Logik

Logik, Wahrheitstafeln, logische Terme und Regeln .. Haftendorn 2011

Logische Terme braucht man vor allem bei Verzweigungen und Schleifen.

if Bedingung then oder While Bedingung

Der Gebrauch ist in der Lerndatei programmieren−einf−tns beschrieben.

Hier geht es also um die **Bedingungen**

if ant="ja" then könnte da z.B. stehen. Vorher muss dann eine Variable ant mit der Zeichenkette "ja" belegt sein.

Typisch ist auch **while eps>0.001** *Block* **endwhile** (in 3 Zeilen geschrieben)

Dann wird der Anweisungsblock solange ausgeführ, wie die Beingung erfüllt ist−

Oft braucht man aber eine Doppelbedungung der Form

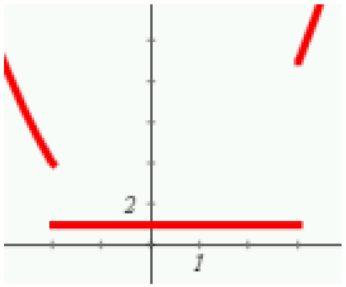
while eps>0.001 or n <20 *Block* endwhile (in 3 Zeilen geschrieben),

also einen logischen Term mit **or .** Damit wird abgesichert, dass der Schleifendurchlauf abgebrochen wird, obwohl eps aus irgendeinem Grund (z.B. Programmeirfehler) nicht klein wird. Dazu muss dann aber die Variable n wirklich bei jedem Durchlauf um 1 größer werden. Die übernächst nachfolgenden Seiten gehen auf die **Logik** als mathematisches Teilgebiet ein.

1.1

Einige Beispiele für den Logik Gebrauch

$$f(x):=\begin{cases} x^2, & x<-2 \text{ or } x>3 \\ 1, & -2\le x\le 3 \end{cases} \blacktriangleright \textit{Fertig}$$



$f(-3) \blacktriangleright 9$ $f(-2) \blacktriangleright 1$ $f(-2.0001) \blacktriangleright 4.0004$ $f(3) \blacktriangleright 1$ $f(3.0001) \blacktriangleright 9.0006$

Diese Funktion ist also außen eine Parabel und im Intervall [−2,3] eine waagerechte Gerade. (Übrigens diese Form für f ist bei den Vorlagen!!!!)

Anstelle von **or** darf oben natürlich nicht **and** stehen. Es gibt kein x, das sowohl links als auch rechts auf dem Zahlenstrahl liegt. Umgangssprachlich sagt man evt. "Die Parabel gilt für $x<-2$ und x>3. Darum muss man sich mit Logik befassen. Die nachfolgenden Seiten gehen auf die **Logik** als mathematisches Teilgebiet ein.

Gleichungen sind übligens **logische Aussagen**, wenn sie WAHR oder FALSCH sein können. $3=4 \blacktriangleright \text{false}$ $3=3 \blacktriangleright \text{true}$. Gleichungen die Variablen einhalten heißen **Aussageformen**, $x>3 \blacktriangleright x>3$ Sie werden erst beim Einsetzen Aussagen.

Logik, Wahrheitstafeln, logische Terme und Regeln .. Haftendorn 2011

Man kann einzel prüfen true or false ▸ true Gleichbedeutendmit 1 or 0 ▸ 1

Hier sind die üblichen Wahrheitstafeln nachgebildet.

```
Define wtt(a,b,c)=Func                                   ▸ Fertig
            Local ma
            ma:=newMat(3,3)
            ma[1,1]:=a: ma[1,2]:=b: ma[1,3]:=c
            ma[2,1]:=a and b:  ma[2,2]:=a and c:  ma[2,3]:=b and c
            ma[3,1]:=a or b:   ma[3,2]:=a or c:   ma[3,3]:=b or c
             Return ma
          EndFunc
```

$$\mathbf{wtt}(\text{true},\text{false},\text{false}) \blacktriangleright \begin{bmatrix} \text{true} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \\ \text{true} & \text{true} & \text{false} \end{bmatrix} \begin{bmatrix} a & b & c \\ a \text{ and } b & a \text{ and } c & b \text{ and } c \\ a \text{ or } b & a \text{ or } c & b \text{ or } c \end{bmatrix}$$

$$\mathbf{wtt}(\text{true},\text{true},\text{false}) \blacktriangleright \begin{bmatrix} \text{true} & \text{true} & \text{false} \\ \text{true} & \text{false} & \text{false} \\ \text{true} & \text{true} & \text{true} \end{bmatrix}$$

1.3

Logische Terme ..

$$\begin{bmatrix} a & b & c \\ a \text{ and } b & (a \text{ or } b) \text{ and } (a \text{ or } c) & a \text{ or } b \text{ and } c \\ a \text{ or } b & a \text{ and } (b \text{ or } c) & a \text{ and } b \text{ or } a \text{ and } c \end{bmatrix}$$

```
Define wt(a,b,c)=Func
    Local ma
    ma:=newMat(3,3)
    ma[1,1]:=a: ma[1,2]:=b: ma[1,3]:=c
    ma[2,1]:=a and b:   ma[2,2]:=(a or b) and (a or c):  ma[2,3]:=a or b and c
    ma[3,1]:=a or b:   ma[3,2]:=a and (b or c):   ma[3,3]:=a and b or a and c
    Return ma
EndFunc
```

▸ *Fertig*

wt(true,false,true) ▸ $\begin{bmatrix} \text{true} & \text{false} & \text{true} \\ \text{false} & \text{true} & \text{true} \\ \text{true} & \text{true} & \text{true} \end{bmatrix}$

Man kann hier und auf der folgenden Seite die Wahrheitswerte der Terme ablesen.

$$\mathbf{wt}(\text{true},\text{true},\text{true}) \rightarrow \begin{bmatrix} \text{true} & \text{true} & \text{true} \\ \text{true} & \text{true} & \text{true} \\ \text{true} & \text{true} & \text{true} \end{bmatrix} \qquad \mathbf{wt}(\text{true},\text{true},\text{false}) \rightarrow \begin{bmatrix} \text{true} & \text{true} & \text{false} \\ \text{true} & \text{true} & \text{true} \\ \text{true} & \text{true} & \text{true} \end{bmatrix}$$

$$\mathbf{wt}(\text{true},\text{false},\text{true}) \rightarrow \begin{bmatrix} \text{true} & \text{false} & \text{true} \\ \text{false} & \text{true} & \text{true} \\ \text{true} & \text{true} & \text{true} \end{bmatrix} \qquad \mathbf{wt}(\text{false},\text{true},\text{true}) \rightarrow \begin{bmatrix} \text{false} & \text{true} & \text{true} \\ \text{false} & \text{true} & \text{true} \\ \text{true} & \text{false} & \text{false} \end{bmatrix}$$

$$\mathbf{wt}(\text{false},\text{true},\text{false}) \rightarrow \begin{bmatrix} \text{false} & \text{true} & \text{false} \\ \text{false} & \text{false} & \text{false} \\ \text{true} & \text{false} & \text{false} \end{bmatrix} \qquad \mathbf{wt}(\text{false},\text{false},\text{false}) \rightarrow \begin{bmatrix} \text{false} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \end{bmatrix}$$

$$\mathbf{wt}(\text{false},\text{false},\text{true}) \rightarrow \begin{bmatrix} \text{false} & \text{false} & \text{true} \\ \text{false} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \end{bmatrix} \qquad \mathbf{wt}(\text{true},\text{false},\text{false}) \rightarrow \begin{bmatrix} \text{true} & \text{false} & \text{false} \\ \text{false} & \text{true} & \text{true} \\ \text{true} & \text{false} & \text{false} \end{bmatrix}$$

$$\mathbf{wt}(\text{false},\text{false},\text{false}) \rightarrow \begin{bmatrix} \text{false} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \\ \text{false} & \text{false} & \text{false} \end{bmatrix}$$

1.5

Hier ist noch **xor** mit eingebaut

$$\begin{bmatrix} a & b & c \\ a \text{ and } b & (a \text{ or } b) \text{ and } (a \text{ or } c) & a \text{ or } b \text{ and } c \\ a \text{ or } b & a \text{ and } (b \text{ or } c) & a \text{ and } b \text{ or } a \text{ and } c \end{bmatrix}$$

$$\begin{bmatrix} a \text{ xor } b & a \text{ and } (b \text{ xor } c) & a \text{ and } b \text{ xor } a \text{ and } c \end{bmatrix}$$

wie(3=4,3=3,3=9) ▸
$$\begin{bmatrix} \text{false} & \text{true} & \text{false} \\ \text{false} & \text{false} & \text{false} \\ \text{true} & \text{false} & \text{false} \\ \text{true} & \text{false} & \text{false} \end{bmatrix}$$

wie(3=4,3=3,3=3) ▸
$$\begin{bmatrix} \text{false} & \text{true} & \text{true} \\ \text{false} & \text{true} & \text{true} \\ \text{true} & \text{false} & \text{false} \\ \text{true} & \text{false} & \text{false} \end{bmatrix}$$

```
Define wie(a,b,c)=Func
                    Local ma
                    ma:=newMat(4,3)
                    ma[1,1]:=a: ma[1,2]:=b: ma[1,3]:=c
                     ma[2,1]:=a and b:   ma[2,2]:=(a or b) and (a or c):  ma[2,3]:=a or b and c
                      ma[3,1]:=a or b:   ma[3,2]:=a and (b or c):    ma[3,3]:=a and b or a and c
                      ma[4,1]:=a xor b:
                         ma[4,2]:=a and (b xor c)
                          ma[4,3]:=a and b xor a and c
                       Return ma
                   EndFunc
```

1.6

De Morgansche Regel ..

nicht (a und b) vgl nicht a und nicht b vgl nicht a oder nicht b vgl nicht (a oder b)

© *Also gilt* [not (*a* and *b*) =,not *a* or not *b*] *und* [not (*a* or *b*)=not *a* and not *b*] *Gesetze von de* **Morgan**

a,b,not (a and b)=not a or not b, not (a or b)=not a and not b

morgan(true,false) ▸ {true,false,true,"=",true,false,"=",false}

morgan(true,true) ▸ {true,true,false,"=",false,false,"=",false}

morgan(false,false) ▸ {false,false,true,"=",true,true,"=",true}

© *Implikation* **A,B,¬AvB** | **aus A folgt B**

folgt(false,true) ▸ {false,true,true} **folgt**(true,false) ▸ {true,false,false}

folgt(false,false) ▸ {false,false,true} **folgt**(true,true) ▸ {true,true,true}

Äquivalenz **A,B, A&B v ¬A &¬B**

aeq(true,false) ▸ {true,false,false} **aeq**(true,true) ▸ {true,true,true}

aeq(false,false) ▸ {false,false,true} **aeq**(false,true) ▸ {false,true,false}

1.7

```
Define LibPub morgan(a,                                          ▸ Fertig
b)=Func
    © Aufruf morgan(true,false)   zum Beispiel
    Disp "a,b,not (a and b)=not a or not b,   not (a or b)=not a and not b"
    {a,b,not (a and b),"=",not a or not b,not (a or b),"=",not a and not b}
    EndFunc


Define LibPub folgt(a,b)=Func                    ▸ Fertig
                         Disp "A,B,¬AvB"
                         {a,b,not a or b}
                         EndFunc
Define LibPub aeq(a,b)=Func                                      ▸ Fertig
                        Disp "A&B   v   ¬A &¬B"
                       {a,b,a and b or not a and not b}
                       EndFunc
```

1.8