Prozeduren, Grundelemente der Programmierung

Prof. Dr. Dörte Haftendorn, Mathematik mit MuPAD 4.02, (ursprünglich 02 ex. in 3.11 Sept. 05)
Feb.07
http://haftendorn.uni-lueneburg.de **www.mathematik-verstehen.de**

##

Weiteres aus Hilfe, Suche mit den Stichworten
schleife, verzweigung, prozedur

Da kann man sehen, dass das in einer beliebigen (funktionalen) Computersprache Gelernte sofort umsetzbar ist.

```
f:=proc(n)
        begin n^2 end_proc:
```

```
f(5)
```

$$25$$

```
f'(n)
```

$$2 \cdot n$$

Man sieht, dass auch die üblichen Funktionen auch als Prozeduren geschrieben werden können.

```
ff:=n->n^2;
ff(5);
ff'(n);
```

$$n \to n^2$$

$$25$$

$$2 \cdot n$$

Herausgreifen aus einer Liste

```
liste:=[]:
for i in [2,4,7] do  liste:=liste.[f(i)] end_for;
```

$$[4, 16, 49]$$

Für diesen Zweck geht es viel einfacher mit dem "Folgenoperator $ "

```
f(i) $ i in [2,4,7];
[f(i) $ i in [2,4,7] ]
```

$$4, 16, 49$$

$$[4, 16, 49]$$

Rekursive Prozedur

```
g:=proc(n) begin
        if n=0 then 1
            else 2*g(n-1)
        end_if;
   end_proc:
```

```
g(n) $ n=1..10   //Erzeugung einer Liste
```

$$2, 4, 8, 16, 32, 64, 128, 256, 512, 1024$$

Prozedur mit einer repeat-Schleife


```
ida:=proc(i) begin
               repeat i:=i-1;print("Da sind noch ".i." Freunde");
               until i=0 end_repeat
             end_proc:
```

```
            end_proc:
ida(4)
"Da sind noch 3 Freunde"
"Da sind noch 2 Freunde"
"Da sind noch 1 Freunde"
"Da sind noch 0 Freunde"
```

Prozeduren mit mehreren Parametern

```
potenzTafelabisb:=proc(a,b,m,n)
                local i,k;
                begin
                  print("1. bis ".n.". Potenzen modulo ".m);
                  matrix([i $ i=1..n]),
                  matrix([[k^i mod m $ k=a..b] $ i=1..n]);
                end_proc
```

proc potenzTafelabisb(a, b, m, n) ... end

```
potenzTafelabisb(2,9,17,8)
"1. bis 8. Potenzen modulo 17"
```

$$\begin{pmatrix}1\\2\\3\\4\\5\\6\\7\\8\end{pmatrix}, \begin{pmatrix}2&3&4&5&6&7&8&9\\4&9&16&8&2&15&13&13\\8&10&13&6&12&3&2&15\\16&13&1&13&4&4&16&16\\15&5&4&14&7&11&9&8\\13&15&16&2&8&9&4&4\\9&11&13&10&14&12&15&2\\1&16&1&16&16&16&1&1\end{pmatrix}$$

```
potenzTafelabisb(2,5,6,4)
"1. bis 4. Potenzen modulo 6"
```

$$\begin{pmatrix}1\\2\\3\\4\end{pmatrix}, \begin{pmatrix}2&3&4&5\\4&3&4&1\\2&3&4&5\\4&3&4&1\end{pmatrix}$$

```
potenzTafelListe:=proc(liste,m,n)
                local i,k;
                begin
                  print(liste, " Potenzen modulo ".m);
                  matrix([i $ i=1..n]),
                  matrix([[k^i mod m $ k in liste] $ i=1..n]);
                end_proc:
potenzTafelListe([3,7,9],10,8)
[3, 7, 9], " Potenzen modulo 10"
```

$$\begin{pmatrix}1\\2\\3\\4\\5\\6\\7\\8\end{pmatrix}, \begin{pmatrix}3&7&9\\9&9&1\\7&3&9\\1&1&1\\3&7&9\\9&9&1\\7&3&9\\1&1&1\end{pmatrix}$$

```
liste:=[3,7,9];
k^1 mod 10 $ k in liste
```

[3, 7, 9]

3, 7, 9

```
 3, 7, 9
```

```
liste."das"
Error: Illegal argument [_concat]
```

##

siehe auch **ggt-prg.mn, die Entwicklung einer Prozedur für Euklidischen Algorithmus**.

Beispiele auch in **zahltheo-prog4.mn**