```
> restart:
```

## Musterlösung Aufgabe 3

### *Iteration versus Rekursion (mit option remember)*

**Schreiben Sie zwei Prozeduren zur effizienten Berechnung des $n$-ten Fibonacci-Polynoms (Leonardo Fibonacci). Die Fibonacci-Polynome sind durch die Rekursion**

$F_0 = 0, F_1 = 1, F_n = x\,F_{n-1} + F_{n-2}$

**definiert. Programmieren Sie die Rekursion**

**a) mittels "for loop" und "double assignment" (Doppelzuweisung, allgemein Mehrfachzuweisung)**

**Hinweis: Expandieren Sie die Polynome in jedem Rekursionsschritt (expand), sowohl in a) als auch in b).**

```
> F1:=proc(n::nonnegint,x::symbol)
      local s0,s1,i;
      s0,s1:=0,1;
      for i from 1 to n do
          s0,s1:=s1,expand(x*s1+s0);
      end do;
      s0;
  end proc:
```

```
> seq(F1(i,x),i=0..5);
```

$$0, 1, x, x^2+1, x^3+2\,x, x^4+3\,x^2+1 \qquad (1)$$

```
> F1(20,y);
```

$$y^{19}+18\,y^{17}+136\,y^{15}+560\,y^{13}+1365\,y^{11}+2002\,y^9+1716\,y^7+792\,y^5+165\,y^3+10\,y \qquad (2)$$

**Der Datentyp `name` erlaubt Symbole und indizierte Symbole, der Datentyp `symbol` dagegen nur Symbole.**

```
> F1(10,y[1]);
Error, invalid input: F1 expects its 2nd argument, x, to be of type symbol, but received y[1]
```

**b) mittels rekursiver Prozedur mit "option remember".**

```
> F2:=proc(n::nonnegint,x::symbol)
      option remember;
      if n <= 1 then n
```

```
      else
         expand(x*F2(n-1,x)+F2(n-2,x))
      end if
  end proc:
```

```
> F2(20,y);
```

$$y^{19} + 18\,y^{17} + 136\,y^{15} + 560\,y^{13} + 1365\,y^{11} + 2002\,y^{9} + 1716\,y^{7} + 792\,y^{5} + 165\,y^{3} + 10\,y \tag{3}$$

**c) Vergleichen Sie die CPU-Zeiten für die Berechnungsmethoden in a) und b) bei** $n = 3000$**. Erklären Sie das beobachtete Laufzeitverhalten.**

```
> forget(F2);
> time(F1(3000,x));
```

$$2.171 \tag{4}$$

```
> time(F2(3000,x));
```

$$8.984 \tag{5}$$

**Die größer werdende "remember table" bei Algorithmus b) verursacht zunehmend Speicherverwaltungszeiten, die in Algorithmus a) vermieden werden.**

```
> op(4,eval(F2))[(100,x)];
```

$$\begin{aligned} & x^{99} + 98\,x^{97} + 4656\,x^{95} + 142880\,x^{93} + 3183545\,x^{91} + 54891018\,x^{89} + 762245484\,x^{87} \\ & + 8760554088\,x^{85} + 84986896995\,x^{83} + 706252528630\,x^{81} + 5085018206136\,x^{79} \\ & + 32006008361808\,x^{77} + 177366629671686\,x^{75} + 870366750378300\,x^{73} \\ & + 3799541229226200\,x^{71} + 14810760713140560\,x^{69} + 51705423561053205\,x^{67} \\ & + 162042085745554410\,x^{65} + 456703981505085600\,x^{63} + 1159120046626942400\,x^{61} \\ & + 2651487106659130740\,x^{59} + 5469191608792974920\,x^{57} + 10173461314258261040\,x^{55} \\ & + 17061084191613232800\,x^{53} + 25778699578994555700\,x^{51} + 35059031427432595752\,x^{49} \\ & + 42858025944553776096\,x^{47} + 47011188276065582912\,x^{45} + 46171702771135840360\,x^{43} \\ & + 40498346384007444240\,x^{41} + 31627280033224861216\,x^{39} + 21912870037044995008\,x^{37} \\ & + 13413576695470557606\,x^{35} + 7219428434016265740\,x^{33} + 3397378086595889760\,x^{31} \\ & + 1388818294740297792\,x^{29} + 489462003181042451\,x^{27} + 147405545359541742\,x^{25} \\ & + 37539612570341700\,x^{23} + 7984465725343800\,x^{21} + 1397281501935165\,x^{19} \\ & + 197548686920970\,x^{17} + 22057981462440\,x^{15} + 1889912732400\,x^{13} + 119653565850\,x^{11} \\ & + 5317936260\,x^{9} + 154143080\,x^{7} + 2598960\,x^{5} + 20825\,x^{3} + 50\,x \end{aligned} \tag{6}$$

**d) Für** $x = 1$ **definiert die obige Rekursion die Fibonacci-Zahlen. Modifizieren Sie die Parameterlisten der Prozeduren in a) und b) so, dass sowohl Fibonacci-Polynome als**

auch Fibonacci-Zahlen berechnet werden können.

```
> F1a:=proc(n::nonnegint,x::{symbol,1})
      local s0,s1,i;
      s0,s1:=0,1;
      for i from 1 to n do
          s0,s1:=s1,expand(x*s1+s0);
      end do;
      s0;
  end proc:
```

```
> F1a(6,1);
```

$$8 \tag{7}$$

```
> F1a(6,2);
Error, invalid input: F1a expects its 2nd argument, x, to be of type {1, symbol}, but received 2
```

```
> F1a(6,x);
```

$$x^5 + 4x^3 + 3x \tag{8}$$

```
> F1a(6,x[1]);
Error, invalid input: F1a expects its 2nd argument, x, to be of type {1, symbol}, but received x[1]
```

```
> F2a:=proc(n::nonnegint,x::{symbol,1})
      option remember;
      if n <= 1 then n
      else
         expand(x*F2a(n-1,x)+F2a(n-2,x))
      end if
  end proc:
```

```
> seq(F2a(i,y),i=0..6);
```

$$0, 1, y, y^2 + 1, y^3 + 2y, y^4 + 3y^2 + 1, y^5 + 4y^3 + 3y \tag{9}$$

```
> seq(F2a(i,1),i=0..6);
```

$$0, 1, 1, 2, 3, 5, 8 \tag{10}$$

```
> forget(F2a);
> time(F1a(3000,1));
```

$$0.015 \tag{11}$$

```
> time(F2a(3000,1));
```

$$0. \tag{12}$$

```
> op(4,eval(F2a))[(100,1)];
```

(13)

$$354224848179261915075 \tag{13}$$

**Probe:**

```
> with(combinat, fibonacci):
  seq(fibonacci(i),i=0..5);

  seq(fibonacci(i,z),i=0..6);
```

$$0, 1, 1, 2, 3, 5$$

$$0, 1, z, z^2 + 1, z^3 + 2\,z, z^4 + 3\,z^2 + 1, z^5 + 4\,z^3 + 3\,z \tag{14}$$