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

23. November 2004

Informatik für Mathematiker und Physiker Serie 6 WS 04/05

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

Ein Lindenmayer-System, benannt nach dem Biologen Aristid Lindenmayer (1925–1985), ist ein Tupel (Σ, P, s), wobei

- Σ eine endliche Menge (*Alphabet*),
- P eine Abbildung $\Sigma \to \Sigma^*$ (*Generator*) und
- $s \in \Sigma^*$ ein Wort (*Startwort*) ist.

Für eine Menge Σ bezeichnen wir mit Σ^* die Menge aller *Wörter*, d.h. aller endlichen Folgen von Elementen aus Σ (inklusive des leeren Wortes, das wir mit ε bezeichnen). Der Generator beschreibt eine Abbildung $\Pi : \Sigma^* \to \Sigma^*$ durch $\sigma_1 \dots \sigma_n \mapsto \Pi(\sigma_1 \dots \sigma_n) := P(\sigma_1) \dots P(\sigma_n)$, $n \in \mathbb{N}_0$. Ein Lindenmayer-System erzeugt ein Folge von Wörtern $w_0, w_1, \dots$ aus Σ^*, wobei $w_i := \Pi^i(s)$, für $i \in \mathbb{N}_0$ (insbesondere $w_0 = s$). Beispiel: Für das System $(\{X\}, \{X \mapsto XX\}, X)$ ist $w_i = X^{2^i}$.

Wir betrachten im folgenden Lindenmayer-Systeme über dem Alphabet $\Sigma' := \{+, -, X, Y\}$. Wörter über diesem Alphabet kann man grafisch interpretieren, indem man die einzelnen Symbole in Turtle-Grafik Befehle übersetzt. Hierbei entsprechen

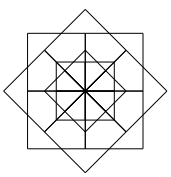
- + einer Rechtsdrehung um einen Winkel α,
- − einer Linksdrehung um einen Winkel α und
- alle anderen Symbole einer Vorwärtsbewegung um einen Schritt,

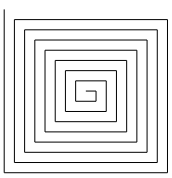
für einen beliebigen aber festen Winkel α. Beispiel: Das Wort "X+Y+X+Y" für $\alpha = 90°$ entspricht einem Quadrat.

Die durch ein Lindenmayer-System definierten Wörter kann man sehr einfach rekursiv berechnen. Hierzu definiert man für jedes Symbol σ des Alphabets eine Funktion mit Parameter $i \in \mathbb{N}_0$, welche $\Pi^i(\sigma)$ berechnet.

Aufgabe 1 *(Diese Aufgabe wird in der Übungsstunde gelöst.)*

Schreibe Programme, die folgende Grafiken mit Hilfe der Turtle-Grafik Funktionen zeichnen.

a) `squares.C`: 

b) `spiral.C`: 

c) Schreibe ein Programm `lind.C`, das eine nichtnegative Zahl i einliest und w_i für das Lindenmayer-System $(\{X, +\}, \{X \mapsto X+X+,\ + \mapsto +\}, X)$ mit Winkel $\alpha = 90°$ grafisch ausgibt.

Abgabe: Aufgabe 2: bis 29. November 2004, 16.00 Uhr, per Email.
Aufgabe 3: am 30. November 2004, in der Pause der Vorlesung, schriftlich.
In der nächsten Woche wird es eine Schnellübung geben!

Aufgabe 2 (6 Punkte)

Schreibe Programme, die eine nichtnegative Zahl i einlesen und w_i für die folgenden Lindenmayer-System (Σ', P, s) grafisch ausgeben.

a) Ein Programm `lind-ko.C` für $P := \{X \mapsto X\text{-}X\text{++}X\text{-}X,\ + \mapsto +,\ - \mapsto -\}$, $\alpha := 45°$ und $s := X+X+X+X+X+X+X+X+$.

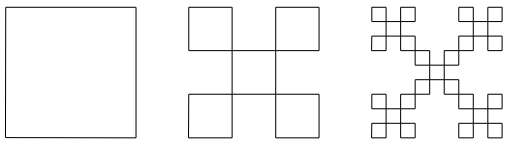
b) Ein Programm `lind-st.C` für $P := \{X \mapsto Y+X+Y,\ Y \mapsto X\text{-}Y\text{-}X,\ + \mapsto +,\ - \mapsto -\}$, $\alpha := 60°$ und $s := Y$.

c) Ein Programm `lind-go.C` für $P := \{+ \mapsto +,\ - \mapsto -,\ X \mapsto X+Y\text{++}Y\text{-}X\text{--}XX\text{-}Y+,\ Y \mapsto -X+YY\text{++}Y+X\text{--}X\text{-}Y\}$, $\alpha := 60°$ und $s := Y$.

Aufgabe 3 (6 Punkte)

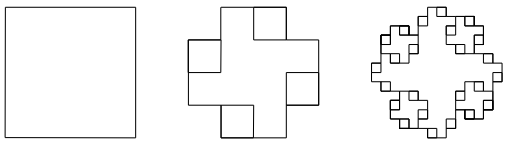
Gib Lindenmayer-Systeme (Σ, P, s) an, welche folgende Wörter generieren.

a)



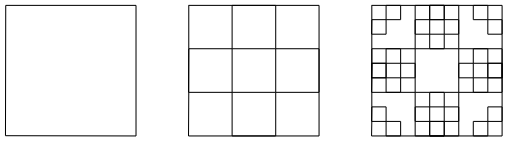
w_0 w_1 w_2

b)



w_0 w_1 w_2

c)



w_0 w_1 w_2

Material aus der Vorlesung

Programm: ggt.h

```
unsigned int ggt(unsigned int a, unsigned int b)
// POST: Rueckgabe ist der groesste gemeinsame
// Teiler von a und b, wobei ggt(x,0):=ggt(0,x):=x.
{
  if (b == 0) return a;
  return ggt(b, a % b);
} // ggt(a, b)
```

Programm: fib.h

```
unsigned int fib(unsigned int n)
// POST: Rueckgabe ist die n-te Fibonacci-Zahl F_n, wobei
//   F_0:=0, F_1:=1 und F_i:=F_{i-1}+F_{i-2}, fuer i>1.
{
  if (n == 0) return 0;
  if (n <= 2) return 1;
  return fib(n-1) + fib(n-2);
} // fib(n)
```

Programm: ackermann.h

```
unsigned int A(unsigned int n, unsigned int m)
// POST: Rueckgabe ist Wert der Ackermann-Funktion
//   A(n,m) : N_0 x N_0 -> N, wobei
//            / m+1            , fuer n=0;
//   A(n,m) := | A(n-1, 1)      , fuer n>0 und m=0;
//            \ A(n-1,A(n,m-1)), fuer n>0 und m>0.
{
  if (n == 0) return m+1;
  if (m == 0) return A(n-1, 1);
  return A(n-1, A(n, m-1));
} // A(n,m)
```

Programm: login.h

```
// Programm: login.h
// Unbekannte login-Funktion...

#include <string>

bool login(std::string s);
// POST: true genau dann, wenn s das gueltige Passwort ist.
```

Programm: code.C

```
// Programm: code.C
// Testet alle 0/1 strings der Laenge 50 mit 4 Einsen.

#include <iostream>
#include <login.h>

void crack(std::string prefix, unsigned int n, unsigned int o)
// POST: Ruft fuer jeden 0/1 string s der Laenge n mit o Einsen
//   login(prefix+s) auf. Falls login(prefix+s), wird prefix+s
//   nach std::cout ausgegeben.
{
  if (n < o) return;
  if (n == 0) {
    if (login(prefix)) std::cout << prefix << std::endl;
  } else {
    crack(prefix + "0", n-1, o);
    crack(prefix + "1", n-1, o-1);
  }
}

int main()
{
  crack("", 50, 4);
  return 0;
}
```