Computerorientierte Mathematik I
Übung 1

Samanta Scharmacher[1]
Nicolas Lehmann[2] (Dipl. Kfm., BSC)

[1] Freie Universität Berlin, FB Mathematik und Informatik, Institut für Informatik, scharbrecht@zedat.fu-berlin.de
[2] Freie Universität Berlin, FB Mathematik und Informatik, Institut für Informatik, AG Datenbanksysteme, Raum 170, mail@nicolaslehmann.de, http://www.nicolaslehmann.de

Lösungen zu den gestellten Aufgaben

Aufgabe 1

Teilaufgabe a)

$55_{10} = 5 \cdot 10^1 + 5 \cdot 10^0 = 55_{10}$
$= 1 \cdot 7^2 + 0 \cdot 7^1 + 6 \cdot 7^0 = 106_7$

Teilaufgabe b)

$42_7 = 4 \cdot 7^1 + 2 \cdot 7^0 = 30_{10}$
$= 1 \cdot 3^3 + 0 \cdot 3^2 + 1 \cdot 3^1 + 0 \cdot 3^0 = 1010_3$

Teilaufgabe c)

$12321_4 = 1 \cdot 4^4 + 2 \cdot 4^3 + 3 \cdot 4^2 + 2 \cdot 4^1 + 1 \cdot 4^0 = 441_{10}$
$= 1 \cdot 2^8 + 1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 110111001_2$

Teilaufgabe d)

$17HAI_{26} = 1 \cdot 26^4 + 7 \cdot 26^3 + H \cdot 26^2 + A \cdot 26^1 + I \cdot 26^0 = 592454_{10}$
$= C \cdot 36^3 + P \cdot 36^2 + 5 \cdot 36^1 + 2 \cdot 36^0 = CY52_{36}$

Aufgabe 2

r lässt sich als k-te Potenz von q darstellen: $\forall k \in \mathbb{N} \setminus \{0, 1\} : r < q^k$.
Für den Fall $k = 1$ gilt: $r = q$.

Um eine Ziffer $a_i \in Z_r$ darzustellen benötigt man k-viele Ziffern $b_i \in Z_q$.

Um eine Zahl a_r mit n vielen Ziffern $a_i \in Z_r$ als Zahl b_q darstellen zu können werden $n \cdot k$ viele Ziffern $b_i \in Z_q$ benötigt, wenn jede Ziffer a_i dargestellt werden soll.

Zur Erinnerung, es gilt: $a_{n-1} \neq 0$

Falls die Darstellung der ersten Ziffer a_{n-1} in b_q, also $b_{m-1}b_{m-2}...b_{m-k-1}$ nicht 0 sind, hat die Zahl b_q tatsächlich auch $m = n \cdot k$ viele Ziffern.

Falls die Darstellung der ersten Ziffer a_{n-1} in b_q, also b_{m-1}, $b_{m-1}b_{m-2}$,...,$b_{m-1}b_{m-2}...b_{m-k-1}$ gleich 0 sind, hat die Zahl b_q zwischen $(n-1) \cdot k$ und $n \cdot k$ viele Ziffern.

Daher gilt: $(n-1) \cdot k \leq m \leq n \cdot k$

Aufgabe 3

Datei: `run_1_3.m`

```
%%% Uebung 1 %%%

%%% clean environment
clear all
clc
close all

disp('Ausgaben der dual1(z,b) Funktion:')
d1_p15_p8   = dual1(15,8)
d1_p42_p8   = dual1(42,8)
d1_n77_p8   = dual1(-77,8)
d1_p714_p8  = dual1(714,16)
d1_n512_p16 = dual1(-512,16)
d1_n77_p16  = dual1(-77,16)

disp('Ausgaben der dual2(z,b) Funktion:')
d2_p15_p8   = dual2(15,8)
d2_p42_p8   = dual2(42,8)
d2_n77_p8   = dual2(-77,8)
d2_p714_p16 = dual2(714,16)
d2_n512_p16 = dual2(-512,16)
d2_n77_p16  = dual2(-77,16)
```

Datei: `dual1.m`

```
function [A] = dual1(z,b)

result = zeros(1,b); % Initialisierung eines b-Bit langen Arrays

if b < 2
    error('Es muessen wenigstens 2 Bit verwendet werden (b>1)!')
end

% Vorzeichenbit setzen
if z < 0
    result(1,1) = 1;
    z=-z;
end

% Binaerzahl berechnen
for i = length(result):-1:2

    % falls z = 0 ist, beende die for-Schleife
    if (z==0)
        break;
    end

    r = mod(z,2);                              % mod(z,2)
    z = idivide(int32(z), int32(2), 'floor'); % div(z,2)

    % Fuege r an die passende Position des Ergebnisarrays ein.
    result(1,i) = r;

end

A = result;

end %eof
```

Datei: dual2.m

```
function [A] = dual2(z,b)

result = zeros(1,b); % Initialisierung eines b-Bit langen Arrays

if b < 2
    error('Es muessen wenigstens 2 Bit verwendet werden (b>1)!')
end

if z < 0
    n = -z;
else
    n = z;
end

% Binaerzahl berechnen
for i = b:-1:2

    % falls z = 0 ist, beende die for-Schleife
    if (n==0)
        break; % breche Schleife ab
    end

    r = mod(n,2);                               % mod(z,2)
    n = idivide(int32(n), int32(2), 'floor'); % div(z,2)

    % Fuege r an die passende Position des Ergebnisarrays ein.
    result(1,i) = r;

end

if z < 0

    % Bit-Flip
    result(1,result(1,:)==0) = 2;
    result(1,result(1,:)==1) = 0;
    result(1,result(1,:)==2) = 1;

    % addiere 1
    imSinn = 1;
    for i = b:-1:2
       if (result(1,i) == 0) && (imSinn == 0)
          result(1,i) = 0;
       elseif (result(1,i) == 0) && (imSinn == 1)
          result(1,i) = 1;
```

```
            imSinn = 0;
         elseif (result(1,i) == 1) && (imSinn == 0)
            result(1,i) = 1;
         elseif (result(1,i) == 1) && (imSinn == 1)
            result(1,i) = 0;
         end
      end

      % Vorzeichenbit setzen
      result(1,1) = 1;
end

A = result;

end %eof
```

Aufgabe 4

Teilaufgabe a)

```
     15 = 01111
     -5 = 11011 <-- 11010 <-- 00101
15+(-5) = 01111 + 11011 = 01010
     10 = 01010

     15 = 01111
      5 = 00101
   15-5 = 01111 - 00101 = 01010
     10 = 01010
```

Solange man in der Rang des b-Bit Zweierkomplement bleibt macht es keinen Unterschied, ob man im Zweierkomplement rechnet oder nicht. Sobald die Range überschritten wird, würde ein falsches Ergebnis entstehen.

Teilaufgabe b)

Rechnung für $3 + (-2)$

```
-------------- Zweierkomplement
     3 = 00011
    -2 = 11110 <-- 11101 <-- 00010
3+(-2) = 00011 + 11011 = 00001
     1 = 00001

-------------- Einerkomplement
     3 = 00011
    -2 = 11101 <-- 00010
3+(-2) = 00011 + 11101 = 00000
     0 = 00000
```

Rechnung für $3 + (-3)$

```
-------------- Zweierkomplement
     3 = 00011
    -3 = 11101 <-- 11100 <-- 00011
3+(-3) = 00011 + 11101 = 00000
     0 = 00000

-------------- Einerkomplement
     3 = 00011
    -3 = 11100 <-- 00011
3+(-3) = 00011 + 11100 = 11111
    -1 = 11111
```

Rechnung für $3 + (-4)$

```
-------------- Zweierkomplement
     3 = 00011
    -4 = 11100 <-- 11011 <-- 00100
3+(-4) = 00011 + 11100 = 11111
    -1 = 11111

-------------- Einerkomplement
     3 = 00011
    -4 = 11011 <-- 00100
3+(-4) = 00011 + 11011 = 11110
    -2 = 11110
```

Das Ergebnis wäre immer 1 kleiner als es seien sollte?!
Durch das addieren der 1 entsteht eine Symmetrie zur 0. (Beispiel: $x_2^{(b)} - x_2^{(b)} = 0$)

Teilaufgabe c)

Die mögliche Range einer 5-Bit Zweierkomplementzahl wird überschritten.

$$minBound(Zweierkomplement_{5Bit}) = -2^4 = -16$$
$$maxBound(Zweierkomplement_{5Bit}) = 2^4 - 1 = 15$$