared to " + (n*0.1));
15   }
16 }
```

(a) Was erwarten Sie (ungefähr) als Ausgabewert des Programms?

(b) Was passiert tatsächlich? Warum?
Tipp: Es kann helfen, die auskommentierten Codezeilen 10 . . . 12 wieder zu aktivieren.

(c) Schreiben Sie das Programm so um, dass es den ursprünglich angedachten Zweck erfüllt.

Aufgabe 5.2 (Punkte 15+5)

ISO-8859-1: Entschlüsseln Sie mit Hilfe der Tabellen aus den Vorlesungsfolien (Zeichensatz nach ISO-8859-1) die folgende Zeichenkette. Schreiben Sie dabei auch alle Steuerzeichen mit auf. Die einzelnen Zeichen sind als Hexwerte angegeben.

(a)
```
44 69 65 73 20 69 73 74 20 64 69 65 0D 0A DC 62
75 6E 67 73 61 75 66 67 61 62 65 20 35 2E 32 21
0D 0A 09 35 37 20 5A 65 69 63 68 65 6E 0D 0A 09
20 34 20 5A 65 69 6C 65 6E
```

(b) Was verrät Ihnen der Text über den Rechner mit dem erstellt worden ist?

Aufgabe 5.3 (Punkte 10+10+10+10)

Radix-50 Codierung: In den Urtagen der Informatik, als Speicher noch sehr teuer war, wandte die Firma DEC (Digital Equipment Corporation) folgendes Verfahren an, um Zeichenketten komprimiert im Speicher ablegen zu können:

1. Zunächst wurden die Zeichen nach folgender Tabelle umcodiert, wobei man schon sieht, dass damals sogar nur ein Subset des ASCII-Codes zulässig war.

Bits	2…0							
5…3	000	001	010	011	100	101	110	111
000	space	A	B	C	D	E	F	G
001	H	I	J	K	L	M	N	O
010	P	Q	R	S	T	U	V	W
011	X	Y	Z	$	.	%	0	1
100	2	3	4	5	6	7	8	9

   Beispielsweise würde der Buchstabe „Q“ zu $(010001)_2$ umcodiert.

2. Im zweiten Schritt wird nach der Formel $rad50 = 40^2 \cdot c_1 + 40 \cdot c_2 + c_3$ aus jeweils drei umcodierten Buchstaben b_i ein 16-bit Wert gemacht, der dann abgespeichert wurde. Die Bezeichnung *Radix-50* kommt wegen der Beziehung $40_{10} = 50_8$.

(a) Wie würde die Zeichenkette „AC9“ in der Radix-50 Darstellung codiert werden? Geben Sie das Ergebnis bitte als Hexadezimalzahl an.

(b) Ersetzen Sie in obiger Formel die Multiplikationen durch Schiebeoperationen und Additionen und schreiben Sie eine möglichst einfache Java-Funktion die Ihre neue Formel implementiert. Dass in Java der Typ `int` 32-bit hat, soll uns dabei nicht weiter stören.

```
int rad50_encode (int c1, int c2, int c3)       // drei Zeichen
{
  ...
}
```

(c) Beschreiben Sie (textuell, ein Programm ist nicht nötig), wie man aus der Radix-50 Darstellung, wieder die drei darin codierten Buchstaben zurückgewinnen kann.

(d) Welche Buchstaben sind in den Radix-50 Werten $(06A4)_{16}$ und $(085B)_{16}$ codiert?

Aufgabe 5.4 (Punkte 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) $y = 10 \cdot x$

(b) $y = -48 \cdot x$

(c) $y = 60 \cdot (x + 5)$