Universität Leipzig **WS 14/15**
Institut für Informatik **Dr. Monika Meiler**

Modellierung und Programmierung 1
Übungsserie 1

Lösungsvorschläge

1. **Konvertierung**

   a) Natürliche Zahlen

Basis 2	**Basis 10**	**Basis 16**
1100 0100 0100 0010	50 242	C4 42
11 0100 0100	836	344
1100 1010 1111 1110	51 966	CA FE

   b) Rationale Zahlen

Basis 2	**Basis 10**	**Basis 16**
1101.1101	13.812 5	D.D
101.011	5.375	5.6
1 1011.0110 1	27.406 25	1B.68

   c) i. Lösung der quadratischen Gleichung $b^2 + 2b + 3 = 198$ (negatives b unzulässig): $b = 13$

   ii.

$$x = (0.\overline{22})_3 = (0.\overline{2})_3 = \frac{(2)_3}{(2)_3} = 1$$

oder

$$x = (0.\overline{2})_3 \rightarrow 3 \cdot (0.\overline{2})_3 = 2 + (0.\overline{2})_3 \rightarrow 3 \cdot x = 2 + x \rightarrow x = \frac{2}{2} = 1$$

oder ($\rightarrow$ Grenzwert geometrischer Reihen)

$$\begin{aligned} x = (0.\overline{2})_3 &= 2 \cdot \sum_{k=1}^{\infty} \left(\frac{1}{3}\right)^k \\ &= \frac{2}{3} \cdot \sum_{k=0}^{\infty} \left(\frac{1}{3}\right)^k \\ &= \frac{2}{3} \cdot \frac{1}{1 - \frac{1}{3}} = \frac{2}{3} \cdot \frac{3}{2} = 1 \end{aligned}$$

2. **Maschinenzahlen**

   a) i. -123

$$\begin{array}{rcrll} 123 & : 2 = & 61 & \text{Rest} & 1 \\ 61 & : 2 = & 30 & \text{Rest} & 1 \\ 30 & : 2 = & 15 & \text{Rest} & 0 \\ 15 & : 2 = & 7 & \text{Rest} & 1 \\ 7 & : 2 = & 3 & \text{Rest} & 1 \\ 3 & : 2 = & 1 & \text{Rest} & 1 \\ 1 & : 2 = & 0 & \text{Rest} & 1 \end{array}$$

$\rightarrow 123 = (111\ 1011)_2 \rightarrow -123 = (000\ 0101)_2 \rightarrow$ `int`-Maschinenzahl:

1111 1111	1111 1111	1111 1111	1000 0101

ii. 123.515 625
Ganzzahliger Anteil (s.o.): $123 = (111\ 1011)_2$
Gebrochener Anteil: $0.515\ 625 = (0.1000\ 01)_2$

$$\begin{array}{llll}
0.515625 & \cdot 2 = & 1.03125 & \rightarrow\ 1 \\
0.03125 & \cdot 2 = & 0.0625 & \rightarrow\ 0 \\
0.0625 & \cdot 2 = & 0.125 & \rightarrow\ 0 \\
0.125 & \cdot 2 = & 0.25 & \rightarrow\ 0 \\
0.25 & \cdot 2 = & 0.5 & \rightarrow\ 0 \\
0.5 & \cdot 2 = & 1.0 & \rightarrow\ 1
\end{array}$$

$\rightarrow 123.515\ 625 = (111\ 1011.1000\ 01)_2 = (1.1110\ 1110\ 0001)_2 \cdot 2^6$
$\rightarrow M' = 1110\ 1110\ 0001, E' = 6 + 127 = 5 + 128 = (1000\ 0101)_2, s = 0$
$\rightarrow$ `float`-Maschinenzahl:

0	1000 0101	1110 1110 0001 0000 0000 000

b) Betrachte Bitfolge:

i. **`int`-Maschinenzahl**
$(1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010)_2 = -(2^{31} - 2) = -2\ 147\ 483\ 646$

ii. **`float`-Maschinenzahl**

1	0000 0000	0000 0000 0000 0000 0000 010

$E' = (0000\ 0000)_2 = 0\ \ M' = (0000\ 0000\ 0000\ 0000\ 0000\ 010)_2$
$\rightarrow$ Sonderfall: $z = (-1)^s \cdot 0.M' \cdot 2^r = -2^{-22} \cdot 2^{-126} = -2^{-148} \approx -2.8 \cdot 10^{-45}$

c) i. Da der größte Exponent für ∞ bzw. *NaN* reserviert ist, ist die größte positive `float`-Maschinenzahl x_{max}:

0	1111 1110	1111 1111 1111 1111 1111 111

$$\begin{aligned}
x_{max} &= (1.1111\ 1111\ 1111\ 1111\ 1111\ 111)_2 \cdot 2^{127} \\
&= (1111\ 1111\ 1111\ 1111\ 1111\ 1111)_2 \cdot 2^{-23} \cdot 2^{127} \\
&= (2^{24} - 1) \cdot 2^{104} = 2^{128} - 2^{104} \approx 3.4 \cdot 10^{38}
\end{aligned}$$

Für den größten Rundungsfehler ϵ_{max} wird die zweitgrößte float-Zahl ermittelt:

0	1111 1110	1111 1111 1111 1111 1111 110

$$\epsilon_{max} = \frac{x_{max} - x_{max-1}}{2} = \frac{2^{-23} \cdot 2^{127}}{2} = 2^{103} \approx 1.0141 \cdot 10^{31}$$

ii. Da der Exponent 0 für die Darstellung der 0 und zur Darstellung von Zahlen im Unterlauf reserviert ist, ist die kleinste positive `float`-Maschinenzahl x_{min}:

0	0000 0001	0000 0000 0000 0000 0000 000

$$x_{min} = (1.0)_2 \cdot 2^{-126} = 2^{-126} \approx 1.2 \cdot 10^{-38}$$

Für den kleinsten Rundungsfehler ϵ_{min} wird die zweitkleinste float-Zahl ermittelt:

0	0000 0001	0000 0000 0000 0000 0000 001

$$\epsilon_{min} = \frac{x_{min-1} - x_{min}}{2} = \frac{2^{-23} \cdot 2^{-126}}{2} = 2^{-150} \approx 7.0065 \cdot 10^{-46}$$

3. **Algorithmen und Programme**

a) Gegenbeispiel: Für $n = 41$ liefert $f(n) = 41^2 + 41 + 41 = 41 \cdot (41 + 2) = 41 \cdot 43$ zwei Teiler.

b) $\boldsymbol{p = prim(n)}$

```
p = 0;
if( n > 1)
{
  if( n == 2)
  {
    p = 1;                                      // 2 ist Primzahl
  }
  else
  {
    t = 2; p = n
    if( p != 0)                                 // n ist ungerade
    {
      a = n / t; t = 3;
      while( a > t)                             // n >= t * t
      {
        p = n % t;
        if( p != 0)
        {
          p = 1; a = n / t;                     // Abbruchkriterium
          t = t + 2;
        }
        else
        {
          a = t;                                // erzwungener Abbruch
        }
      }
    }
  }
}
```

Protokoll für $n = 37$

n	p	t	a
37	0		
37	1	2	18
37	1	3	12
37	2	5	12
37	1	7	7

Die Zahl 27 ist eine Primzahl.

Protokoll für $n = 679$

n	p	t	a
679	0		
679	1	2	339
679	1	3	226
679	4	5	226
679	1	7	135
679	0	7	7

Die Zahl 679 ist keine Primzahl ($679 = 7 \cdot 97$).

4. **JavaScript**

   a) siehe **HalloWeltAppletPlus.java**

   b) siehe **HalloWeltAppletPlus.html**