Funktionen

COLLABORATORS			
	TITLE : Funktionen		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 7, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Funktionen

1.1 TurboCalc by Michael Friedrich

TurboCalc - copyright Michael Friedrich.

Formeln bestehen neben Operatoren und normalen Werten hauptsächlich aus Funktionen, die auf den folgenden Seiten nach Gebieten getrennt beschrieben werden.

Hier zunächst noch eine Übersicht über alle gängigen "Werte", wie sie von den Funktionen als Eingabeparameter (oder auch als Ausgabewert) benötigt werden.

Zahlen: Dies sind Zahlen im üblichen Format (+/-123[.123][E+-1]). Eine nähere Beschreibung finden Sie im Abschnitt "Eingabe".

Wahrheitswerte: Dies sind WAHR oder FALSCH. Statt normalen Wahrheitswerten können auch Zahlen benutzt werden. (0 bedeutet dann FALSCH, alles andere WAHR).

Texte: Diese müssen in Anführungszeichen stehen und können beliebige Zeichen enthalten. Liefert eine Formel als Ergebnis einen Text, so wird dieser natürlich ohne Anführungszeichen angezeigt.

Tip: Soll in einem Text das Anführungszeichen benutzt werden, so müssen für jedes Anführungszeichen zwei aufeinanderfolgende Anführungszeichen geschrieben werden ("a""b" ist der Text a"b)

Datum: Dies ist eine fortlaufende Ganzzahl (0 bedeutet 1.1.1900,1: 2.1.1900...) die jedoch nicht als Zahl sondern als richtiges Datum eingegeben werden sollte. Hierzu muß jedoch, im Gegensatz zur Eingabe eines Datums in eine Zelle, die Funktion DATWERT("Text") oder einfach WERT("Text") benutzt werden. Beispiel: WERT("1-9-93").

Uhrzeit: Dies ist eine fortlaufende Ganzzahl (in Sekundenschritten) (0 bedeutet 0 Uhr, 1: 0:00:01..) die jedoch nicht als Zahl sondern als richtige Uhrzeit eingegeben werden sollte. Hierzu muß jedoch, im Gegensatz zur Eingabe in eine Zelle, die Funktion ZEITWERT("Text") oder einfach WERT("Text") benutzt werden. Beispiel: WERT("12:03:07").

Bezug: Hiermit kann man Inhalte von Zellen auslesen und den Inhalt als Wert benutzen. Bezüge bestehen aus einem (oder zwei) Buchstaben gefolgt von einer Zahl (ohne Leerzeichen!), etwa A1 oder CC23. Die Buchstaben geben dabei die Spalte und die Zahl die Zeile an. Steht in der Zelle C7 etwa 123, so ergibt "=C7" die Zahl 123.

Bezüge können auch absolut angegeben werden (d.h sie werden beim Kopieren, Verschieben,... nicht verändert), indem man vor die jeweilige "feste" Richtung (oder beide) ein Dollarzeichen stellt, etwa C7 oder $C7 oder C$7.

Möchte man Werte aus anderen Tabellen auslesen, so kann man dazu die Funktion @ (oder AT) benutzen, siehe bei Zell-Funktionen.

Bereich: Dies sind Parameter für die Bereichs- und die Datenbankfunktionen. Bereiche bestehen aus zwei durch Doppelpunkt getrennten Bezügen (etwa A1:C5 oder $C3: $E5) und bestimmen das Rechteck zwischen diesen beiden Zellen. (Ein Bereich kann auch aus nur einem Bezug bestehen, dann ist damit der 1 mal 1 große Bereich des Bezugs gemeint).

Namen: Für alle obigen Typen können auch Namen (d.h. Variablen) benutzt werden. Einzelheiten dazu siehe im Abschnitt "Namen".

Tip: In der Anleitung finden Sie die Beschreibung zu den Funktionen unter dem deutschen Stichwort. Suchen Sie nun eine "englische" Funktion, so können Sie in der Übersicht "Deutsch-Englisch" im Anhang schnell den passenden deutschen Funktionsnamen suchen.

1.2 Operatoren

Operatoren

Operatoren stehen, im Gegensatz zu den unten aufgeführten Funktionen, zwischen zwei Werten und bestimmen, wie diese verknüpft werden sollen.

Hier eine Übersicht über alle möglichen Operatoren.

Pri. Operator Beschreibung

5 ^ (hoch) potenziert

4 * multipliziert

4 / dividiert

4 MOD Berechnet den Modulo, siehe bei der Mathematikfunktion REST

REST

3 + addiert (auch Texte, Uhrzeiten und Daten, siehe bei den entsprechenden Funktionsbeschreibungen)

3 - subtrahiert (auch Texte,... siehe bei "+")

2 = Vergleicht die beiden Werte und gibt entweder WAHR oder FALSCH zurück

> (auch Texte, Uhrzeiten und Daten)

<

<>

>=

<=

1 UND Verknüpft die Werte mit logischem UND. Siehe bei Wahrheitswertfunktionen

AND unter UND.

1 ODER Verknüpft die Werte mit logischem ODER. Siehe bei Wahrheitswertfunktionen

OR unter ODER.

1 XOR Verknüpft die Werte mit logischen XOR. Siehe bei Wahrheitswertfunktionen unter XOR.

Die Priorität (Pri.) gibt dabei an, in welcher Reihenfolge die Berechnung stattfinden soll, falls keine Klammern gesetzt sind. Dann werden zuerst die Operationen mit der höchsten Priorität ausgeführt, danach die anderen. Haben Operatoren die gleiche Priorität, so wird von links nach rechts gerechnet. Dies entspricht der Standard-Regel "Punkt vor Strich".

Beispiele:

2+2*3^2+2 wird als (2+(2* (3^2))) + 2 berechnet.

2-2-2 wird als (2-2)-2 berechnet (bei gleicher Priorität von links nach rechts) und ergibt -2, obwohl natürlich auch 2-(2-2) = 2 möglich wäre.

1.3 Mathematikfunktionen

Mathematikfunktionen

Dies sind die normalen mathematischen Funktionen. Die meisten davon werden Sie auf Taschenrechnern oder in Programmiersprachen finden.

Zahl: Falls nicht anders bezeichnet, handelt es sich beim Eingabeparameter "Zahl" um eine normale reelle Zahl.

ABS(Zahl)

ARCCOS(Zahl)

ARCSIN(Zahl)

ARCTAN(Zahl)

BOGEN(Zahl)

COS(Zahl)

COSHYP(Zahl)

EXP(Zahl)

FAKULTÄT(Zahl)

GANZZAHL(Zahl)

LG(Zahl)

LN(Zahl)

LOG(Zahl)

LOG10(Zahl)

PI()

QUADRAT(Zahl)

REST(Zahl1;Zahl2)

RUNDEN(Zahl;Stellen)

SIN(Zahl)

SINHYP(Zahl)

TAN(Zahl)

TANHYP(Zahl)

VORZEICHEN(Zahl)

WINKEL(Zahl)

WURZEL(Zahl)

ZUFALLSZAHL()

1.4 ABS(Zahl)

ABS(Zahl)

Die Funktion ABS liefert als Ergebnis den Absolutwert (d.h. den Betrag) einer Zahl, d.h. die Zahl ohne Vorzeichen.

Beispiel:

ABS(2) ergibt 2

ABS(-2) ergibt 2

Verwandte Funktionen:

Vorzeichen

1.5 ARCCOS(Zahl)

ARCCOS(Zahl)

Die Funktion ARCCOS liefert den Arcuscosinus einer Zahl zurück. Dies ist der Winkel dessen Kosinus (COS) Zahl ergibt. Der Ergebniswinkel wird im Bogenmaß im Wertebereich 0 bis PI angegeben. (Zur Umrechnung ins Gradmaß: siehe WINKEL)

Zahl ist der Kosinus des gesuchten Winkels und muß zwischen -1 und 1 liegen (sonst wird der Fehler #WERT ausgegeben).

Beispiel:

ARCCOS(-0.5) ergibt 2.094

WINKEL(ARCCOS(-0.5)) ergibt 120 (Grad)

Verwandte Funktionen:

COS , PI , WINKEL

1.6 ARCSIN(Zahl)

ARCSIN(Zahl)

Die Funktion ARCSIN liefert den Arcussinus einer Zahl zurück. Dies ist der Winkel dessen Sinus (SIN) Zahl ergibt. Der Ergebniswinkel wird im Bogenmaß im Wertebereich -PI/2 bis PI/2 angegeben. (Zur Umrechnung ins Gradmaß: siehe WINKEL)

Zahl ist der Sinus des gesuchten Winkels und muß zwischen -1 und 1 liegen (sonst wird der Fehler #WERT ausgegeben).

Beispiel:

ARCSIN(-0.5) ergibt 0.524

WINKEL(ARCSIN(-0.5)) ergibt -30 (Grad)

Verwandte Funktionen:

SIN , PI , WINKEL

1.7 ARCTAN(Zahl)

ARCTAN(Zahl)

Die Funktion ARCTAN liefert den Arkustangens einer Zahl zurück. Dies ist der Winkel dessen Tangens (TAN) Zahl ergibt. Der Ergebniswinkel wird im Bogenmaß im Wertebereich -PI/2 bis PI/2 angegeben. (Zur Umrechnung ins Gradmaß: siehe WINKEL)

Beispiel:

ARCTAN(1) ergibt 0.785

WINKEL(ARCTAN(-1)) ergibt 45 (Grad)

Verwandte Funktionen:

TAN , PI , WINKEL

1.8 BOGEN(Zahl)

BOGEN(Zahl)

(Englisch: DEGTORAD)

Die Funktion BOGEN wandelt einen Winkel vom Gradmaß in das Bogenmaß um.

Damit können normale Winkelangaben als Parameter für die Funktionen Sinus, Cosinus und Tangens benutzt werden.

Beispiel:

BOGEN(30) = 0.523 (= PI/6)

SIN(BOGEN(30)) = 0.5

Verwandte Funktionen:

WINKEL , SIN , COS , TAN

1.9 COS(Zahl)

COS(Zahl)

Die Funktion COS liefert den Kosinus des Winkels Zahl (im Bogenmaß). Zur Umrechnung von Winkeln ins Bogenmaß siehe BOGEN().

Beispiel:

COS(1.047) ergibt 0.5

COS(BOGEN(60)) ergibt 0.5

Verwandte Funktionen:

ARCCOS , PI , BOGEN

1.10 COSHYP(Zahl)

COSHYP(Zahl)

Die Funktion COSHYP liefert den Cosinus hyperbolicus von Zahl (d.h. =1/2*(e^z+e^-z))

Beispiel:

COSHYP(2) ergibt 3.762

Verwandte Funktionen:

EXP , SINHYP , TANHYP

1.11 EXP(Zahl)

EXP(Zahl)

Die Funktion EXP liefert als Ergebnis den Wert e (=2.718281828) hoch Zahl.

Beispiel:

EXP(1) ergibt 2.71828...

EXP(LN(5)) ergibt 5

Verwandte Funktionen:

LN , LOG , LOG10 , LG

1.12 FAKULTÄT(Zahl)

FAKULTÄT(Zahl)

(Englisch: FAC)

Die Funktion FAKULTÄT liefert als Ergebnis die Fakultät einer Zahl, d.h. 1*2*3*...*Zahl

Zahl sollte daher positiv und ganz sein. (Kommazahlen werden abgeschnitten)

Hinweis: Statt FAKULTÄT(3) kann auch einfach 3! geschrieben werden.

Beispiel:

FAKULTÄT(5) ergibt 120

FAKULTÄT(5.2) ergibt 120

5! ergibt auch 120

1.13 GANZZAHL(Zahl)

GANZZAHL(Zahl)

(Englisch: INTEGER, oder auch kurz GANZ bzw. INT)

Die Funktion GANZZAHL rundet Zahl auf die nächste ganze Zahl ab.

Beispiel:

GANZZAHL(5.4) ergibt 5

GANZZAHL(5.7) ergibt 5

GANZZAHL (-5.2) ergibt -6

Verwandte Funktionen:

REST , RUNDEN

1.14 LG(Zahl)

LG(Zahl)

Synonym zu LOG10 (siehe unten) welches bei Mathematikern sehr geläufig ist.

1.15 LN(Zahl)

LN(Zahl)

Die Funktion LN liefert den natürlichen Logarithmus einer Zahl (d.h. zur Basis e=2.718281828...)

Zahl ist eine positive reelle Zahl (d.h. beliebige Zahl >0). Ansonsten erscheint der Fehlerwert #WERT.

LN ist die Umkehrfunktion zu EXP

Beispiel:

LN(100) = 4.605

LN(EXP(3) = 3

Verwandte Funktionen:

EXP , LOG , LOG10

1.16 LOG(Zahl)

LOG(Zahl)

Synonym zu LN (siehe oben) welches bei Mathematikern sehr geläufig ist.

1.17 LOG10(Zahl)

LOG10(Zahl)

Die Funktion LOG10 liefert den Logarithmus einer Zahl zur Basis 10

Zahl ist eine positive reelle Zahl.

Beispiel:

LN(100) = 2

LN(10^5) =5

Verwandte Funktionen:

EXP , LN , LG , LOG , LOG10

1.18 PI()

PI()

Die Funktion PI liefert die Zahl 3.1415926...

Beispiel:

PI()/2 = 1.5707...

PI/2 = 1.5707...

SIN(PI/2) = 1

Dies wird zur Berechnung des Umfangs und der Fläche von Kreisen/Kugeln benutzt. Steht in A1 der Radius eines Kreises, so ergibt:

2*PI*A1 den Umfang und

PI*(A1^2) die Fläche des Kreises.

1.19 QUADRAT(Zahl)

QUADRAT(Zahl)

(Englisch: SQR)

Die Funktion QUADRAT liefert das Quadrat von Zahl, also Zahl*Zahl.

Beispiel:

QUADRAT(5) = 25

5^2=25

Verwandte Funktionen:

WURZEL

1.20 REST(Zahl1;Zahl2)

REST(Zahl1;Zahl2)

(Englisch: MOD)

Die Funktion REST bestimmt den Rest bei der Division von Zahl1 durch Zahl2.

Zahl1 und Zahl2 können dabei beliebige reelle Zahlen sein (Zahl2 ungleich 0!)

Beispiel:

REST(10;3) ergibt 1

REST(10.5;3) ergibt 1.5

REST(2.25;0.5) ergibt 0,.25

1.21 RUNDEN(Zahl;Stellen)

RUNDEN(Zahl;Stellen)

(Englisch: ROUND)

Die Funktion RUNDEN rundet eine Zahl auf eine bestimmte Dezimal-Stelle.

Zahl ist die zu rundende reelle Zahl.

Stellen bestimmt, auf wieviele Dezimalstellen gerundet werden soll.

Stellen>0: rundet auf "Stellen" Stellen nach dem Komma.

Stellen<0: rundet auf die entsprechenden Stellen links vom Komma (also 10,100,1000...)

Stellen=0: rundet auf die ganze Zahl

Beispiel:

RUNDEN(3.14;1) = 3.1

RUNDEN(3.15;1) = 3.2

RUNDEN(314.5;-2) = 300

1.22 SIN(Zahl)

SIN(Zahl)

Die Funktion SIN liefert den Sinus des Winkels Zahl (im Bogenmaß). Zur Umrechung von Winkeln ins Bogenmaß siehe BOGEN().

Beispiel:

SIN(1.047) ergibt 0.8659

SIN(BOGEN(60)) ergibt 0.8659

Verwandte Funktionen:

ARCSIN , PI , BOGEN

1.23 SINHYP(Zahl)

SINHYP(Zahl)

Die Funktion SINHYP liefert den Sinus hyperbolicus von Zahl (d.h. =1/2*(eˆz-eˆ-z))

Beispiel:

SINHYP(2) ergibt 3.627

Verwandte Funktionen:

EXP , COSHYP , TANHYP

1.24 TAN(Zahl)

TAN(Zahl)

Die Funktion TAN liefert den Tangens des Winkels Zahl (im Bogenmaß) (d.h. SIN/COS). Zur Umrechung von Winkeln ins Bogenmaß siehe BOGEN().

Beispiel:

TAN(0.7854) ergibt 1

TAN(BOGEN(45)) ergibt 1

Verwandte Funktionen:

ARCTAN , COS , PI , BOGEN , SIN

1.25 TANHYP(Zahl)

TANHYP(Zahl)

Die Funktion TANHYP liefert den Tangens hyperbolicus von Zahl (d.h. =SINHYP(Zahl)/COSHYP(Zahl))

Beispiel:

TANHYP(2) ergibt 0.964

Verwandte Funktionen:

EXP , SINHYP , COSHYP

1.26 VORZEICHEN(Zahl)

VORZEICHEN(Zahl)

(Englisch: SIGN)

Die Funktion Vorzeichen liefert als Ergebnis -1,0,1 in Abhängigkeit vom Vorzeichen der Zahl:

-1 wenn Zahl<0, 0 falls Zahl=0 und 1 falls Zahl>0.

Beispiel:

VORZEICHEN(20) = 1

VORZEICHEN(-20) = -1

VORZEICHEN(0) = 0

VORZEICHEN(A1)*ABS(A1) liefert gerade A1 zurück (falls A1 eine Zahl enthält)

Verwandte Funktionen:

ABS

1.27 WINKEL(Zahl)

WINKEL(Zahl)

(Englisch: RADTODEG)

Die Funktion WINKEL wandelt einen Winkel vom Bogenmaß in das Gradmaß um.

Damit können Winkelangaben der Funktionen ARCSIN, ARCCOS und ARCTAN in normale Winkel umgerechnet werden.

Beispiel:

WINKEL(0.5236) = 30

WINKEL(ARCSIN(0.5)) = 30

Verwandte Funktionen:

BOGEN , ARCSIN , ARCCOS , ARCTAN

1.28 WURZEL(Zahl)

WURZEL(Zahl)

(Englisch: SQRT)

Die Funktion WURZEL liefert die positive Quadratwurzel einer Zahl.

Zahl ist ein reelle positive Zahl

Beispiel:

WURZEL(25) = 5

WURZEL(QUADRAT(5)) = 5

WURZEL(-25) = #WERT

WURZEL(ABS(-25)) = 5

Verwandte Funktionen:

QUADRAT , ABS

1.29 ZUFALLSZAHL()

ZUFALLSZAHL()

(Englisch: RND)

Die Funktion ZUFALLSZAHL liefert eine Zufallszahl zwischen 0 und 1.

Will man eine Zufallszahl zwischen 1 und N, so kann dies wie folgt erreicht werden;

GANZZAHL(ZUFALLSZAHL()*N+1)

Beispiel:

GANZZAHL(ZUFALLSZAHL()*6+1) liefert eine zufällige Zahl zwischen 1 und 6, simuliert also einen Würfel.

1.30 Wahrheitswert-Funktionen

Wahrheitswert-Funktionen

Diese Funktionen liefern als Ergebnis einen Wahrheitswert (WAHR bzw FALSCH, d.h. 1 bzw. 0) zurück bzw. verarbeiten solche Wahrheitswerte.

Diese Funktionen eignen sich damit gut für bedingte Berechnungen und Entscheidungen. (Siehe vor allem WENN.)

FALSCH()

ISTDATUM(Wert)

ISTFEHLER(Bezug)

ISTLEER(Bezug)

ISTTEXT(Wert)

ISTZAHL(Wert)

ISTZEIT(Wert)

NICHT(Wert)

ODER(Wert1;Wert2;...)

UND(Wert1;Wert2;...)

WAHR()

WENN(Bedingung; Wert1; Wert2)

XOR(Wert1;Wert2;...)

1.31 FALSCH()

FALSCH()

(Englisch: FALSE)

Die Funktion FALSCH liefert den Wert FALSCH (=0) zurück. Sie kann entweder als FALSCH oder als FALSCH() geschrieben werden.

Beispiel:

WENN(FALSCH;1;2) ergibt 2

1.32 ISTDATUM(Wert)

ISTDATUM(Wert)

(Englisch: ISDATE)

Die Funktion ISTDATUM prüft, ob es sich bei Wert um ein Datum handelt. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Dies wird meist dazu benutzt, um zu prüfen, ob der Inhalt einer Zelle ein Datum ist.

Beispiel:

ISTDATUM(12) ergibt FALSCH

ISTDATUM(A1) ergibt WAHR, falls in A1 ein Datum steht.

1.33 ISTFEHLER(Bezug)

ISTFEHLER(Bezug)

(Englisch: ISERROR)

Die Funktion ISTFEHLER prüft, ob die Zelle Bezug eine Fehlermeldung enthält. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Beispiel:

ISTFEHLER(A1) mit A1 enthält "12" ergibt FALSCH

ISTFEHLER(A1) mit A1 enthält "=2+" ergibt WAHR

1.34 ISTLEER(Bezug)

ISTLEER(Bezug)

(Englisch: ISEMPTY)

Die Funktion ISTLEER prüft, ob die Zelle, auf die Bezug zeigt, leer ist. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Bezug: Ist ein Bezug auf eine Zelle, etwa A1

Beispiel:

ISTLEER(A1) ergibt FALSCH, falls dort etwa eine Zahl, ein Text oder etwas anderes steht.

1.35 ISTTEXT(Wert)

ISTTEXT(Wert)

(Englisch: ISSTRING)

Die Funktion ISTTEXT prüft, ob es sich bei Wert um einen Text handelt. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Beispiel:

ISTTEXT("12") ergibt WAHR

ISTTEXT(A1) ergibt WAHR, falls in A1 ein Text steht.

1.36 ISTZAHL(Wert)

ISTZAHL(Wert)

(Englisch: ISVALUE)

Die Funktion ISTZAHL prüft, ob es sich bei Wert um eine Zahl handelt. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Beispiel:

ISTZAHL("12") ergibt FALSCH

ISTZAHL(A1) ergibt WAHR, falls in A1 eine Zahl steht.

1.37 ISTZEIT(Wert)

ISTZEIT(Wert)

(Englisch: ISTIME)

Die Funktion ISTZEIT prüft, ob es sich bei Wert um eine Zeit handelt. Ist dies der Fall, so wird WAHR zurückgegeben, ansonsten FALSCH.

Beispiel:

ISTZEIT(12) ergibt FALSCH

ISTZEIT(A1) ergibt WAHR, falls in A1 eine Zeit steht.

1.38 NICHT(Wert)

NICHT(Wert)

(Englisch: NOT)

Die Funktion NICHT negiert den Wahrheitswert, d.h. ist Wert WAHR, dann wird FALSCH zurückgegeben und umgekehrt.

Wert: Kann ein Wahrheitswert oder eine Zahl sein. 0 stellt dann FALSCH dar, alle anderen Zahlen ergeben WAHR.

Beispiel:

NICHT(WAHR) ergibt FALSCH

NICHT(FALSCH) ergibt WAHR

NICHT(3) ergibt FALSCH

NICHT(WAHR UND FALSCH) ergibt WAHR

Verwandte Funktionen:

UND , ODER , XOR

1.39 ODER(Wert1;Wert2;...)

ODER(Wert1;Wert2;...)

(Englisch: OR)

Die Funktion ODER gibt die Oder-Verknüpfung aller Werte als Wahrheitswert zurück, d.h. es wird WAHR zurückgegeben, falls mindestens einer der Werte WAHR oder nicht Null ist. Sind alle Werte Null oder FALSCH, so wird FALSCH zurückgegeben.

Werte: Dies können beliebig viele, durch Semikolon getrennte Werte sein, und zwar: Wahrheitswerte oder Zahlen. (eine Zahl gilt dann als Wahr, falls sie nicht 0 ist)

Hinweis: Statt dieser Funktionsschreibweise kann ODER (oder OR) auch einfach als Operator (wie etwa "+" benutzt werden), also: A1 ODER A3. Dies ist vor allem dann sinnvoller und übersichtlicher, wenn nur zwei Werte Oder-Verknüpft werden sollen. (Weiterhin kann in dieser Schreibweise OR auch als Oder-Verknüpfung zweier Zahlen betrachtet werden: 3 OR 4 ergibt dann 7)

Beispiel:

ODER(WAHR;FALSCH) ergibt WAHR

ODER(FALSCH;0;0) ergibt FALSCH

ODER(FALSCH;0;1) ergibt WAHR

WENN(ODER(0;1);3;4) ergibt 4.

dies entspricht auch:

WAHR ODER FALSCH,

FALSCH ODER 0 ODER 0

FALSCH ODER 0 ODER 1

WENN(0 ODER 3;3;4)

2 OR 1 ergibt 3 und 3 OR 2 ergibt 3

Verwandte Funktionen:

UND , NICHT , XOR

1.40 UND(Wert1;Wert2;...)

UND(Wert1;Wert2;...)

(Englisch: AND)

Die Funktion UND gibt die Und-Verknüpfung aller Werte als Wahrheitswert zurück, d.h. es wird WAHR zurückgegeben, falls alle Werte WAHR oder nicht Null sind. Ist auch nur ein Wert FALSCH oder Null, so wird FALSCH zurückgegeben..

Werte: Dies können beliebig viele, durch Semikolon getrennte Werte sein, und zwar: Wahrheitswerte oder Zahlen. (eine Zahl gilt dann als Wahr, falls sie nicht 0 ist)

Hinweis: Statt dieser Funktionsschreibweise kann UND (oder AND) auch einfach als Operator (wie etwa "+" benutzt werden), also: A1 UND A3. Dies ist vor allem dann sinnvoller und übersichtlicher, wenn nur zwei Werte Und-Verknüpft werden sollen. (Weiterhin kann in dieser Schreibweise UND auch als Und-Verknüpfung zweier Zahlen betrachtet werden: 7 UND 4 ergibt dann 4)

Beispiel:

UND(WAHR;FALSCH) ergibt FALSCH

UND(WAHR;1;2) ergibt WAHR

UND(FALSCH;WAHR;1) ergibt FALSCH

WENN(UND(0;1);3;4) ergibt 4.

dies entspricht auch:

WAHR UND FALSCH,

WAHR UND 1 UND 2

FALSCH UND WAHR UND 1

WENN(0 UND 1;3;4)

3 UND 1 ergibt 1

7 UND 3 ergibt 3

Verwandte Funktionen:

ODER , NICHT , XOR

1.41 WAHR()

WAHR()

(Englisch: TRUE)

Die Funktion WAHR liefert den Wert WAHR (=1) zurück. Sie kann entweder als WAHR oder als WAHR() geschrieben werden.

Beispiel:

WENN(WAHR;1;2) ergibt 1

1.42 WENN(Bedingung; Wert1; Wert2)

WENN(Bedingung; Wert1; Wert2)

(Englisch: IF)

Die Funktion WENN gibt in Abhängigkeit von Bedingung entweder Wert1 (falls Bedingung erfüllt, WAHR) oder aber Wert2 zurück.

Bedingung kann dabei ein beliebiger Boolscher Ausdruck sein, d.h. Vergleich ggf. mit UND oder ODER verknüpft.

Wert1, Wert2 können beliebige Ausdrücke (Texte, Zahlen...) sein. (Auch weitere WENN(..;..;..) als Parameter sind möglich - WENN läßt sich also schachteln).

Tip: Diese Funktion ist sehr praktisch, da damit auf einfache Weise Fallunterscheidungen gemacht werden können.

Beispiel:

WENN(WAHR;"Hallo";"Du") ergibt "Hallo"

WENN(1=3;"Hallo";"Du") ergibt "Du"

WENN(A1>0;"Gewinn";"Verlust") bestimmt in Abhängigkeit von A1, ob Gewinn oder Verlust gemacht wurde.

WENN(A1<10;25;25+(A1-10)*0.23) berechnet die Telefonkosten bei A1 Einheiten (mit Grundgebühr: 25 DM, Einheit 23 Pf und 10 Freieinheiten)

Verwandte Funktionen:

AUSWAHL

1.43 XOR(Wert1;Wert2;...)

XOR(Wert1;Wert2;...)

Die Funktion XOR gibt die Exklusiv-Oder-Verknüpfung aller Werte als Wahrheitswert zurück, d.h. es wird FALSCH zurückgegeben, falls eine gerade Anzahl von Werten WAHR oder nicht Null ist. Sonst wird WAHR zurückgegeben.

Wert1 Wert2 Wert1 XOR Wert2

FALSCH FALSCH FALSCH

FALSCH WAHR WAHR

WAHR FALSCH WAHR

WAHR WAHR FALSCH

Werte: Dies können beliebig viele, durch Semikolon getrennte Werte sein, und zwar: Wahrheitswerte oder Zahlen. (eine Zahl gilt dann als WAHR, falls sie nicht 0 ist)

Hinweis: Statt dieser Funktionsschreibweise kann XOR auch einfach als Operator (wie etwa "+") benutzt werden, also: A1 XOR A3. Dies ist vor allem dann sinnvoller und übersichtlicher, wenn nur zwei Werte Exklusiv-Oder-Verknüpft werden sollen. (Weiterhin kann in dieser Schreibweise XOR auch als Exklusiv-Oder-Verknüpfung zweier Zahlen betrachtet werden: 3 XOR 4 ergibt dann 7)

Beispiel:

XOR(WAHR;FALSCH) ergibt WAHR

XOR(FALSCH;0;0) ergibt FALSCH

XOR(FALSCH;0;1) ergibt WAHR

WENN(XOR(0;1);3;4) ergibt 3.

dies entspricht auch:

WAHR XOR FALSCH,

FALSCH XOR 0 XOR 0

FALSCH XOR 0 XOR 1

WENN(0 XOR 3;3;4)

2 XOR 1 ergibt 3

3 XOR 2 ergibt 1

Verwandte Funktionen:

ODER ; UND , NICHT

1.44 Text-Funktionen

Text-Funktionen

Plus: +

CODE(Text)

GLÄTTEN(Text)

GROSS(Text)

GROSS2(Text)

HEX(Zahltext)

INTEXT(String;Muster[;Pos])

KLEIN(Text)

KOMPRIMIEREN(String[;Liste])

LÄNGE(Text)

LINKS(Text;Anzahl)

MITTE(Text;Zahl1;Zahl2)

RECHTS(Text;Anzahl)

SÄUBERN(Text)

SCHIEBENL(Text)

SCHIEBENR(Text)

SPIEGELN(Text)

TEIL(Text;Zahl1;Zahl2)

TEXT(Data[;Format])

WERT(Text)

WIEDERHOLEN(Text; Anzahl)

ZEICHEN(asc-code)

1.45 Plus: +

Plus: +

Der Operator "+" kann zum Verketten (Hintereinanderschreiben) von Texten benutzt werden. Er wird dabei genauso wie bei normalen Zahlen benutzt:

Hinweis: Eine Minusoperation sowie eine Operation zwischen einem Text und einer Zahl ist dabei nicht definiert, es erscheint dann #TYP.

Beispiel:

"Hallo" +" "+"Welt" ergibt "Hallo Welt"

"Ab"+"C" ergibt "AbC"

1.46 CODE(Text)

CODE(Text)

Die Funktion Code liefert den ASCII-Code (American Standard Code for Information Interchange - siehe Handbuch zum Amiga) des ersten Zeichens des Texts.

Beispiel:

CODE("A") ergibt 65

CODE("ABC") ergibt 65

CODE("a") ergibt 97 (da klein geschrieben!)

Verwandte Funktionen:

ZEICHEN

1.47 GLÄTTEN(Text)

GLÄTTEN(Text)

(Englisch: TRIM)

Die Funktion GLÄTTEN löscht aus Text alle "unnötigen" Leerzeichen, d.h. es läßt nur noch jeweils ein Leerzeichen übrig.

Beispiel:

GLÄTTEN("Januar Februar März") ergibt "Januar Februar März"

Verwandte Funktionen:

SÄUBERN

1.48 GROSS(Text)

GROSS(Text)

(Englisch: UPPER)

Die Funktion GROSS wandelt alle Buchstaben des Texts in Großbuchstaben um.

Beispiel:

GROSS("Dies ist ein Text") liefert "DIES IST EIN TEXT"

Verwandte Funktionen:

GROSS2 , KLEIN

1.49 GROSS2(Text)

GROSS2(Text)

(Englisch: UPPER2)

Die Funktion GROSS2 wandelt die ersten Buchstaben jedes Wortes in Groß- und alle anderen in Kleinbuchstaben um.

Beispiel:

GROSS2("Dies ist ein Text") liefert "Dies Ist Ein Text"

Verwandte Funktionen:

GROSS , KLEIN

1.50 HEX(Zahltext)

HEX(Zahltext)

Diese Funktion wandelt den Text ’Zahltext’ aus dem Hexadezimalsystem

in eine ’normale’ Zahl im Dezimalsystem um.

Zahltext ist ein Text der aus den Ziffern ’0’-’9’ und den Zeichen ’A’-’F’ bestehen sollte (Kleinschreibung natürlich möglich). Jedes andere Zeichen beendet die Umwandlung

Beispiel:

HEX("C8") ergibt 200

HEX("C8X1") ergibt 200, die Umwandlung wird beim ’X’ abgebrochen.

1.51 INTEXT(String;Muster[;Pos])

INTEXT(String;Muster[;Pos])

(Englisch: INSTRING)

Gibt die erste Position (ab der Stelle Pos) zurück, ab der Muster in String zu finden ist. Konnte der Text nicht gefunden werden, so wird 0 zurückgegeben.

String: zu durchsuchender Text

Muster: Text, der in String gesucht werden soll.

Pos: Die Position, ab der die Suche begonnen werden soll (1=erste Position - wird dieser Parameter ausgelassen, so wird ab dem Beginn durchsucht)

Beispiel:

INTEXT("ein Beispieltext";"ei") ergibt 1

INTEXT("ein Beispieltext";"ei";2) ergibt 6

INSTRING("ein Beispieltext";"abc";) ergibt 0 (d.h.nicht gefunden)

1.52 KLEIN(Text)

KLEIN(Text)

(Englisch: LOWER)

Die Funktion KLEIN wandelt alle Buchstaben des Textes in Kleinbuchstaben um.

Beispiel:

KLEIN("Dies ist ein Text") liefert "dies ist ein text"

Verwandte Funktionen:

GROSS , GROSS2

1.53 KOMPRIMIEREN(String[;Liste])

KOMPRIMIEREN(String[;Liste])

(Englisch: COMPRESS)

Entfernt alle Vorkommen der Zeichen aus Liste aus String und gibt diesen zurück. (Wird Liste ausgelassen, so werden alle Leerzeichen entfernt).

String: Text, der komprimiert werden soll.

Liste: Text mit allen Zeichen, die entfernt werden sollen. Wird dieser Parameter ausgelassen, so wird " " (Leerzeichen) angenommen.

Beispiel:

KOMPRIMIEREN(" a b c") ergibt "abc"

KOMPRIMIEREN("++ab++c";"+-") ergibt "abc"

1.54 LÄNGE(Text)

LÄNGE(Text)

(Englisch: LENGTH)

Die Funktion LÄNGE liefert die Anzahl der Zeichen des Textes als Zahl.

Beispiel:

LÄNGE("Michael Friedrich") ergibt 17

LÄNGE("") liefert 0

1.55 LINKS(Text;Anzahl)

LINKS(Text;Anzahl)

(Englisch: LEFT)

Die Funktion LINKS liefert die ersten Anzahl Zeichen des Textes (von links)

Anzahl muß dabei größer als 0 sein (sonst erscheint die Fehlermeldung #WERT). Ist Anzahl größer als die Länge des Textes, so wird der Text zurückgegeben.

Wird Anzahl ausgelassen (also LINKS(Text)), so wird das erste Zeichen des Textes zurückgegeben.

Beispiel:

LINKS("Michael Friedrich";4) ergibt "Mich"

LINKS("Hallo") ergibt "H"

Verwandte Funktionen:

MITTE , TEIL , RECHTS

1.56 MITTE(Text;Zahl1;Zahl2)

MITTE(Text;Zahl1;Zahl2)

(Englisch: MID)

Die Funktion MITTE liefert ab der Stelle Zahl1 einen höchstens Zahl2-langen Teiltext von Text.

Zahl1 muß größer als 0 sein (ansonsten wird der Fehler #WERT zurückgegeben) und bestimmt die Position, ab der der Text entnommen werden soll.

Zahl2 muß größer 0 sein (auch sonst: #WERT) und bestimmt, wieviele Zeichen entnommen werden sollen. Ist Zahl2 größer als die restliche Textlänge, so wird dieser Resttext zurückgegeben.

Beispiel:

MITTE("Michael Friedrich";2;3) liefert "ich"

MITTE("Michael Friedrich";8;20) liefert "Friedrich"

MITTE("Michael Friedrich";20;20) liefert ""

Verwandte Funktionen:

LINKS , RECHTS , TEIL

1.57 RECHTS(Text;Anzahl)

RECHTS(Text;Anzahl)

(Englisch: RIGHT)

Die Funktion RECHTS liefert die letzten Anzahl Zeichen des Textes (von rechts)

Anzahl muß dabei größer als 0 sein (sonst erscheint die Fehlermeldung #WERT). Ist Anzahl größer als der Länge des Textes, so wird der Text zurückgegeben.

Wird Anzahl ausgelassen (also RECHTS(Text)), so wird das letzte Zeichen des Textes zurückgegeben.

Beispiel:

RECHTS("Michael Friedrich";3) ergibt "ich"

RECHTS("Hallo") ergibt "o"

Verwandte Funktionen:

LINKS , MITTE , TEIL

1.58 SÄUBERN(Text)

SÄUBERN(Text)

(Englisch: CLEAN)

Die Funktion SÄUBERN entfernt aus dem Text alle nicht druckbaren Zeichen, also Steuercodes. Dies ist sinnvoll, wenn Sie einen Text importiert haben und dieser Steuerzeichen enthielt (dies ist am Bildschirm oft an einem rechteckigen Kasten zu erkennen).

Beispiel:

SÄUBERN(ZEICHEN(7)+"Text") liefert "Text"

Verwandte Funktionen:

ZEICHEN , GLÄTTEN

1.59 SCHIEBENL(Text)

SCHIEBENL(Text)

(Englisch: SHIFTL)

Verschiebt die Buchstabereihenfolge um eins nach links (d.h. das zweite Zeichen wird erstes), das Zeichen ganz links wird am Ende angefügt.

Text: ein beliebiger Text

Beispiel:

SCHIEBENL("Hallo") ergibt "alloH"

1.60 SCHIEBENR(Text)

SCHIEBENR(Text)

(Englisch: SHIFTR)

Verschiebt die Buchstabereihenfolge um eins nach rechts, d.h. das letzte Zeichen wird entfernt und am Anfang angefügt.

Text: ein beliebiger Text

Beispiel:

SCHIEBENR("Hallo") ergibt "oHall"

1.61 SPIEGELN(Text)

SPIEGELN(Text)

(Englisch: REVERSE)

Dreht die Buchstabenreihenfolge von Text um.

Text: ein beliebiger Text

Beispiel:

SPIEGELN("Hallo") ergibt "ollaH"

1.62 TEIL(Text;Zahl1;Zahl2)

TEIL(Text;Zahl1;Zahl2)

(Englisch: PART)

Diese Funktion ist äquivalent zu MITTE. Sie wurde nur zur Kompatibilität aufgenommen, Beschreibung siehe bei MITTE.

1.63 TEXT(Data[;Format])

TEXT(Data[;Format])

Die Funktion TEXT wandelt alle Datentypen (Zahlen, Wahrheitswerte, Datum, Uhrzeit und Texte) in Text um.

Data: Kann ein beliebiger Ausdruck sein

Format: Gibt das Format an, wie Data als Text formatiert werden soll. 0 bzw. Auslassen des Parameters entspricht dem Standard-Format. Die weiteren Formatnummern können Sie in der Tabelle bei Makrobefehl ZAHLENFORMAT ablesen. (bzw. im Fenster zu <Format-Zahlenformat> abzählen und dabei bei 0 beginnen).

Diese Funktion ist somit das Gegenstück zu WERT.

Beispiel:

TEXT(123;3) ergibt "123.00"

TEXT(12)+" DM" ergibt "12 DM"

WERT(TEXT(123.45;1)) ergibt 123 (da TEXT(123.45;1) "123" erzeugt).

1.64 WERT(Text)

WERT(Text)

(Englisch: VALUE)

Die Funktion Wert wandelt den Text in eine Zahl, ein Datum, eine Uhrzeit oder einen Fehlerwert um.

Dies ist sinnvoll, wenn man Datums- oder Zeitwerte in Formeln eingeben möchte, da dort eine normale Eingabe nicht möglich ist. Weiterhin kann diese Funktion gut benutzt werden, bestimmte Zellinhalte zu prüfen, ohne eine Fehlermeldung zu riskieren (enthalten sie Zahlen, so werden diese einfach weitergegeben, bei normalen Texten wird 0 zurückgegeben).

Diese Funktion ist somit das Gegenstück zu TEXT.

Wert: kann als Parameter auch eine Zahl, Boolean, Datum, Uhrzeit sein. In diesem Fall wird dann dies entsprechend zurückgegeben. (WERT(42)=42, WERT(b1) mit b1=13:02 ergibt 13:02)

Damit können jetzt sehr leicht Zellinhalte, die sowohl Zahlen als auch Texte enthalten können, getestet werden, ohne einen #TYP Fehler zu erhalten. (etwa: SPRINGEWENN(WERT(b1)=0;c5) - springt wenn b1 0 ist oder Text enhält, der nicht als Zahl interpretiert werden kann)

Beispiel:

WERT("1000") ergibt 1000.

WERT("20 Jan") liefert 20.01.1993 (im Jahr 1993)

WERT(123) ergibt 123

Enthält die Zelle A1 ein Datum, so kann man damit die Tage seit 1.1.1992 berechnen; A1-WERT("1.1.1992")

1.65 WIEDERHOLEN(Text; Anzahl)

WIEDERHOLEN(Text; Anzahl)

(Englisch: REPEAT)

Die Funktion WIEDERHOLEN liefert als Ergebnis einen Text, der aus Anzahl Wiederholungen des Parameters Text besteht.

Anzahl muß dabei größer als 0 sein, ansonsten wird #WERT zurückgegeben.

Beispiel:

WIEDERHOLEN("-";5) ergibt ""

WIEDERHOLEN("Text";3) ergibt "TextTextText"

1.66 ZEICHEN(asc-code)

ZEICHEN(asc-code)

(Englisch: CHAR)

Die Funktion ZEICHEN liefert das Zeichen (d.h. Text der Länge 1), den die Zahl asc-code repräsentiert. (Hierzu wird der ASCII-Code benutzt). Dies ist also die Umkehrung zu CODE.

asc-code muß sich zwischen 1 und 255 befinden!

Beispiel:

ZEICHEN(65) ergibt "A"

ZEICHEN(CODE("T")) ergibt "T"

Verwandte Funktionen:

CODE

1.67 Datum-Funktionen

Datum-Funktionen

Den meisten der unten aufgeführten Funktionen muß ein Datum bzw. eine Uhrzeit übergeben werden (oder sie liefern diese zurück).

Deswegen hier zu diesen beiden Werten einige Informationen:

Datum: Das Datum wird intern als laufende Ganzzahl gespeichert, welches der Zahl 0 den 1.1.1900 zuweist. So hat etwa das Datum 1.7.93 den Wert 34149.

(Mittels <Format-Zahlenformat> kann man die Anzeige als Zahl ("0") oder als Datum ("TT-MM-JJJJ") erzwingen - Standard entscheidet selbständig).

Diese Möglichkeit kann man für fortschrittliche Anwendungen benutzen (etwa ergibt "Datum-Datum" gerade die Anzahl der Tage zwischen den beiden Tagen, siehe unten.)

Uhrzeit: Entsprechend wird auch die Uhrzeit gespeichert: Als Anzahl der Sekunden seit 0 Uhr!

Somit kann meist, falls ein Datum als Parameter verlangt wird, auch eine normale Zahl eingegeben werden. Beachten Sie jedoch dann die Interpretation dieser Zahl!

Plus: +, Minus: -

DATUM(Jahr;Monat;Tag)

DATWERT(Text)

HEUTE()

JAHR(Datum)

JETZT()

MONAT(Datum)

TAG(Datum)

WOCHENTAG(Datum)

ZEITWERT(Text)

1.68 Plus: +, Minus: -

Plus: +, Minus: -

Die Operationen "+" und "-" sind auch für das Datum und die Uhrzeit definiert, und zwar nach folgenden Regeln:

Datum + Zahl = Datum (Datum um Zahl Tage erhöhen)

Zahl + Datum = Datum (das gleiche)

Datum - Zahl = Datum (Datum um Zahl Tage vermindern)

Datum - Datum = Zahl (Anzahl der Tage zwischen den beiden Daten)

(dabei stellt bei Zahl der Wert 1 genau einen Tag dar - Kommawerte werden ignoriert)

Uhrzeit + Zahl = Uhrzeit (Uhrzeit um Zahl Sekunden erhöhen)

Zahl + Uhrzeit = Uhrzeit (das gleiche)

Uhrzeit - Zahl = Uhrzeit (Uhrzeit um Zahl Sekunden vermindern)

Uhrzeit - Uhrzeit = Zahl (Anzahl der Sekunden zwischen den beiden Uhrzeiten)

(dabei stellt bei Zahl der Wert 1 genau eine Sekunde dar - Kommawerte werden ignoriert)

Hinweis: Bei anderen Operationen (etwa ZAHL-DATUM) wird das Datum in eine Zahl gewandelt und dann normal mit der Zahl weitergerechnet.

Beispiel:

WERT("5-1-1993")-WERT("1-1-1993") liefert 4

WERT("1-1-1993"+4) ergibt 5-4-1993

WERT("18:00")-WERT("12:00") ergibt 21600 (=6*60*60)

Verwandte Funktionen:

WERT , DATWERT , ZEITWERT , DATUM

1.69 DATUM(Jahr;Monat;Tag)

DATUM(Jahr;Monat;Tag)

(Englisch: DATE)

Die Funktion DATUM wandelt die drei Parameter in ein Datum um.

Jahr, Monat und Tag müssen dabei einen existierenden Tag ab 1.1.1900 bis 31.12.2400 darstellen - ansonsten wird der Fehler #WERT zurückgegeben.

Beispiel:

DATUM(1993;4;1) ergibt den 1. April 1993

Verwandte Funktionen:

DATWERT , TAG , MONAT , JAHR , HEUTE , JETZT , ZEITWERT

1.70 DATWERT(Text)

DATWERT(Text)

(Englisch: DATEVALUE)

Die Funktion DATWERT wandelt einen Text in ein Datum um. (Es kann auch einfach WERT benutzt werden, was alle Formate umwandeln kann, siehe dort.)

Text kann ein beliebiges Datumsformat in Textform sein, etwa:

"30-9-93"

"30 Sep 93"

"Sep 93"

"30 Sep"

(eine vollständige Beschreibung aller Möglichkeiten siehe im Kapitel "Die Eingabe")

Tip: Diese Funktion ist sehr praktisch, wenn man Datumswerte in Formeln verwenden möchte, da man obige Daten nicht einfach direkt eingeben kann.

Beispiel:

DATWERT("1.4.93") ergibt den 1. April 93

Verwandte Funktionen:

WERT , DATUM , HEUTE , ZEITWERT

1.71 HEUTE()

HEUTE()

(Englisch: TODAY)

Die Funktion HEUTE liefert als Ergebnis das Datum des heutigen Tages.

Hinweis: Achten Sie darauf, daß das Datum auch stimmt, falls Sie diese Funktion benutzen. Haben Sie eine interne (gepufferte) Uhr eingebaut, so sollte dies der Fall sein. Ansonsten sollten Sie das Datum mittels DATE per CLI oder aber per Preferences einstellen, wenn Sie sinnvolle Ergebnisse wünschen.

Beispiel:

HEUTE()+2 liefert das Datum von Übermorgen

Verwandte Funktionen:

DATUM , DATWERT , JETZT

1.72 JAHR(Datum)

JAHR(Datum)

(Englisch: YEAR)

Die Funktion JAHR liefert von einem Datum die Jahreszahl als Zahl zurück.

Beispiel:

JAHR(WERT("23-3-93") ergibt 1993

JAHR(2000) = 1905

Verwandte Funktionen:

MONAT , TAG , WOCHENTAG , HEUTE , JETZT

1.73 JETZT()

JETZT()

(Englisch: NOW)

Die Funktion JETZT liefert als Ergebnis die aktuelle Uhrzeit zurück.

Hinweis: Achten Sie darauf, daß die Uhrzeit auch stimmt, falls Sie diese Funktion benutzen. Haben Sie eine interne (gepufferte) Uhr eingebaut, so sollte dies der Fall sein. Ansonsten sollten Sie das Datum und die Uhrzeit mittels DATE per CLI oder aber per Preferences einstellen, wenn Sie sinnvolle Ergebnisse wünschen.

Beispiel:

JETZT() liefert die aktuelle Uhrzeit

Verwandte Funktionen:

ZEITWERT , HEUTE

1.74 MONAT(Datum)

MONAT(Datum)

(Englisch: MONTH)

Die Funktion MONAT liefert von einem Datum die Monatszahl als Zahl zurück.

Beispiel:

MONAT(WERT("23-3-93") ergibt 3

MONAT(2000) = 6

Verwandte Funktionen:

JAHR , TAG , WOCHENTAG , HEUTE , JETZT

1.75 TAG(Datum)

TAG(Datum)

(Englisch: DAY)

Die Funktion TAG liefert von einem Datum die Tageszahl als Zahl zurück.

Beispiel:

TAG(WERT("23-3-93") ergibt 23

TAG(2000) = 24

Verwandte Funktionen:

MONAT , JAHR , WOCHENTAG , HEUTE , JETZT

1.76 WOCHENTAG(Datum)

WOCHENTAG(Datum)

(Englisch: WEEKDAY)

Die Funktion WOCHENTAG berechnet aus einem Datum den Wochentag dieses Datums und gibt ihn als Zahl von 1 bis 7 zurück (1 = Sonntag. 7 = Samstag)

Beispiel:

WOCHENTAG(DATUM("1 Juni 92")) ergibt 3 (Dienstag)

AUSWAHL(WOCHENTAG(HEUTE);"So";"Mo";"Di";"Mi";"Do";"Fr";"Sa") ergibt das zweistellige Kürzel des heutigen Tages

Verwandte Funktionen:

JAHR , MONAT , TAG , HEUTE , JETZT

1.77 ZEITWERT(Text)

ZEITWERT(Text)

(Englisch: TIMEVALUE)

Die Funktion ZEITWERT wandelt einen Text in eine Uhrzeit um. (Es kann auch einfach WERT benutzt werden, was alle Formate umwandeln kann, siehe dort.)

Text kann ein beliebiges Zeitformat in Textform sein, also:

13:03 oder

13:03:05 (mit Sekundenangabe)

Tip: Diese Funktion ist sehr praktisch, wenn man Zeitwerte in Formeln verwenden möchte, da man obige Daten nicht einfach direkt eingeben kann.

Beispiel:

ZEITWERT("13:45") ergibt 13 Uhr 45 und 0 Sekunden

Verwandte Funktionen:

WERT , DATUM , HEUTE , DATWERT

1.78 Tabellen-Funktionen

Tabellen-Funktionen

Die Tabellenfunktionen haben als Parameter alle eine Liste aus Bereichen.

Ein Bereich ist dabei entweder eine Zelle (also A1...) oder aber ein rechteckiger Zellblock, der durch Zelle1 + ":" + Zelle2 angegeben wird (also etwa A1:C3). Statt einem Doppelpunkt kann zur Kompatibilität auch ".." (zwei Punkte nacheinander) verwandt werden.

Natürlich können die Zellen/Bereiche auch absolut sein (mit Dollar "$") oder aber durch Namen ersetzt werden, siehe dazu bei "Namen".

Für alle unten stehenden Funktionen können beliebig viele Bereiche, jeweils durch Semikolon getrennt, nacheinander aufgeführt werden. Weiterhin können auch direkt Zahlen als Parameter eingegeben werden; dies ist vorallem bei MIN und MAX sinnvoll, Beispiele siehe dort.

ANZAHL(Bereich[;...])

ANZAHL2(Bereich[;...])

MAX(Bereich[;...])

MIN(Bereich[;...])

MITTELWERT(Bereich[;...])

PRODUKT(Bereich[;...])

STABW(Bereich[;...])

SUMME(Bereich[;...])

VARIANZ(Bereich[;...])

Für alle Beispiele wird jeweils folgende Tabelle benutzt:

A B C

1 2 4

2 3 5

3 Text

4

5

1.79 ANZAHL(Bereich[;...])

ANZAHL(Bereich[;...])

(Englisch: COUNT)

Die Funktion ANZAHL liefert die Anzahl der Zahlen in den Bereichen. Texte und leere Zellen werden ignoriert.

Beispiel:

ANZAHL(A1:A5) = 2

ANZAHL(A1;A2:B2;A3:B3) = 4

Verwandte Funktionen:

ANZAHL2 , MAX , MIN , SUMME , PRODUKT ...

1.80 ANZAHL2(Bereich[;...])

ANZAHL2(Bereich[;...])

(Englisch: COUNT2)

Die Funktion ANZAHL2 liefert die Anzahl der nichtleeren Zellen in den angegebenen Bereichen. Leerzellen werden ignoriert, Texte, Daten sowie Uhrzeiten werden jedoch gezählt.

Beispiel:

ANZAHL(A1:A5) = 3

ANZAHL(A1;A2:B2;A3:B3) = 4

Verwandte Funktionen:

ANZAHL , MAX , MIN , SUMME , PRODUKT ...

1.81 MAX(Bereich[;...])

MAX(Bereich[;...])

Die Funktion MAX liefert die größte Zahl der angegebenen Bereiche. Texte und leere Zellen werden ignoriert. Enthält der Bereich keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

MAX(A1:A5) = 3

MAX(A1;A2:B2;A3:B3) = 5

MAX(A1:F20;1) sucht das Maximum aus dem angegebenen Bereich. Ist es kleiner als 1, so wird 1 verwandt.

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MIN , SUMME , PRODUKT ...

1.82 MIN(Bereich[;...])

MIN(Bereich[;...])

Die Funktion MIN liefert die kleinste Zahl der angegebenen Bereiche. Texte und leere Zellen werden ignoriert. Enthält der Bereich keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

MIN(A1:A5) = 1

MIN(A1;A2:B2;A3:B3) = 1

MIN(A1;A4) = 0

MIN(A1:F20;100) bestimmt das Minimum des angegebenen Bereichs. Ist dieses größer als 100, so wird 100 verwandt.

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , SUMME , PRODUKT ...

1.83 MITTELWERT(Bereich[;...])

MITTELWERT(Bereich[;...])

(Englisch: AVERAGE)

Die Funktion MITTELWERT berechnet den Durchschnitt aller Zahlen der angegebenen Bereiche. Texte und leere Zellen werden ignoriert. Enthält der Bereich keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

MITTELWERT(A1:A5) = 2.5

MITTELWERT(A1;A2:B2;A3:B3) = 3.5

MITTELWERT(A1;A4) = 0

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , SUMME , PRODUKT ...

1.84 PRODUKT(Bereich[;...])

PRODUKT(Bereich[;...])

(Englisch: PRODUCT)

Die Funktion PRODUKT liefert das Produkt aller Zahlen der angegebenen Bereiche. Texte und leere Zellen werden ignoriert. Enthält der Bereich keine Zahlen, so wird 0 zurückgegeben - ist eine der Zahlen 0, so wird natürlich auch das Produkt 0.

Beispiel:

PRODUKT(A1:A5) = 6

PRODUKT(A1;A2:B2;A3:B3) = 120

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , MIN , SUMME ...

1.85 STABW(Bereich[;...])

STABW(Bereich[;...])

(Englisch: STDEV)

Die Funktion STABW berechnet die Standardabweichung der angegebenen Werte.

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , MIN , PRODUKT ...

1.86 SUMME(Bereich[;...])

SUMME(Bereich[;...])

(Englisch: SUM)

Die Funktion SUMME liefert die Summe aller Zahlen der angegebenen Bereiche. Texte und leere Zellen werden ignoriert. Enthält der Bereich keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

SUMME(A1:A5) = 5

SUMME(A1;A2:B2;A3:B3) = 14

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , MIN , PRODUKT ...

1.87 VARIANZ(Bereich[;...])

VARIANZ(Bereich[;...])

(Englisch: VAR)

Die Funktion VARIANZ berechnet die Varianz der angegebenen Werte.

Verwandte Funktionen:

ANZAHL , ANZAHL2 , MAX , MIN , PRODUKT ...

1.88 Datenbank-Funktionen

Datenbank-Funktionen

Diese Funktionen sind ähnlich den obigen Tabellen-Funktionen, jedoch wird jetzt statt einem "normalen" Bereich eine Datenbank zugrunde gelegt. Alle unten aufgeführten Funktionen haben deswegen den gleichen Syntax:

DBFunktion(Datenbankbereich;Spalte;Kriterienbereich)

Datenbankbereich: Es handelt sich um einen Bereich wie er im Abschnitt "Datenbank" beschrieben wurde (also: erste Zeile Titel, restliche Zeilen Daten). Es muß sich nicht um die aktuelle definierte Tabelle handeln (kann aber, indem der Name "DATENBANK" benutzt wird).

Spalte: Dies gibt die Spalte (des Datenbankbereichs) an, auf die sich die Funktionen (nicht der Kriterienbereich) beziehen sollen. Diese kann als Nummer (0= erste Spalte, 1 = zweite Spalte...) oder als Spaltentitel-Text (in Anführungszeichen!) eingegeben werden.

Kriterienbereich: Dieser Bereich bestimmt, welche Datensätze zur Berechnung herangezogen werden sollen. Es werden nur die Datensätze (der ausgewählten Spalte) benutzt, die mit den Kriterien übereinstimmen. Näheres (auch zum Aufbau dieses Bereichs) siehe im Abschnitt "Datenbank" unter "Suchkriterien".

Tip: Da diese Funktionen nicht nur auf den aktuellen Datenbankbereich angewandt werden können, sondern auf beliebige (richtig eingerichtete) Bereiche, kann man diese Funktionen auch als Auswahlkriterien nutzen (um etwa die Spalten mit einem Wert>2000 oder ein Datum zwischen 1.1.1990 und 1.1.1991 zu zählen bzw. aufzusummieren...)

Tip: Da (vor allem bei großen Bereichen) diese Funktionen relativ lange zum Berechnen brauchen (da sie ja alle Datensätze vergleichen müssen), empfiehlt es sich, hier die automatische Berechnung auszuschalten.

DBANZAHL(Datenbank;Spalte;Kriterien)

DBANZAHL2(Datenbank;Spalte;Kriterien)

DBMAX(Datenbank;Spalte;Kriterien)

DBMIN(Datenbank;Spalte;Kriterien)

DBMITTELWERT(Datenbank;Spalte;Kriterien)

DBPRODUKT(Datenbank;Spalte;Kriterien)

DBSTABW(Datenbank;Spalte;Kriterien)

DBSUMME(Datenbank;Spalte;Kriterien)

DBVARIANZ(Datenbank;Spalte;Kriterien)

1.89 DBANZAHL(Datenbank;Spalte;Kriterien)

DBANZAHL(Datenbank;Spalte;Kriterien)

(Englisch: DCOUNT)

Die Funktion DBANZAHL liefert die Anzahl der Zahlen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Beispiel:

DBANZAHL(DATENBANK;0;SUCHKRITERIEN) liefert die Anzahl der Zahlen in der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBANZAHL(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.90 DBANZAHL2(Datenbank;Spalte;Kriterien)

DBANZAHL2(Datenbank;Spalte;Kriterien)

(Englisch: DCOUNT2)

Die Funktion DBANZAHL liefert die Anzahl der nichtleeren Zellen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Leerzellen werden ignoriert, Texte, Daten sowie Uhrzeiten werden jedoch gezählt.

Diese Funktion ist sehr nützlich zum Bestimmen der Anzahl der Datensätze, die den Suchkriterien entsprechen.

Beispiel:

DBANZAHL2(DATENBANK;0;SUCHKRITERIEN) liefert die Anzahl der nichtleeren Zellen in der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBANZAHL2(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.91 DBMAX(Datenbank;Spalte;Kriterien)

DBMAX(Datenbank;Spalte;Kriterien)

(Englisch: DMAX)

Die Funktion DBMAX liefert die größte Zahl der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Stellen die Suchkriterien keine Übereinstimmung fest bzw. sind in diesen Zeilen in der ausgewählten Spalte keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

DBMAX(DATENBANK;0;SUCHKRITERIEN) liefert die größte Zahl der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBMAX(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.92 DBMIN(Datenbank;Spalte;Kriterien)

DBMIN(Datenbank;Spalte;Kriterien)

(Englisch: DMIN)

Die Funktion DBMIN liefert die kleinste Zahl der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Stellen die Suchkriterien keine Übereinstimmung fest bzw. sind in diesen Zeilen in der ausgewählten Spalte keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

DBMIN(DATENBANK;0;SUCHKRITERIEN) liefert die kleinste Zahl der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBMIN(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.93 DBMITTELWERT(Datenbank;Spalte;Kriterien)

DBMITTELWERT(Datenbank;Spalte;Kriterien)

(Englisch: DAVERAGE)

Die Funktion DBMITTELWERT liefert den Mittelwert aller Zahlen der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Stellen die Suchkriterien keine Übereinstimmung fest bzw. sind in diesen Zeilen in der ausgewählten Spalte keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

DBMITTELWERT(DATENBANK;0;SUCHKRITERIEN) liefert den Durchschnitt der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBMITTELWERT(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.94 DBPRODUKT(Datenbank;Spalte;Kriterien)

DBPRODUKT(Datenbank;Spalte;Kriterien)

(Englisch: DPRODUCT)

Die Funktion DBPRODUKT liefert das Produkt aller Zahlen der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Stellen die Suchkriterien keine Übereinstimmung fest bzw. sind in diesen Zeilen in der ausgewählten Spalte keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

DBPRODUKT(DATENBANK;0;SUCHKRITERIEN) multipliziert alle Zahlen der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBPRODUKT(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.95 DBSTABW(Datenbank;Spalte;Kriterien)

DBSTABW(Datenbank;Spalte;Kriterien)

(Englisch: DSTDEV)

Die Funktion DBSTABW berechnet die Standardabweichung aller Zahlen der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.96 DBSUMME(Datenbank;Spalte;Kriterien)

DBSUMME(Datenbank;Spalte;Kriterien)

(Englisch: DSUM)

Die Funktion DBSUMME liefert die Summe aller Zahlen der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Stellen die Suchkriterien keine Übereinstimmung fest bzw. sind in diesen Zeilen in der ausgewählten Spalte keine Zahlen, so wird 0 zurückgegeben.

Beispiel:

DBSUMME(DATENBANK;0;SUCHKRITERIEN) addiert alle Zahlen der ersten Spalte des Datenbankbereichs, die mit den Suchkriterien übereinstimmen

DBSUMME(DATENBANK;"Nummer";SUCHKRITERIEN) entsprechend, falls die erste Spalte den Titel "Nummer" hat.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.97 DBVARIANZ(Datenbank;Spalte;Kriterien)

DBVARIANZ(Datenbank;Spalte;Kriterien)

(Englisch: DVAR)

Die Funktion DBVARIANZ berechnet die Varianz aller Zahlen der gewünschten Spalte aller Zeilen des Datenbankbereichs, die mit den Kriterien übereinstimmen - näheres zu den Parametern siehe Einführung zu den Datenbankfunktionen.

Verwandte Funktionen:

Datenbankfunktionen , Tabellenfunktionen

1.98 Zell-Funktionen

Zell-Funktionen

#Bezug

@Tabelle;Bezug

AUSWAHL(Index; Wert1; Wert2; Wert3...)

BEREICHABS(Zeile;Spalte;Höhe;Breite)

BLOCKBREITE([Bereich])

BLOCKHÖHE([Bereich])

BLOCKX([Bereich])

BLOCKY([Bereich])

INDIREKT(Text)

SPALTENNUMMER([Bereich])

SVERWEIS(Wert;Bereich;Offset[;Exakt])

TABELLENNAME()

VERWEIS(Wert;Bereich[;Exakt])

WVERWEIS(Wert;Bereich;Offset[;Exakt])

ZEILENNUMMER([Bereich])

ZELLE(Zeile;Spalte)

ZELLEABS(Zeile;Spalte)

1.99 #Bezug

#Bezug

(Diese Funktion ist nur in Makro-Befehlen definiert, ansonsten wird sie meist eine Fehlermeldung oder einen beliebigen Wert anzeigen).

Bezüge innerhalb von Makrobefehlen beziehen sich fast immer auf die Tabelle, aus der das Makro aufgerufen wurde und nicht auf die Makro-Tabellen (Ausnamen: SPRINGE, SPRINGEWENN, siehe dort).

Will man eine Zelle der Makro-Tabelle ansprechen, so ist dies mit #Bezug möglich, wobei Bezug die gewünschte Zelle bzw. den gewünschten Bereich darstellt.

(#Bezug entspricht also @Makrotabelle;Bezug, wenn Makrotabelle der Tabellenname ist, in der sich das gerade laufende Makro befindet)

Beispiel:

folgender Makroausschnitt realisiert eine Sicherheitsabfrage:

A10 =ANFRAGE("Ja oder Nein?")

A11 =SPRINGEWENN(not(#A10);A20)

A12 hier steht das eigentliche Programm

... ...

A20 =RETURN()

Der erste Befehl ANFRAGE fragt nach (hier kurz: "Ja oder Nein?"), ob etwas ausgeführt werden soll.

Der Anwender kann dann auf >OK< oder >Abbruch< klicken. Bei >OK< befindet sich WAHR in Zelle A10, sonst FALSCH

In Zelle A11 wird dann dieser Inhalt geprüft. (Würde hier das # fehlen und wurde das Makro aus einer anderen Tabelle heraus gestartet, so würde der Inhalt der Zelle dieser Tabelle ausgelesen - und dies kann ja beliebig sein!). Ist er FALSCH, dann ist not(#A10) WAHR und so wird zu Zelle A20 gesprungen, also die eigentliche Routine ausgelassen.

Verwandte Funktionen:

1.100 @Tabelle;Bezug

@Tabelle;Bezug

(oder AT(Tabelle;Bezug))

Die AT-Funktion (d.h. der "Klammeraffe" (Alt-2)) liest Werte aus Zellen von anderen Tabellen aus, kann also benutzt werden, um mehrere Tabellen miteinander zu verknüpfen.

Tabelle: Dies ist der Name einer Tabelle (wie er als Fenster-Titel erscheint). Es die Datei geladen, so reicht der Name ohne Pfadangabe und Endung, sofern diese .TCD ist. Der Name wird mit Semikolon ";" abgeschlossen. Der Name kann in Anführungszeichen eingeschlossen werden (und muß es, falls er ein Semikolon ";" enthält!) Ist die Tabelle nicht geladen, so muß eine Pfadangabe mit angegeben werden, damit TurboCalc die Datei finden kann (siehe auch unten).

Die geladene Datei "Dh0:Daten/Test.TCD" kann also mit Test, Test.TCD, Daten/Test.TCD oder dh0:Daten/Test.TCD angesprochen werden.

Bezug: Gibt einen normalen Bezug (etwa A1 oder A1). Es spielt dabei keine Rolle, ob der Bezug relativ oder absolut (d.h. mit Dollarzeichen) angegeben wird.

Hinweis: Es kann wahlweise @ oder AT benutzt werden, weiterhin die beiden (durch Semikolon getrennten Parameter!) auch geklammert werden.

Ab der Version 3 kann mit der AT-Funktion auch auf geschlossene Tabellen zugegriffen werden: (Jedoch nur zum Auslesen von Zellen, etwa @Test;a7 - für Bereichs-Angaben mit diesem Befehl muß die andere Tabelle weiterhin geöffnet.)

Ist jedoch nun die gewünschte Tabelle nicht geöffnet, so wird versucht, die angegebene Zelle aus der Datei auszulesen. Das gelesene Ergebnis wird intern zwischengespeichert, so daß nur einmal auf Disk zugegriffen wird. Die Suchreihenfolge ist wie folgt:

i) Enthält der Dateinamen einen Doppelpunkt ":", so wird nur in diesem Pfad gesucht (da es sich ja um eine "absoluten" Pfadangabe handelt).

ii) Ansonsten wird nacheinander in folgenden Pfaden gesucht:

1. Pfad der Tabelle, in der die Funktion eingegeben wurde. (Falls hier ein Pfad gesetzt ist, d.h. im Dateinamen ein ":" oder "/" vorhanden ist).

2. Pfad aus dem eine Tabelle per Workbench gestartet wurde (Doppelklick auf Icon oder Shift-Klick auf Icon und Programm)

3. Pfad, in dem sich TurboCalc befindet.

4. Aktiver Pfad beim Start von TurboCalc (nur Shell)

Konnte die Datei bis jetzt nicht gefunden werden, so wird ein ".TCD" angehängt und die Suche von 1. bis 4. wiederholt.

Daher: Die Angabe von .TCD ist nicht notwendig. Es empfiehlt sich jedoch, den Pfad komplett angzugeben (und ggf. auch .TCD). Umso sicherer kann TurboCalc die Tabelle finden. Aber: Dadurch wird auch ein Portieren erschwert. Je nach Anwendung ist daher auch eine "relative" Angabe (zur aktuellen Tabelle) sinnvoll.

Tip: Der gelesene Wert (oder Text) wird intern mit Tabellenname und Zelle zwischengespeichert, so daß er nur einmal gelesen werden muß, aber: hierbei wird zwischen verschiendenen Namen unterschienden (mit/ohne ".TCD", anderer Pfad) - auch wenn alle Namen möglicherweise zu den gleichen Daten führen! (Groß/Kleinschreibung ist egal) Deshalb: Möglichst immer die gleiche Schreibweise benutzen (etwa einfach durch Kopieren der Formel). Ansonsten kann es passieren, daß die gleichen Daten mehrfach gelesen und zwischengespeichert werden. (Dies führt natürlich zu keinen Fehler, sondern erhöht nur den Speicherverbrauch.)

Hinweis: Konnte die Tabelle nicht geöffnet werden, so erscheint weiterhin eine Fehlermeldung (ist die Zelle leer, so wird 0 zurückgegeben.)

Anmerkung: Mit Passwort geschützte Zellen können hiermit nicht angesprochen werden (diese müssen geöffnet sein). Diese werden als "nicht gefunden" interpretiert.

WICHTIGER HINWEIS: In den älteren Version von TC2 (bis TC2.18) enthielt die Speicherroutine einen kleinen Fehler, der ein Einlesen von Werten aus so gespeicherten Tabellen verhinderte (sobald Formeln gesspeichert wurden).Einfache Abhilfe: Tabelle(n) laden und mit der aktuellen Version wieder Speichern.

Beispiel:

Wenn die Datei DH0:Daten/Test.TCD geöffnet ist und in der Zelle C5 der Wert 123 steht, so liefern alle folgenden Ausdrücke diesen Wert zurück:

TestDaten/TestDH0:daten/testAT(Test;C5)

Lautet der Dateiname z.B "Abrechnung Januar; Februar.TCD", so müssen Anführungszeichen stehen:

Abrechnung Januar; FebruarAbrechnung Januar; Februar.TCDVerwandte Funktionen:

#Bezug

1.101 AUSWAHL(Index; Wert1; Wert2; Wert3...)

AUSWAHL(Index; Wert1; Wert2; Wert3...)

(Englisch: CHOOSE)

Die Funktion AUSWAHL gibt in Abhängigkeit von Index den Wert1, 2... zurück.

Dabei stellt Index eine positive Zahl ab 1 dar, die angibt, welcher der darauf folgenden Werte zurückgegeben werden soll. Ist Index kleiner eins oder aber größer als die Anzahl der Werte, so wird #WERT zurückgegeben.

Wert1,2... können beliebige Ausdrücke (Texte, Zahlen...) sein, die dann zurückgegeben werden. (Auch weitere AUSWAHL(..;..;..) als Parameter sind möglich - AUSWAHL läßt sich also schachteln).

Beispiel:

AUSWAHL(1;"Hallo";"Du";123) ergibt "Hallo"

AUSWAHL(2;"Hallo";"Du";123) ergibt "Du"

AUSWAHL(3;"Hallo";"Du";123) ergibt 123

AUSWAHL(4;"Hallo";"Du";123) ergibt #WERT

AUSWAHL(WOCHENTAG(HEUTE);"So";"Mo";"Di";"Mi";"Do";"Fr";"Sa") ergibt den zweistelligen Kürzel des heutigen Tages (WOCHENTAG gibt dabei eine Zahl zwischen 1 und 7 zurück!)

Verwandte Funktionen:

WENN

1.102 BEREICHABS(Zeile;Spalte;Höhe;Breite)

BEREICHABS(Zeile;Spalte;Höhe;Breite)

(Englisch: RANGEABS)

Die Funktion BEREICHABS definiert einen Block der Breite Breite und der Höhe Höhe ab der Position Zeile, Spalte. Alle Werte müssen größer als 0 sein (bei kleineren Werten erscheint #WERT)!

Hinweis: Bitte beachten Sie, daß zuerst die Zeile und dann die Spalte kommt. Dies ist die Art, wie Tabellen normalerweise gelesen werden (entsprechend Matrizen!) - im Koordinatensystem würde dies jedoch Y,X bedeuten! Gleiches gilt dann auch für die Höhe und Breite.

Tip: Diese Funktion kann in allen Funktionen bzw. Befehlen als Parameter benutzt werden, die einen Block als Parameter erwarten. Somit lassen sich innerhalb von Makros recht einfach benutzerdefinierte Bereiche angeben.

Tip: Statt BEREICHABS kann auch ZELLEABS benutzt werden, etwa: ZELLEABS(1;3):ZELLEABS(7;5)

Beispiel:

BEREICHABS(3;4;1;2) entspricht (D3:D4).

SELECT(BEREICHABS(3;4;1;2)) wählt den entsprechenden Bereich aus.

Verwandte Funktionen:

ZELLE , INDIREKT , ZELLEABS

1.103 BLOCKBREITE([Bereich])

BLOCKBREITE([Bereich])

(Englisch: RANGEWIDTH)

Diese Funktion gibt die Breite des als Parameter angegebenen Bereichs zurück. Dies kann vor allem im Zusammenhang mit Makros ganz nützlich sein.

Wird Bereich nicht angegeben, so wird die Breite des aktuellen Blockes zurückgegeben (also der, der in der aktuellen Tabelle markiert ist). Damit kann, zusammen mit BLOCKX, BLOCKY und BLOCKHÖHE der aktuelle Block von einem Makro ausgelesen werden.

Beispiel:

BLOCKBREITE(A1:C5) ergibt 3

BLOCKBREITE(F7) ergibt 1

BLOCKBREITE() gibt die Breite des aktuellen Blocks zurück.

Verwandte Funktionen:

BLOCKX , BLOCKY , BLOCKHÖHE , ZEILENNUMMER , SPALTENNUMMER

1.104 BLOCKHÖHE([Bereich])

BLOCKHÖHE([Bereich])

(Englisch: RANGEHEIGHT)

Diese Funktion gibt die Höhe des als Parameter angegebenen Bereichs zurück. Dies kann vor allem im Zusammenhang mit Makros ganz nützlich sein.

Wird Bereich nicht angegeben, so wird die Höhe des aktuellen Blockes zurückgegeben (also der, der in der aktuellen Tabelle markiert ist). Damit kann, zusammen mit BLOCKX, BLOCKY und BLOCKBREITE der aktuelle Block von einem Makro ausgelesen werden.

Beispiel:

BLOCKHÖHE(A1:C5) ergibt 5

BLOCKHÖHE(F7) ergibt 1

BLOCKHÖHE() gibt die Höhe des aktuellen Blocks zurück.

Verwandte Funktionen:

BLOCKX , BLOCKY , BLOCKBREITE , ZEILENNUMMER , SPALTENNUMMER

1.105 BLOCKX([Bereich])

BLOCKX([Bereich])

(Englisch: RANGEX)

Diese Funktion gibt die X-Position des als Parameter angegebenen Bereichs zurück. Dies kann vor allem im Zusammenhang mit Makros ganz nützlich sein.

Wird Bereich nicht angegeben, so wird die X-Position des aktuellen Blocks zurückgegeben (also der, der in der aktuellen Tabelle markiert ist). Damit kann, zusammen mit BLOCKY, BLOCKBREITE und BLOCKHÖHE der aktuelle Block von einem Makro ausgelesen werden.

Hinweis: Diese Funktion ist ähnlich der Funktion SPALTENNUMMER, jedoch mit folgendem wichtigen Unterschied: SPALTENNUMMER bezieht sich immer auf die Cursor-Position und diese muß nicht immer mit der linken oberen Ecke des Blocks übereinstimmen!

Beispiel:

BLOCKX(A1:C5) ergibt 1

BLOCKX(F7) ergibt 5

BLOCKX() gibt die X-Position des aktuellen Blocks zurück.

Verwandte Funktionen:

BLOCKY , BLOCKBREITE , BLOCKHÖHE , ZEILENNUMMER , SPALTENNUMMER

1.106 BLOCKY([Bereich])

BLOCKY([Bereich])

(Englisch: RANGEY)

Diese Funktion gibt die Y-Position des als Parameter angegebenen Bereichs zurück. Dies kann vor allem im Zusammenhang mit Makros ganz nützlich sein.

Wird Bereich nicht angegeben, so wird die Y-Position des aktuellen Blocks zurückgegeben (also der, der in der aktuellen Tabelle markiert ist). Damit kann, zusammen mit BLOCKX, BLOCKBREITE und BLOCKHÖHE der aktuelle Block von einem Makro ausgelesen werden.

Hinweis: Diese Funktion ist ähnlich der Funktion ZEILENNUMMER, jedoch mit folgendem wichtigen Unterschied: ZEILENNUMMER bezieht sich immer auf die Cursor-Position und diese muß nicht immer mit der linken oberen Ecke des Blocks übereinstimmen!

Beispiel:

BLOCKY(A1:C5) ergibt 1

BLOCKY(F7) ergibt 7

BLOCKY() gibt die Y-Position des aktuellen Blocks zurück.

Verwandte Funktionen:

BLOCKX , BLOCKBREITE , BLOCKHÖHE , ZEILENNUMMER , SPALTENNUMMER

1.107 INDIREKT(Text)

INDIREKT(Text)

(Englisch: INDIRECT)

Die Funktion INDIREKT liest den Zellbezug aus dem Text Text aus.

Text: Ist ein Text (in Anführungszeichen!), der einen gültigen Zellbezug (bzw. Namen) enthält. (Auch die Funktionen ZELLE, ZELLEABS und INDIREKT sind im Text möglich).

Beispiel:

INDIREKT("A1") ergibt die Zelle A1

INDIREKT(A1) ergibt F5, falls in der Zelle A1 der Text "F5" steht.

Verwandte Funktionen:

ZELLE ; ZELLEABS , BEREICHABS

1.108 SPALTENNUMMER([Bereich])

SPALTENNUMMER([Bereich])

(Englisch: COLUMNNUMBER)

Diese Funktion gibt die Nummer der Spalte des angegebenen Bereichs (obere linke Ecke) oder der aktuellen Zelle an, falls kein Bereich eingegeben wurde. Die Zelle A1 hat dabei die Spaltennummer 1.

Diese Funktion kann vor allem mit Makros oder aber etwa mit der Funktion ZELLEABS gut benutzt werden. Ein Beispiel dazu finden Sie auch im Abschnitt <Namen-Namen festlegen>

Beispiel:

SPALTENNUMMER(F3) ergibt 6

SPALTENNUMMER(F3:H7) ergibt 6

SPALTENNUMMER() gibt die Spaltennummer der aktuellen Zelle zurück.

Verwandte Funktionen:

BLOCKBREITE , BLOCKHÖHE , ZEILENNUMMER , ZELLEABS , BEREICHABS

1.109 SVERWEIS(Wert;Bereich;Offset[;Exakt])

SVERWEIS(Wert;Bereich;Offset[;Exakt])

(Englisch: VLOOKUP)

Vertikales Suchen innerhalb von Bereich nach Wert: Diese SVERWEIS-Funktion sucht vertikal in der ersten Spalte von Bereich nach der ersten Zelle, deren Inhalt größer oder gleich Wert ist. Wird eine solche Zelle gefunden, gibt VLOOKUP den Wert der Zelle zurück, die sich Offset-Spalten rechts der Fundstelle befinden (ist Offset negativ, so entsprechend links der Stelle)

Sollte die Suche nicht erfolgreich sein, so wird #WERT zurückgegeben.

Dieser Befehl kann gut dazu benutzt werden, um einen bestimmten Wert in Abhängigkeit von einer Zelle aus einer Tabelle auszulesen.

Wert: Kann ein beliebiger Wert (Zahl, Text, Datum...) sein, nach dem gesucht wird.

Bei Text muß der Eintrag genau (bis auf Groß- und Kleinschreibung übereinstimmen). Bei allen anderen Werten wird (normalerweise) nach dem ersten größeren (oder gleichen) Wert gesucht.

Bereich: Der Bereich, innerhalb die Suche stattfindet. (Die Tabelle des Bereiches, d.h. die erste Spalte, sollte normalerweise sortiert sein!)

Offset: Bestimmt, wieviel Spalten rechts bzw. links der Wert in der gefundenen Zelle ausgelesen werden soll. (1=eins rechts, -1=eins links)

Exakt: Kann (und wird meist) ausgelassen. Wird der Paramater angegeben und ist er ungleich Null (etwa WAHR oder 1), so wird die "exakte" Suche ausgewählt, d.h. es wird nicht nach dem ersten größeren Wert, sondern nach GENAU dem Wert gesucht (Die Tabelle muß in diesem Fall nicht mehr sortiert sein!)

Hinweis: Es kann auch der Parameter Offset weggelassen werden, falls auch Exakt nicht angegeben wird. Offset wird in diesem Fall als 1 angenommen.

Beispiel:

SVERWEIS(10;A1:F5;1)

Verwandte Funktionen:

VERWEIS , WVERWEIS

1.110 TABELLENNAME()

TABELLENNAME()

(Englisch: SHEETNAME)

Gibt den Namen der Tabelle, in der sich die Formel befindet bzw. den aktuellen Namen im Makromodus zurück. Dieser Befehl kann im Zusammenhang mit dem Makro-Befehl WÄHLETABELLE sinnvoll eingesetzt werden.

Beispiel:

TABELLENNAME()

In der Standardtabelle "Tabelle1" ergibt dies den Text "Tabelle1"

1.111 VERWEIS(Wert;Bereich[;Exakt])

VERWEIS(Wert;Bereich[;Exakt])

(Englisch: LOOKUP)

VERWEIS ist eine Abkürzung für WVERWEIS oder SVERWEIS (abhängig von Bereich) mit einem Offset von 1. Einzelheiten finden Sie bei den entsprechenden beiden Befehlen.

Normalerweise enspricht diese Funktion WVERWEIS(Wert;Bereich;1;Exakt). Ist der Bereich jedoch nur eine Spalte breit, so wird stattdessen SVERWEIS(Wert;Bereich;1;Exakt) berechnet.

Verwandte Funktionen:

WVERWEIS , SVERWEIS

1.112 WVERWEIS(Wert;Bereich;Offset[;Exakt])

WVERWEIS(Wert;Bereich;Offset[;Exakt])

(Englisch: HLOOKUP)

Horizontales Suchen innerhalb von Bereich nach Wert: Diese WVERWEIS-Funktion sucht horizontal in der ersten Reihe von Bereich nach der ersten Zelle, deren Inhalt größer oder gleich Wert ist. Wird eine solche Zelle gefunden, gibt HLOOKUP den Wert der Zelle zurück, die sich Offset-Zeilen unter der Fundstelle befinden (ist Offset negativ, so entsprechend über der Stelle)

Sollte die Suche nicht erfolgreich sein, so wird #WERT zurückgegeben.

Dieser Befehl kann gut dazu benutzt werden, um einen bestimmten Wert in Abhängigkeit von einer Zelle aus einer Tabelle auszulesen.

Wert: Kann ein beliebiger Wert (Zahl, Text, Datum...) sein, nach dem gesucht wird.

Bei Text muß der Eintrag genau (bis auf Groß- und Kleinschreibung übereinstimmen). Bei allen anderen Werten wird (normalerweise) nach dem ersten größeren (oder gleichen) Wert gesucht.

Bereich: Der Bereich, innerhalb die Suche stattfindet. (Die Tabelle des Bereiches, d.h. die erste Zeile, sollte normalerweise sortiert sein!)

Offset: Bestimmt, wieviel Zeilen tiefer bzw. höher der Wert in der gefundenen Zelle ausgelesen werden soll. (1=eins tiefer, -1=eins hoch)

Exakt: Kann (und wird meist) ausgelassen. Wird der Paramater angegeben und ist er ungleich Null (etwa WAHR oder 1), so wird die "exakte" Suche ausgewählt, d.h. es wird nicht nach dem ersten größeren Wert, sondern nach GENAU dem Wert gesucht (Die Tabelle muß in diesem Fall nicht mehr sortiert sein!)

Hinweis: Es kann auch der Parameter Offset weggelassen werden, falls auch Exakt nicht angegeben wird. Offset wird in diesem Fall als 1 angenommen.

Beispiel:

WVERWEIS(10;A1:F5;1)

HLOOKUP(10;A1:F5)

HLOOKUP(10;A1:F5;1;0)

habe alle die gleiche Funktion

Etwa: Kosten pro Stück bei Abnahme von n Stück:

A B C D E

1: 5 10 50 100

2: 15 12 10 8

(Kosten bei Abnahme von max. n Stück)

Befindet sich in A3 dann die Stückzahl, so kann mit folgender Formel der Gesamtpreis berechnet werden: (Ergebnis für 3: 45, für 6: 60, für 10: 120)

=A3*HLOOKUP(A3;A1:E1;1)

Verwandte Funktionen:

SVERWEIS , VERWEIS

1.113 ZEILENNUMMER([Bereich])

ZEILENNUMMER([Bereich])

(Englisch: ROWNUMBER)

Diese Funktion gibt die Nummer der Zeile des angegebenen Bereichs (obere linke Ecke) oder der aktuellen Zelle an, falls kein Bereich eingegeben wurde. Die Zelle A1 hat dabei die Zeilennummer 1

Diese Funktion kann vor allem mit Makros oder aber etwa mit der Funktion ZELLEABS gut benutzt werden.

Beispiel:

ZEILENNUMMER(F3) ergibt 3

ZEILENNUMMER(F3:H7) ergibt 3

ZEILENNUMMER() gibt die Zeilennummer der aktuellen Zelle zurück.

Verwandte Funktionen:

BLOCKBREITE , BLOCKHÖHE , SPALTENNUMMER , ZELLEABS , BEREICHABS

1.114 ZELLE(Zeile;Spalte)

ZELLE(Zeile;Spalte)

(Englisch: CELL)

Die Funktion ZELLE liest den Wert aus der mit Spalte und Zeile relativ angegebenen Zelle aus und gibt ihn zurück.

Zeile und Spalte können beliebige ganze Zahlen ab 1 sein. 0 bedeutet die aktuelle Zeile/Spalte, negative Zahlen bestimmen, wie weit höher bzw. links die Zelle liegen, soll und positive geben die Richtung nach rechts unten an.

Bei der Bearbeitung von Makro-Befehlen (also den Parametern dieser Befehle) gibt diese Funktion die Zelle relativ zur aktuellen Cursorposition des aktiven Fensters zurück. Hier bedeutet dann ZELLE(0;0) die Zelle, in der sich der Cursor befindet. (Ausnahme: Beim Parameter zum Makro-Befehl SPRINGE und SPRINGEWENN, siehe dort!)

Hinweis: Bitte beachten Sie, daß zuerst die Zeile und dann die Spalte kommt. Dies ist die Art, wie Tabellen normalerweise gelesen werden (entsprechend Matrizen!) - im Koordinatensystem würde dies jedoch Y,X bedeuten!

Beispiel:

ZELLE(-1;-1) ergibt den Wert, links über der aktuellen Zelle

Verwandte Funktionen:

ZELLEABS , INDIREKT , BEREICHABS

1.115 ZELLEABS(Zeile;Spalte)

ZELLEABS(Zeile;Spalte)

(Englisch: CELLABS)

Die Funktion ZELLEABS liest den Wert aus der mit Spalte und Zeile angegebenen Zelle aus und gibt ihn zurück.

Zeile und Spalte können beliebige positive Zahlen ab 1 sein (bei kleineren Werten erscheint #WERT)

Hinweis: Bitte beachten Sie, daß zuerst die Zeile und dann die Spalte kommt. Dies ist die Art, wie Tabellen normalerweise gelesen werden (entsprechend Matrizen!) - im Koordinatensystem würde dies jedoch Y,X bedeuten!

Tip: Diese Funktion kann gut in Makros benutzt werden, wenn man Zellen in Schleifen durchgehen möchte - siehe dazu das Beispiel bei SPRINGEWENN.

Tip: Soll ein Block angegeben werden, so kann auch die Funktion ZELLEABS benutzt werden, etwa: ZELLEABS(1;3):ZELLEABS(7;5 einfacher ist jedoch BEREICHABS.

Beispiel:

ZELLEABS(1;1) ergibt den Wert, der in der Zelle A1 steht.

CELLABS(3;2) ergibt den Wert der Zelle B3

Verwandte Funktionen:

ZELLE , INDIREKT , BEREICHABS

1.116 Finanz-Funktionen

Finanz-Funktionen

TurboCalc verfügt auch über eine ganze Reihe von finanzmathematischen Funktionen. Diese benutzen als Parameter meist folgende Angaben:

Kapital: Ein fester Betrag, der als Grundlage für eine Berechnung dient. (Auch Start- bzw. Endkapital genannt.)

Ratenhöhe: Betrag, der monatlich... (siehe Periode) bezahlt wird

Zinssatz: Zinssatz (also etwa 7.5% oder 0.075, jedoch nicht 7.5!), der eine Verzinsung beschreibt.

Zeitraum: Gibt den Zeitraum der Verzinsung an (also meist: Jahre). 1 entspricht in diesem Fall einem Jahr; 1/12 einem Monat, 1/360 einem (Bank-)Tag.

Periode: Bestimmt, wie oft pro "Jahr" die Verzinsung bzw. Ratenzahlung stattfinden soll. (1 bedeutet jährlich, 2 halbjährlich, 4 quartalsweise, 12 monatlich...) Dieser Parameter ist immer der letzte der Funktion und kann auch ausgelassen werden, dann wird 1 (also jährlich) angenommen.

Beispiele zu diesen Begriffen finden Sie bei den einzelnen Funktionen.

ENDKAPITAL(Kapital;Zinssatz;Zeitraum[;Periode])

LAUFZEIT(Kapital;Endwert;Zinssatz;[;Periode])

RATENENDKAPITAL(Ratenhöhe;Zinssatz;Zeitraum[;Periode])

RATENHÖHE(Endwert;Zinssatz;Zeitraum[;Periode])

RATENLAUFZEIT(Entwert;Ratenhöhe;Zinssatz;[;Periode])

STARTKAPITAL(Endwert;Zinssatz;Zeitraum[;Periode])

ZINSSATZ(Kapital;Endwert;Zeitraum[;Periode])

1.117 ENDKAPITAL(Kapital;Zinssatz;Zeitraum[;Periode])

ENDKAPITAL(Kapital;Zinssatz;Zeitraum[;Periode])

Diese Funktion berechnet das Endkapital eines einmalig angelegten Betrages. Einzugeben sind Anfangskapital, Laufzeit und Zinssatz.

Die übliche Fragestellung, die hiermit gelöst wertden kann, lautet: "Auf welchen Betrag wächst ein bestimmter Betrag, den ich heute bei meiner Bank anlege, in einer festgelegten Laufzeit ?"

1.Beispiel:

Bei der Geburt eines Kindes wird auf einem Sparbuch ein Betrag von DM 2500,- zu einem jährlichen Zinssatz von 3.5% angelegt. Auf welchen Betrag wächst das Sparguthaben bis zum 18. Lebensjahr des Kindes an ?

ENDKAPITAL(2500;0.035;18)

Ergebnis: Das Kapital ist nach 18 Jahren auf DM 4643,72 angewachsen.

2.Beispiel:

Ein Betrag von DM 10000,- soll für 18 Monate als Festgeld angelegt werden. Die Bank bietet einen Zinssatz von 7.5% bei halbjährlicher Verzinsung. Auf welchen Betrag wächst das Kapital an ?

ENDKAPITAL(10000;7.5%;1.5;2)

Ergebnis: Das Kapital ist nach 18 Monaten auf DM 11167,71 angewachsen.

1.118 LAUFZEIT(Kapital;Endwert;Zinssatz;[;Periode])

LAUFZEIT(Kapital;Endwert;Zinssatz;[;Periode])

Der Funktion "LAUFZEIT" berechnet, wie lange ein Betrag bei bekanntem Zinssatz angelegt werden muß, um einen gewünschten Endwert zu erreichen.

Beispiel:

Ihr Sohn möchte sich gern einen neuen Computer kaufen. Er hat DM 4000,- zusammengespart. Leider kostet das Gerät, das er sich wünscht DM 4500,-. Wie lange muß er das Geld auf seinem Sparbuch (Zinssatz 3%) anlegen, damit er den erforderlichen Betrag erreicht ?

LAUFZEIT(4000;4500;3)

Ergebnis: Ihr Sohn müßte (knapp) 4 Jahre warten, bis der Endwert erreicht ist.

1.119 RATENENDKAPITAL(Ratenhöhe;Zinssatz;Zeitraum[;Periode])

RATENENDKAPITAL(Ratenhöhe;Zinssatz;Zeitraum[;Periode])

Mit dieser Funktion können Sie ermitteln, welches Kapital Sie erreichen, wenn Sie über eine bestimmte Laufzeit in regelmäßigen Intervallen einen festen Betrag einzahlen. Vorausgesetzt wird ein über die gesamte Laufzeit gleichbleibender Zinssatz.

Beispiel:

Ein Auszubildender spart während seiner dreijährigen Ausbildungszeit monatlich einen Betrag von DM 200,-. Die Bank verzinst ihm seine Zahlungsreihe mit 6%, nach 3 Jahren möchte er sich von dem angesparten Kapital einen Gebrauchtwagen kaufen. Wieviel Geld wird er nach 3 Jahren für seinen Wagen zur Verfügung haben ?

RATENENDKAPITAL(200;6%;3;12)Ergebnis: Nach seiner Ausbildung kann er sich einen Gebrauchtwagen im Wert von DM 7867,22 kaufen.

1.120 RATENHÖHE(Endwert;Zinssatz;Zeitraum[;Periode])

RATENHÖHE(Endwert;Zinssatz;Zeitraum[;Periode])

Diese Funktion benötigen Sie, wenn Sie zu einem bekannten oder gewünschten Endwert einer Zahlungsreihe die Höhe der regelmäßigen Einzahlungen ermitteln wollen. Es wird davon ausgegangen, daß der Zinssatz während der Laufzeit konstant bleibt.

Beispiel:

Eine Familie möchte sich in 5 Jahren eine neue Wohnzimmereinrichtung im Wert von DM 8000 kaufen. Die Bank bietet ihnen eine Verzinsung der regelmäßigen Einzahlungen in Höhe von 6.5% an. Wie hoch ist die Sparrate bei monatlichen Einzahlungen ?

RATENHÖHE(8000;6.5%;5;12)

Ergebnis: Die Familie muß monatlich DM 113,20 auf das Sparkonto einzahlen.

1.121 RATENLAUFZEIT(Entwert;Ratenhöhe;Zinssatz;[;Periode])

RATENLAUFZEIT(Entwert;Ratenhöhe;Zinssatz;[;Periode])

Mit dieser Funktion können Sie ermitteln, wie lange Sie eine Zahlungsreihe aufrechterhalten müssen, um ein gewünschtes Endkapital zu erreichen. Zinssatz, Ratenhöhe und die Häufigkeit der Ratenzahlungen müssen bekannt sein.

Beispiel:

Der Traum Ihres Lebens ist ein sündhaft teurer Sportwagen im Wert von DM 80000. Da Sie das Geld nicht zur Verfügung haben beschließen Sie, monatlich einen festen Betrag von DM 500,- zu sparen, bis das Kapital inklusive der Zinsen von jährlich 6.5% den erforderlichen Betrag erreicht. Wie lange müssen Sie nun diese Zahlungsreihe aufrechterhalten ?

RATENLAUFZEIT(80000;500;6.5%;12)

Ergebnis: Sie müssen noch 10 Jahre sparen, bis Sie sich den Wagen leisten können (sofern die Wagen bis dahin nicht teurer geworden ist!).

1.122 STARTKAPITAL(Endwert;Zinssatz;Zeitraum[;Periode])

STARTKAPITAL(Endwert;Zinssatz;Zeitraum[;Periode])

Diese Formel gestattet es, zu einem bekannten oder gewünschten Endwert ein anzulegendes Startkapital zu berechnen. Der Zinssatz pro Jahr, die Laufzeit und die Häufigkeit der Verzinsungen müssen ebenfalls bekannt sein.

Sie benötigen diese Funktion zum Beispiel, wenn Sie in absehbarer Zeit eine größere Ausgabe tätigen werden. Es wäre unnötig, den gesamten erforderlichen Betrag bereitzuhalten. Legen Sie stattdessen nur den Betrag an, der inklusive der Zinsen den benötigten Endbetrag ergibt.

Endwert: Hierbei handelt es sich um das Kapital, das Sie am Ende der Laufzeit erreichen möchten.

Beispiel:

Sie wollen sich in 3 Jahren ein neues Auto kaufen. Sie wissen, daß der Wagen ziemlich genau DM 30000,- kosten wird. Welchen Betrag müßten Sie jetzt bei der Bank anlegen, um in 3 Jahren den erforderlichen Betrag zu erhalten, wenn Ihnen die Bank 7.5% Zinsen bietet ?

STARTKAPITAL(30000;7.5%;3)

Ergebnis: Sie müssen heute DM 24148,82 anlegen, um in 3 Jahren den erforderlichen Betrag zur Verfügung zu haben.

1.123 ZINSSATZ(Kapital;Endwert;Zeitraum[;Periode])

ZINSSATZ(Kapital;Endwert;Zeitraum[;Periode])

Diese Funktion berechnet den Zinssatz eines einmalig angelegten Betrages. Anfangs- und Endwert müssen bekannt sein, ebenso die Laufzeit und die Häufigkeit der Zinszahlungen.

Beispiel:

Sie haben vor 2 Jahren bei Ihrer Bank einen Betrag von DM 6000,- festgelegt. Am Ende der Laufzeit war der Betrag auf DM 6933,80 angewachsen. Da Sie den zugehörigen Vertrag verlegt haben, versuchen Sie nun selbst herauszufinden, wie hoch der vereinbarte Zinssatz war.

Eingabe: ZINSSATZ(6000;69938.80;2)

Ergebnis: Ihr angelegtes Kapital wurde mit 7,5% verzinst.

1.124 Sonstige Funktionen

Sonstige Funktionen

DATEIVORHANDEN(Datei)

DEMOVERSION()

INFO(Nr)

LETZTERFEHLER()

OBJEKTINFO(Name;Nr)

OBJID(Text)

REVISION()

SETZExxx(Bedingung;Wert1;Wert2[;Bezug])

TCFKT(TCLib;Offset;Num1;Num2;Text)

VERSION()

ZELLINFO(Nr[;Zelle])

1.125 DATEIVORHANDEN(Datei)

DATEIVORHANDEN(Datei)

(Englisch: FILEEXISTS)

Diese Funktion prüft, ob die angegebene Datei vorhanden ist und gibt entsprechnd WAHR (falls vorhanden) oder FALSCH zurück.

Datei ist ein Text mit dem Dateinamen

Hinweis: Obwohl dies eine Funktion ist, sollte sie nur im Zusammenhang mit Makros benutzt werden (siehe etwa Beispiel). Ansonsten sollte zumindest "automatisches Neuberechnen" deaktiviert werden. Wird dies nämlich als normale Funktion in der Tabelle eingesetzt, so wird bei jedem Neuberechnen geprüft, ob die Datei vorhanden ist, was das Neuberechnen sehr verzögern kann.

Anmerkung: Zum Prüfen werden keine Requester mit "Bitte Diskette... einlegen" angezeigt (falls ein Laufwerk nicht vorhanden ist, etwa bei "XXX:tes"") sondern in diesem Fall sofort FALSCH zurückgegeben.

Beispiel:

=SPRINGEWENN(NOT(DATEIVORHANDEN("test.tcd"));A20)

=LADEN("test.tcd")

Dieses Fragment prüft, ob die Datei "test.tcd" vorhanden ist. Falls ja, wird sie geladen, ansonsten an die Stelle A20 zur Fehlerbehandlung verzeigt.

1.126 DEMOVERSION()

DEMOVERSION()

Diese Funktion gibt WAHR zurück, falls die aktuelle TurboCalc-Version nur eine Demoversion ist. Ansonsten wird FALSCH zurückgegeben.

Dies kann dazu benutzt werden, in Makros die aktuelle TurboCalc-Umgebung zu prüfen.

1.127 INFO(Nr)

INFO(Nr)

Die Funktion gibt Informationen zu einigen interessanten Gebieten zurück.

Nr ist eine der folgenden Zahlen:

0 Anzahl geöffneter Tabellen

1 Anzahl geöffneter Tabellenfenster

2 Anzahl geöffneter und sichtbarer Tabellenfenster (die ausgeblendeten zählen hierbei nicht!)

3 freier Chip-Speicher

4 freier Fast-Speicher

5 freier Speicher gesamt.

1.128 LETZTERFEHLER()

LETZTERFEHLER()

(Englisch: LASTERROR)

Gibt die Fehlernummer des letzten Fehlers zurück. Vorallem innerhalb einer Fehlerbehandlungsroutine interessant. Näheres dazu beim Makrobefehl BEIFEHLER().

1.129 OBJEKTINFO(Name;Nr)

OBJEKTINFO(Name;Nr)

(Englisch: OBJECTINFO)

Die Funktion gibt Auskunkt über bestimmte Parameter eines Objekts.

Name: Der Name des Objekts. Existiert das Objekt nicht, erscheint die Fehlermeldung "WERT".

Nr: Bestimmt den Typ der Information, eine der folgenden Zahlen:

0 Name (da der Name angegeben werden muß, nicht sonderlich sinnvoll, sondern nur aus Vollständigkeit vorhanden)

1 Makro-Text

2 Makro-aktiv (WAHR/FALSCH)

3 Hintergrundfarbe (0=aus, 1=Farbe 0...)

4 Rahmen (0=aus, 1=normal, 2=3D, 3=3D-2)

5 aktiviert (WAHR/FALSCH)

6 X-Position

7 Y-Position

8 Breite

9 Höhe

1.130 OBJID(Text)

OBJID(Text)

Diese Funktion berechnet aus einem Text (der Länge 1 bis 4) die entsprechende Objekttyp-ID für den Befehl OBJEKT. Dies erlaubt eine bessere Objekttypauswahl

Text: Ein Text, aus dem die Objekttyp-ID berechnet wird, nur die ersten vier Zeichen sind relevant, Groß- und Kleinschreibung wird unterscheiden. (Für Insider: Die Funktion summiert die ASCII-Werte nacheinander auf, jeweils mit 256^3, 256^2 bzw. 256 multipliziert)

Beispiel

OBJID("TEXT") - siehe auch OBJEKT-Makro

1.131 REVISION()

REVISION()

Diese Funktion gibt die Revisionsnummer der aktuellen TurboCalc-Version zurück (also die Nummer nach dem Komma).

Dies kann dazu benutzt werden, in Makros die aktuelle TurboCalc-Umgebung zu prüfen.

1.132 SETZExxx(Bedingung;Wert1;Wert2[;Bezug])

SETZExxx(Bedingung;Wert1;Wert2[;Bezug])

Diese Funktionen erlauben es, die Formatierung der aktuellen oder einer anderen Zelle abhängig von der Bedingung zu setzen/ändern. So ist es sehr einfach möglich (ohne Makroprogrammierung), die Tabelle je nach eingegebenen Daten automatisch zu ändern.

Sie lauten genau:

SETZEAUSRICHTUNG ändert die aktuelle Ausrichtung (links-, rechtsbündig...)

SETZEFARBE ändert die Schriftfarbe

SETZEFORMAT ändert das Zahlenformat

SETZESTIL ändert den Schriftstil

(Bzw. auf Englisch: SETALIGNMENT, SETCOLOR, SETFORMAT und SETSTYLE)

Bedingung ist dabei ein Wahrheitswert. Ist er WAHR (bzw. ungleich 0), so wird die entsprechende Formatierung auf Wert1 gesetzt, ansonsten auf Wert2.

Wert1, Wert2 sind die Werte, auf die das entsprechende Stilmerkmal je nach Bedingung gesetzt wird:

Bei Ausrichtung: 0=Standard, 1=Links, 2=Rechts, 3=Zentriert

Bei Farbe: 0=Standardfarbe, 1=Farbe1... (siehe auch beim Makro-Befehl FARBEN)

Bei Format: 0=Standard, 1... gibt die Zahlenformatnummer an, siehe bei Makro-Befehl ZAHLENFORMAT

Bei Stil: 0=Normal, 1=Unterstrichen, 2=Fett, 4=Kursiv (auch mehrere durch Addition möglich) siehe Makro-Befehl SCHRIFT.

Bezug: Gibt die Zelle an, für die die Formatierung bestimmt werden soll. Fehlt dieser Eintrag, so wird die aktuelle Zelle genommen.

Alle Funktionen geben den Wert 0 zurück, falls die Änderung durchgeführt werden konnte. Ansonsten (etwa falls die Zielzelle leer ist!) wird eine entsprechende Fehlermeldung zurückgegeben.

Tip: Da diese Funktion etwas ändert und der Rückgabeparameter eigentlich uninteressant ist, kann er wie folgt sehr einfach ignoriert werden: Addieren Sie einfach diese Formel zu dem ursprünglichen Inhalt dazu (da ja 0 zurückgegeben wird, ändert dies nichts). Siehe auch erstes Beispiel unten.

Beispiel:

A1+SETZEFARBE(A1>=0;5;6) zeigt in der aktuellen Zelle den Wert der Zelle A1 an, und zwar in Farbe 5 falls er positiv ist und sonst in Farbe 6 (Dies kann also gut benutzt werden, um negative Einkünfte oder anderes besonders zu markieren).

SETZEFARBE(A1>0;5;6;A1) wie oben, jedoch wird die Farbe der Zelle A1 jetzt direkt geändert und es wird in der aktuelle Zelle einfach 0 zurückgegeben.

A1+SETZEAUSRICHTUNG(A1>=0;2;1) druckt den Inhalt von A1 aus und zwar rechtsbündig, falls er positiv ist und sonst linksbündig.

SUMME(A1:A10)+SETZESTIL(SUMME(A1:A10)>2000;3;4) berechnet das Ergebnis der Zellen A1 bis A10 und druckt es fett und unterstrichen, falls es größer als 2000 ist und sonst in kursiv.

1.133 TCFKT(TCLib;Offset;Num1;Num2;Text)

TCFKT(TCLib;Offset;Num1;Num2;Text)

(Englisch: TCFUNCTION)

Hiermit können externe Funktionen eingebunden und aufgerufen werden. Dies ist eine einfache Möglichkeit, TurboCalc um weitere Funktionen zu erweitern. Dazu werden TurboCalc-Libraries benutzt. Einzelheiten dazu finden im entsprechenden Abschnitt des Kapitels "Dateien".

TCLib: Dies ist der Name der TurboCalc-Library (als Text)

Offset: Gibt die Funktionsnummer an (somit können in einer Bibliothek mehrere Funktionen vorhanden sein).

Num1, Num2, Text: Sind die möglichen Parameter für die Funktion (Num1 und Num sind Zahlenwerte, Text ist ein Text)

Die genaue Funktionsweise, die Parameterbelegung sowie den Rückgabewert finden Sie in der Anleitung zu der jeweiligen TC-Bibliothek. Möchten Sie selbst eine eigene Bibliothek erstellen, so finden Sie Einzelheiten dazu im oben genannten Abschnitt über TCLibs.

Hinweis: Möchten Sie externe Makrobefehle aufrufen, so können Sie dazu TCMAKRO benutzen.

1.134 VERSION()

VERSION()

Diese Funktion gibt die Versionsnummer der aktuellen TurboCalc-Version zurück (also die Nummer vor dem Komma).

Dies kann dazu benutzt werden, in Makros die aktuelle TurboCalc-Umgebung zu prüfen.

1.135 ZELLINFO(Nr[;Zelle])

ZELLINFO(Nr[;Zelle])

(Englisch: CELLINFO)

Diese Funktion liefert gewünschte Informationen zu einer bestimmten Zelle.

Nr gibt den Bereich an, zu dem Informationen gewünscht werden.

Zelle bestimmt die Zelle, auf die sich die Informationen befinden Kann ausgelassen werden, dann wird die Zelle verwandt, in der sich die Formel befindet.

Nr kann einer der folgenden Werte sein:

0 Inhalt der Zelle (*immer* als Text)

1 Formel (als Text)

2 Datentyp (0: leere Zelle, 1: Fliesskommazahl, 2: Ganzzahl, 3: Datum, 4: Zeit, 5: Boolean, 6: Text, 7: Zellbezug 8: Fehler) (7 normalerweise nicht möglich!)

3 Zahlenformat

4 Textfarbe

5 Hintergrundfarbe

6 Muster (0-15)

7 Textstil (1: Unterstrichen, 2: Fett, 4: Kursiv - und Summen daraus)

8 Horizontale Ausrichtung (0: Std, 1: Links, 2: Mitte, 3: Rechts)

9 Vertikale Ausrichtung (0: Std, 1: Open, 3: Mitte, 2: Unten)

10 Rahmen (je zwei Bit (0-3, aus-max) je Seite -> 0-255!)

11 Schutzmerkmale (1: geschützt, 2: formel verbergen - Summe!)

12 Spaltenbreite (hierbei wird die Spalte genommen, in der sich die angegebene Zelle befindet)

13 Zeilenhöhe (hierbei wird die Zeile genommen, in der sich die angegebene Zelle befindet)

Beispiele siehe gleichnamige Beispiel-Tabelle!

1.136 Komplettes Inhaltsverzeichnis

BOGEN(Zahl)

COS(Zahl)

COSHYP(Zahl)

EXP(Zahl)

FAKULTÄT(Zahl)

GANZZAHL(Zahl)

LG(Zahl)

LN(Zahl)

LOG(Zahl)

LOG10(Zahl)

PI()

QUADRAT(Zahl)

REST(Zahl1;Zahl2)

RUNDEN(Zahl;Stellen)

SIN(Zahl)

SINHYP(Zahl)

TAN(Zahl)

TANHYP(Zahl)

VORZEICHEN(Zahl)

WINKEL(Zahl)

WURZEL(Zahl)

ZUFALLSZAHL()

Wahrheitswert-Funktionen

FALSCH()

ISTDATUM(Wert)

ISTFEHLER(Bezug)

ISTLEER(Bezug)

ISTTEXT(Wert)

ISTZAHL(Wert)

ISTZEIT(Wert)

NICHT(Wert)

ODER(Wert1;Wert2;...)

UND(Wert1;Wert2;...)

WAHR()

WENN(Bedingung; Wert1; Wert2)

XOR(Wert1;Wert2;...)

Text-Funktionen

Plus: +

CODE(Text)

GLÄTTEN(Text)

GROSS(Text)

GROSS2(Text)

HEX(Zahltext)

INTEXT(String;Muster[;Pos])

KLEIN(Text)

KOMPRIMIEREN(String[;Liste])

LÄNGE(Text)

LINKS(Text;Anzahl)

MITTE(Text;Zahl1;Zahl2)

RECHTS(Text;Anzahl)

SÄUBERN(Text)

SCHIEBENL(Text)

SCHIEBENR(Text)

SPIEGELN(Text)

TEIL(Text;Zahl1;Zahl2)

TEXT(Data[;Format])

WERT(Text)

WIEDERHOLEN(Text; Anzahl)

ZEICHEN(asc-code)

Datum-Funktionen

Plus: +, Minus: -

DATUM(Jahr;Monat;Tag)

DATWERT(Text)

HEUTE()

JAHR(Datum)

JETZT()

MONAT(Datum)

TAG(Datum)

WOCHENTAG(Datum)

ZEITWERT(Text)

Tabellen-Funktionen

ANZAHL(Bereich[;...])

ANZAHL2(Bereich[;...])

MAX(Bereich[;...])

MIN(Bereich[;...])

MITTELWERT(Bereich[;...])

PRODUKT(Bereich[;...])

STABW(Bereich[;...])

SUMME(Bereich[;...])

VARIANZ(Bereich[;...])

Datenbank-Funktionen

DBANZAHL(Datenbank;Spalte;Kriterien)

DBANZAHL2(Datenbank;Spalte;Kriterien)

DBMAX(Datenbank;Spalte;Kriterien)

DBMIN(Datenbank;Spalte;Kriterien)

DBMITTELWERT(Datenbank;Spalte;Kriterien)

DBPRODUKT(Datenbank;Spalte;Kriterien)

DBSTABW(Datenbank;Spalte;Kriterien)

DBSUMME(Datenbank;Spalte;Kriterien)

DBVARIANZ(Datenbank;Spalte;Kriterien)

Zell-Funktionen

#Bezug

@Tabelle;Bezug

AUSWAHL(Index; Wert1; Wert2; Wert3...)

BEREICHABS(Zeile;Spalte;Höhe;Breite)

BLOCKBREITE([Bereich])

BLOCKHÖHE([Bereich])

BLOCKX([Bereich])

BLOCKY([Bereich])

INDIREKT(Text)

SPALTENNUMMER([Bereich])

SVERWEIS(Wert;Bereich;Offset[;Exakt])

TABELLENNAME()

VERWEIS(Wert;Bereich[;Exakt])

WVERWEIS(Wert;Bereich;Offset[;Exakt])

ZEILENNUMMER([Bereich])

ZELLE(Zeile;Spalte)

ZELLEABS(Zeile;Spalte)

Finanz-Funktionen

ENDKAPITAL(Kapital;Zinssatz;Zeitraum[;Periode])

LAUFZEIT(Kapital;Endwert;Zinssatz;[;Periode])

RATENENDKAPITAL(Ratenhöhe;Zinssatz;Zeitraum[;Periode])

RATENHÖHE(Endwert;Zinssatz;Zeitraum[;Periode])

RATENLAUFZEIT(Entwert;Ratenhöhe;Zinssatz;[;Periode])

STARTKAPITAL(Endwert;Zinssatz;Zeitraum[;Periode])

ZINSSATZ(Kapital;Endwert;Zeitraum[;Periode])

Sonstige Funktionen

1.137 Index

DATUM(Jahr;Monat;Tag)

Datum-Funktionen

DATWERT(Text)

DBANZAHL(Datenbank;Spalte;Kriterien)

DBANZAHL2(Datenbank;Spalte;Kriterien)

DBMAX(Datenbank;Spalte;Kriterien)

DBMIN(Datenbank;Spalte;Kriterien)

DBMITTELWERT(Datenbank;Spalte;Kriterien)

DBPRODUKT(Datenbank;Spalte;Kriterien)

DBSTABW(Datenbank;Spalte;Kriterien)

DBSUMME(Datenbank;Spalte;Kriterien)

DBVARIANZ(Datenbank;Spalte;Kriterien)

DEMOVERSION()

E

ENDKAPITAL(Kapital;Zinssatz;Zeitraum[;Periode])

EXP(Zahl)

F

FAKULTÄT(Zahl)

FALSCH()

Finanz-Funktionen

G

GANZZAHL(Zahl)

GLÄTTEN(Text)

GROSS(Text)

GROSS2(Text)

H

HEUTE()

HEX(Zahltext)

I

INDIREKT(Text)

INFO(Nr)

INTEXT(String;Muster[;Pos])

ISTDATUM(Wert)

ISTFEHLER(Bezug)

ISTLEER(Bezug)

ISTTEXT(Wert)

ISTZAHL(Wert)

ISTZEIT(Wert)

J

JAHR(Datum)

JETZT()

K

KLEIN(Text)

KOMPRIMIEREN(String[;Liste])

L

LAUFZEIT(Kapital;Endwert;Zinssatz;[;Periode])

LETZTERFEHLER()

LG(Zahl)

LINKS(Text;Anzahl)

LN(Zahl)

LOG(Zahl)

LOG10(Zahl)

LÄNGE(Text)

M

Mathematikfunktionen

MAX(Bereich[;...])

MIN(Bereich[;...])

MITTE(Text;Zahl1;Zahl2)

MITTELWERT(Bereich[;...])

MONAT(Datum)

N

NICHT(Wert)

O

OBJEKTINFO(Name;Nr)

OBJID(Text)

ODER(Wert1;Wert2;...)

Operatoren

P

PI()

Plus: +

Plus: +, Minus: -

PRODUKT(Bereich[;...])

Q

QUADRAT(Zahl)

R

RATENENDKAPITAL(Ratenhöhe;Zinssatz;Zeitraum[;Periode])

RATENHÖHE(Endwert;Zinssatz;Zeitraum[;Periode])

RATENLAUFZEIT(Entwert;Ratenhöhe;Zinssatz;[;Periode])

WENN(Bedingung; Wert1; Wert2)

WERT(Text)

WIEDERHOLEN(Text; Anzahl)

WINKEL(Zahl)

WOCHENTAG(Datum)

WURZEL(Zahl)

WVERWEIS(Wert;Bereich;Offset[;Exakt])

X

XOR(Wert1;Wert2;...)

Z

ZEICHEN(asc-code)

ZEILENNUMMER([Bereich])

ZEITWERT(Text)

Zell-Funktionen

ZELLE(Zeile;Spalte)

ZELLEABS(Zeile;Spalte)

ZELLINFO(Nr[;Zelle])

ZINSSATZ(Kapital;Endwert;Zeitraum[;Periode])

ZUFALLSZAHL()