# Programmieren mit MuPAD - modulares Rechnen

Thomas Wassong

Arbeitsgruppe Didaktik
FB 17 Mathematik
Universität Kassel
wassong@mathematik.uni-kassel.de

03. Dezember 2008

## Übersicht





## Die Fibonacci-Zahlen

Wir haben schon des Öfteren die Fibonacci-Zahlen hier in der Vorlesung besprochen. Hier nochmal die Definition:

$$F_1 := 1,$$
$$F_2 := 1,$$
$$F_n := F_{n-1} + F_{n-2}.$$



## eine schnellere Methode

Man hat herausgefunden, dass folgende Bedingungen für die Fibonacci-Zahlen gelten:

$$F_{2n} = F_n(F_n + 2F_{n-1})$$
$$F_{2n+1} = F_{n+1}^2 + F_n^2$$

Bei genauerer Betrachtung fällt folgendes auf: Die Rekursionstiefe für die Berechnung Fibonacci-Zahl verringert sich erheblich. Somit verringert sich auch die Zeit zur Berechnung einer Fibonacci-Zahl. Dies wollen wir nun überprüfen.



## Aufgabe

Die obige Vermutung werden wir nun überprüfen. Dazu folgende Aufgabenstellung:

- Programmieren Sie eine rekursive Prozedur zur Berechung der Fibonacci-Zahlen nach der obigen Methode.
- Vergleichen Sie die Rechenzeiten Ihrer Funktion mit der eingebauten Funktion numlib::fibonacci für n=1.000.000.



## Der Euklidische Algorithmus

Sie haben in der Vorlesung von Prof. Dr. Seiler den Euklidischen Algorithmus kennengelernt. Zur Wiederholung sehen Sie hier den Algorithmus in seiner rekursiven Form:

```
Algorithmus EA:
Eingabe: zwei natürliche Zahlen x und y mit x<=y
Ausgabe: d=ggt(x,y)
1: if x|y then
2:    return x
3: else
4:    return EA(y mod x, x)
5: end if
```



## Aufgabe

Nun wollen wir diesen Algorithmus `EA` einmal in MuPAD umsetzen. Wir setzen dabei aber nicht zwingend voraus, dass $x \leq y$ gilt
Dazu folgende Bemerkung:
`x|y` setzt man am Besten mit der Abfrage `(x mod y = 0)` um.
Berechnen Sie danach das Ergebnis von $ggt(1234, 56789)$.



## Der erweiterte euklidische Algorithmus

Nachdem wir nun den Euklidischen Algorithmus programmiert haben, stellt sich noch die Frage nach den Bézout-Koeffizienten, also ganze Zahlen $r$ und $s$, so dass gilt:

$$ggt(x, y) = r * x + s * y$$

z.B. $ggt(35, 126) = -7 * 35 + 2 * 126 = 7$, mit den Bézout-Koeffizienten $r = -7$ und $s = +2$.

## Der erweiterte Euklidische Algorithmus

```
Algorithmus EEA:
Eingabe: zwei natürliche Zahlen x und y mit x<=y
Ausgabe: Eine Liste mit den Einträgen [d, r, s],
         wobei d=ggt(x,y) und r und s die beiden
         Bezout-Koeffizienten sind
1: if x|y then
2:    return [x,1,0]
3: else
4:    [g,rAlt,sAlt] = EEA(y mod x, x)
5:    r = sAlt - rAlt(y div x); s = rAlt;
6:    return [g,r,s]
7: end if
```



## Aufgabe

Programmieren Sie den erweiterten Euklidischen Algorithmus.