Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für theoretische Informatik
Dr. B. Gärtner

9. November 2004

Informatik für Mathematiker und Physiker Serie 4 WS 04/05

URL: `http://www.ti.inf.ethz.ch/ew/courses/Info1_04/`

Aufgabe 1 [Schnellübung – 20 Min.] (5 Punkte)

Gruppe A–H

Schreiben Sie ein Programm `fak.C`, das eine natürliche Zahl n einliest und $n! := \prod_{i=1}^{n} i$ berechnet und ausgibt.

Gruppe I

Schreiben Sie ein Programm `three.C`, das eine nichtnegative ganze Zahl n einliest und 3^n berechnet und ausgibt.

Gruppe J und K

Schreiben Sie ein Programm `even.C`, das eine natürliche Zahl n einliest und die Summe der ersten n geraden Zahlen berechnet und ausgibt. Beispiel: Für $n = 3$ ergibt sich $2 + 4 + 6 = 12$.

Aufgabe 2 (2 Punkte)

Werten Sie folgende Ausdrücke „per Hand“ **in Einzelschritten** und in der durch Priorität und Links-/Rechtsassoziativität sowie den Regeln für boolsche Ausdrücke bestimmten Reihenfolge aus. Zum Beispiel:

```
3 + 4 < 8 && 3 == 5 - 3
   ⟶   7 < 8 && 3 == 5 - 3
   ⟶   true && 3 == 5 - 3
   ⟶   true && 3 == 2
   ⟶   true && false
   ⟶   false.
```

a) `5 * 7 <= 32 && 11 == 2 * 5 + 1 || 6 > 5`

b) `3 * 2 > 2 || 1.0 / 0.0 != 0.0 && 3 + 4 >= 7 || 2 < 8 + 0.876`

Aufgabe 3 (5 Punkte)

Schreiben Sie ein Programm `randmax.C`, das eine natürliche Zahl n einliest und das Maximum von n Zufallszahlen berechnet. Das Programm soll ausgeben, wie oft sich das Maximum im Verlaufe der Berechnung geändert hat. Vergleichen Sie die Ergebnisse für verschiedene n mit denen des Programms `harmonic`.

Hinweis: Verwenden Sie den Pseudozufallszahlengenerator aus der Vorlesung. Das Programm `random.C` finden Sie auf der Vorlesungswebseite.

Eidgenössische Technische Hochschule Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für theoretische Informatik
Dr. B. Gärtner

9. November 2004

Informatik I: Material aus der Vorlesung

Programm: newton.C

```
// Programm: newton.C
// Berechnet Quadratwurzelm mittels Newton-Iteration.

#include <iostream>

int main()
{
  // Eingabe
  std::cout << "Berechnung der Quadratwurzel von n,"
            << " n > 0. Eingabe n :? ";
  double n;
  std::cin >> n;

  // Newton-Iteration: x_{i+1} = 1/2 (x_i + n/x_i)
  // Um sicherzustellen, dass der Algorithmus terminiert,
  // halten wir eine untere Schranke fuer den aktuellen
  // Approximationswert aufrecht. Diese Schranke wird
  // hochgesetzt, wann immer der folgende Wert groesser
  // wird als der aktuelle. Auf diese Weise wird ein
  // etwaiger Zyklus erkannt und abgebrochen.

  double curr = n;           // aktueller Wert
  double prev = 0;           // voriger Wert
  double lb = 0;             // untere Schranke
  unsigned int counter = 0;  // Anzahl Iterationen

  do {
    ++counter;
    if (curr > prev)
      lb = prev;
    prev = curr;
    curr = (prev + n / prev) / 2;
  } while (curr != prev && curr > lb);

  // Ausgabe
  std::cout << "Approximation der Quadratwurzel aus "
            << n << " ist: " << curr
            << "\nApproximationsfehler des Quadrats ist "
            << curr * curr - n << "."
            << "\nAnzahl der Iterationen war "
            << counter << "." << std::endl;
  return 0;
}
```

Programm: bigsmall.C

```
// Programm: bigsmall.C
// demonstriert drastisch den Einfluss der
// Summationsreihenfolge auf das Ergebnis
// von Fliesskommaadditionen

#include <iostream>

int main()
{
  const unsigned int m = 16777216u; // 2^24

  //      -24   -24          -24
  // 1 + 2   + 2    + ... + 2
  //     ------- 2^24 mal ------

  float sum1 = 1.0f;
  for (unsigned int i = 0; i < m; ++i)
    sum1 += 1.0f / m; // kein Effekt

  //  -24   -24          -24
  // 2   + 2    + ... + 2    + 1
  // ------- 2^24 mal ------

  float sum2 = 0.0f;
  for (unsigned int i = 0; i < m; ++i)
    sum2 += 1.0f / m; // exakte Addition
  sum2 += 1.0f;       // exakte Addition

  std::cout << "Vorwaertssumme: " << sum1
            << "\nRueckwaertssumme: " << sum2
            << std::endl;
  return 0;
}
```

Abgabe: Aufgabe 2: am 16. November 2004, in der Pause der Vorlesung, schriftlich.
Aufgabe 3: bis 15. November 2004, 16.00 Uhr, per Email.