BERGISCHE UNIVERSITÄT
GESAMTHOCHSCHULE WUPPERTAL
GAUSS-STRASSE 20
42097 WUPPERTAL
(Korrespondenzanschrift)
42119 WUPPERTAL
(Lieferanschrift)
TELEX 8 592 262 bughw
TELEFAX (0202) 439-2901
TELEFON (0202) 439-1

Fachbereich 7

MATHEMATIK

Prof. Dr. Hans-Jürgen Buhl
Praktische Informatik / Numerik

e-mail: Juergen.Buhl@math.uni-wuppertal.de

Programming by Contract

WS 2003/2004 – Übungsblatt 6

18. Dezember 2003
Ausgabe: 8. Dezember 2003

Aufgabe 1. *Spezifikation der Potenzfunktion*

a) Spezifizieren Sie eine Funktion x^f zur Berechnung (echt) gebrochener Potenzen von x (also: $0 \leq f < 1$).

b) Schreiben Sie einen entsprechenden Algorithmus. Benutzen Sie dabei nicht den *log()* sondern die Beziehungen:

$$x^f = (\sqrt{x})^{2f}$$

$$x^{1+f} = x \cdot x^f$$

Aufgabe 2. *Ackermann-Funktion*

Die Ackermann-Funktion wird definiert als

```
ackermann(n,m) =
        If n = 0
        Then m + 1
        Else
                If m = 0
                Then ackermann(n-1, 1)
                Else ackermann(n-1, ackermann(n, m-1))
```

Spezifizieren Sie eine implementierbare Version. (Berücksichtigen Sie den möglichen Floatingpoint-Overflow (`Infty`).) Berechnen Sie die Ackermann-Funktion für:

$$(n, m) \in \{(1, 1), (2, 2), (3, 3)\}$$

Das Abfangen von Overflows ist bei der Ackermann-Funktion besonders wichtig, da sie sehr schnell wächst: $ackermann(4, 2)$ besitzt bereits über 21000 Ziffern und $ackermann(4, 4)$ ist größer als $10^{10^{10^{21000}}}$.

Aufgabe 3. *Akkumulation*

Bei numerischen Approximationen wird häufig mit äquidistanten Gittern

$$\Delta x = \frac{b-a}{N}$$

gerechnet. Warum ist das effiziente Verfahren der Akkumulation

$$y = a + i * \Delta x \quad (i = 0, 1, ..., N)$$

dem Verfahren

$$y = \frac{a * (N-i) + i * b}{N}$$

unterlegen? Welche Werte von N sollte man bevorzugen? In welcher Reihenfolge sollte man in Analogie zur Mittelwertberechnung die einzelnen Rechenoperationen ausführen?

Aufgabe 4. *Virtuelle Annotationsfunktionen*

Schreiben Sie eine *virtuelle Annotationsfunktion*

$$\texttt{MAX\_IN\_SLICE}(\texttt{X}, \texttt{A}(\texttt{I..J})) \stackrel{\text{def}}{=} \left(X := \max_{j:=I,...,J} A(j) \right)$$

ähnlich wie die Funktion ORDERED:

```
--: function ORDERED(A : VECTOR) return BOOLEAN
--|      where
--|      return for all I, J : A'RANGE => I ≤ J →
--|                          A(I) ≤ A(J);
```