Macros

COLLABORATORS

	TITLE : Macros	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 17, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Macros

1.1 TurboCalc by Michael Friedrich

TurboCalc 4.0 - User Manual

Commands

This chapter is a complete overview of ARexx- and MACRO commands. For details about both languages, please refer to the "Macro/ARexx" chapter.

A command usually consists of a leading key-word followed by parameters (separated by semicolons) enclosed in brackets (e.g. DELETE(A1:C5)). If a command does not require any parameters, then the brackets may be omitted.

ARexx allows specification of parameters without brackets; wherein the parameters are separated by blanks as per normal ARexx practice.

Thus, the following are allowable:

FILL(0;A1:C5)

FILL 0 A1:C5

FILL(0)

FILL 0

(To find out why FILL can be invoked with one or two parameters, consult the description of the FILL command.).

If you use the notation without brackets, text can be entered without quotation marks if is does not contain blanks or quotation marks.

If you use brackets, the quotations marks are mandatory for strings!

MESSAGE("This is a message";"title")

MESSAGE "This is a message" "title"

MESSAGE message title

1.2 General

General

Before introducing the command set, here a few notes which apply to almost all commands:

Many commands refer to the menu item of the same or similar name. If so, the explanation of the respective command is limited to basic information. If you need further details about these commands you can refer to the sections which deal with the appropriate menu item.

Most of the commands require one or more of the following standard parameters:

Numbers

Booleans

Text

Cell/Range

Omission of Parameters

Omission of all Parameters

Return Value of Commands

Menu Commands

1.3 Numbers

Numbers

In most cases, a function expects an integer value as parameter; limits are determined by the respective command. You can specify them as integer, as real value (where allowed), or as formulae which return one of these values.

1.4 Booleans

Booleans

They are essential for yes/no -decisions. (For example in DBSORT for ascending/descending sorting). Here, the value 0 or FALSE corresponds to NO, any other value is interpreted as YES. (If this parameter is omitted, it corresponds to YES or this flag is ignored. For more details about interpretation, see the respective commands).

1.5 Text

Text

Here you may enter any text as long as it is enclosed in quotation marks (remember: in the ARexx mode they can be omitted). In addition to that, it is possible to use any of the text functions provided.

Note: As with functions, quotation marks which occur in the text must be doubled: The string a"b must be written as "a""b" and the text a""b must be written as "a""""b".

For ARexx mode the following exceptions apply:

Strings need not be enclosed in quotation marks, except when they contain blanks or quotation marks.

Then (if the text contains blanks) the text must be put in single as well as double quotation marks. e.g.:

MESSAGE '"This is text"'

or the whole command is enclosed by inverted commas (which makes reading a lot easier!):

'MESSAGE "This is text"'

Further information about this topic can be found in your ARexx Manual.

1.6 Cell/Range

Cell/Range

This is the last parameter of nearly every command and specifies the range on which command is to be applied. If it is omitted, the current cell/block will be used.

Range can be replaced by any cell specification (e.g. C5 or C13:F25). The functions CELL or CELLABS and the AT-function (@sheet;range) are also possible.

Note: With the AT-function (for detailed description, see cell functions), a macro command can be extended to any loaded sheet without prior activation. Thus, COLORS(3;;@database:C5) changes the color of cell C5 of the sheet named "database".

1.7 Omission of Parameters

Omission of Parameters

Many commands have parameters which are optional. In this case, the respective parameters may be left out (simply enter the final semicolon or the closing bracket). If there are no more parameters to follow, or if you want to specify only a certain number of them, you can close the brackets after the last parameter and leave out all other semicolons.

Example: Instead of COLOR(3;4;A1:C5) you may enter COLOR(3;;A1:C5). Thus, the background color will not be changed. If you enter COLOR(3), the current cell or the current block will be used and colored.

Note: For this reason, the last parameter of nearly every command is block information. If it is omitted, the command will refer to the current block - if it is given, it represents the desired range.

Within ARexx the omission of parameters can be substituted by a simple "_" or "#" (underscore or "hash" mark). (If parameters need to be specified at the end of a command, they may be left them out altogether).

COLOR 3 _ C5

COLOR 3 # C5

COLOR 3

1.8 Omission of all Parameters

Omission of all Parameters

Many commands allow the omission of all parameters. The execution of such a parameter-less command causes TurboCalc to open the corresponding requester, which enables the user to set the data (e.g. COLORS() for the display of the color-selection window).

Warning: These commands can only be used within a macro or ARexx program with the following limitations: To ensure true multi-tasking, TurboCalc starts its own task for screen display and window management (e.g. windows can be moved during a sorting process in the background). The commands need to hand over system control to the other task. Thus, the macro or ARexx command need not to wait until the user presses >OK< or >Abort<; the command will be finished at once with the status OK. Although the window is still visible to the user, the macro is continued with its next command. Please take note of this when programming macros!

1.9 Return Value of Commands

Return Value of Commands

Some commands can fail (e.g. LOAD or SELECTSHEET). An error message will be issued if the command is used interactively.

But if the corresponding command is called in the macro mode, this error message will be suppressed and instead of this message, a return value will be given back. Thus the success of the operation can be checked without the user of the macro receiving an error message. The return value will be written into the cell which contains the macro command. From there, the return value can usually be interrogated and if necessary, be handled accordingly. Normally TRUE will be returned if successful and FALSE otherwise. See the relevant commands for details.

Here is a simple example - in a real application "Could not open file" should be a little more detailed (which file and what can be done about it, etc) so that the user is not driven to despair by the computer.

A10 =LOAD("Example.TCD")

A11 =IFGOTO(#A10;A14)

A12 =MESSAGE("Could not open file!")

A13 =GOTO(A30)

A14 the real program begins here.

....

A30 =RETURN()

Please note the "#" in front of the cell A10. It ensures that the cell of the macro sheet which will be used and not the current worksheet's.

1.10 Menu Commands

Menu Commands

Some (few) command descriptions are restricted to "This corresponds to ...", these commands only call the corresponding menu item, if necessary the corresponding window (e.g. RECORDING to start macro recording) appears. These commands are of no practical importance in macro programming. Their main purpose lies in the menu and key definition, e.g:

ADDMENUITEM("Recording";"RECORDING")

KEY("CTRL-A";"RECORDING)

1.11 Block Commands

Block Commands

ADD(Data;Block])

CLEAR(Data;block])

COPY([Block])

CUT([Block])

FILL(Mode;[Block])

LANGUAGE(Mode;Block)

MADD(Block1;Block2;ResBlock)

MMUL(Block1;Block2;ResBlock)

MSUB(Block1;Block2;ResBlock)

PASTE([Block])

PASTEDATA(Mode;[Range])

REFERENCESHIFT(X;Y;Flag;Block)

REFERENCETYPE(Flag[;Block])

REMOVE(Data;[Block])

SERIES(Type;Increment;Columns;Range)

SORT(Ascending;Direction;Cell;Range)

TRANSPOSE([Block])

1.12 ADD(Data;Block])

ADD(Data;Block])

Inserts a block of the size of the current or specified block at the top left corner of it and moves the former contents to the right/down, or adds complete rows/columns as follows:

Data: Determines, how ADD is to be carried out:

0: Entire column(s) will be inserted.

1: Entire row(s) will be inserted.

2: A block will be inserted and the former contents (left of the block) will be shifted to the right.

3: A block will be inserted and the former contents (above the block) will be shifted downwards.

4-7: as for 0-3 but these do not adjust cell references (in functions, macros and names)!

All other values produce an error message!

Block: Determines the block to be inserted. If this information is missing, the current block or cell will be used.

If no parameter is specified this command corresponds to <Edit-Insert Cells> - The corresponding selection window will be displayed. (For further information, refer to the introductory "macro-commands" !)

Related Commands:

CUT , CLEAR , REMOVE

1.13 CLEAR(Data;Block])

CLEAR(Data;Block])

Deletes the specified information from the current or specified block:

Data: Determines, which information should be deleted: 0: all, 1: format, 2: formula (result of the last calculation is preserved), 3: values and formula (formatting is preserved).

All other values or the omission of this parameter produce an error message!

Block: Determines the block which should be affected. If this information is missing the current cell or block will be used.

If called without parameters, it corresponds to <Edit-Clear> - the respective selection-window will be displayed.

Related Commands:

CUT , REMOVE , ADD

1.14 COPY([Block])

COPY([Block])

Copies the current block or the block specified by block into the internal buffer (for PASTE)

Block: Determines the block to copy. If this information is missing, the current block is used.

If called without parameters, this command corresponds to the <Edit-Copy> menu item.

Related Commands:

CUT , PASTE , PASTEDATA

1.15 CUT([Block])

CUT([Block])

Cuts out the current block or the one specified by block. Cells are emptied and stores the block internally for a subsequent PASTE.

Block: Determines the block to cut. If this information is missing, the current block will be used.

If called without parameters, this corresponds to the menu-item <Edit-Cut>.

Related Commands:

COPY , PASTE , PASTEDATA

1.16 FILL(Mode;[Block])

FILL(Mode;[Block])

Fills the current or specified block as follows:

Mode: Determines, how the filling is to be carried out.

0: Right - the value in the left-most column of every row in the block is copied to all other cells to the right.

1: Down - values in the top row of the block are copied downwards column-by-column

2: Left - values in the right-most column determine the fill.

3: Up - the row at the bottom determines the fill pattern.

All other values produce an error message!

Block: Determines the block, which is to be filled. If this information is missing, the current block or cell will be used.

This command corresponds to the menu item <Edit-Fill-xxx>.

Related Commands:

COPY , PASTE , RECALC

1.17 LANGUAGE(Mode;Block)

LANGUAGE(Mode;Block)

This command corresponds to <Edit-Translate Formula> and converts the formula and macro commands of the block into the desired language:

Mode: 0 = German, 1 = English

Block: Determines the block, which is to be affected by LANGUAGE. If this information is missing, the current block or cell will be used.

1.18 MADD(Block1;Block2;ResBlock)

MADD(Block1;Block2;ResBlock)

This is a matrix arithmetic command. It calculates the sum of two matrices (ResBlock=Block1+Block2).

TurboCalc interprets a matrix from a block, the elements of the matrix correspond to the cells of the block.

Block1,Block2: These are the source blocks. They must have compatible dimensions (i.e. the blocks must have equal size). If Block1 & Block2 are not matched, an error message will be returned.

ResBlock: Only the top left corner is significant, the block size is determined from block1 and block2 (and the operation) - the size of ResBlock will be ignored.

Examples:

see example sheet "Arrays.TCD".

Related Commands:

MMUL , MSUB

1.19 MMUL(Block1;Block2;ResBlock)

MMUL(Block1;Block2;ResBlock)

This is a matrix command. It calculates the product of two matrices (ResBlock=Block1*Block2).

TurboCalc interprets a matrix from a block, the elements of the matrix correspond to the cells of the block.

Block1,Block2: These are the source blocks. They must have compatible sizes (that means that the width of the first block must be equal to the height of the second block). If Block1 & Block2 are unsuitable, an error message will be returned.

ResBlock: Only the top left corner is significant as the block size is determined by block1 and block2 (and the operation) - the size of ResBlock will be ignored.

Examples:

see example sheet "Arrays.TCD".

Related Commands:

MADD , MSUB

1.20 MSUB(Block1;Block2;ResBlock)

MSUB(Block1;Block2;ResBlock)

This is a matrix command. It calculates the difference between the two matrices (ResBlock=Block1-Block2).

TurboCalc interprets a matrix from a block, the elements of the matrix correspond to the cells of the block.

Block1,Block2: These are the source blocks. They must have compatible sizes (that means the two blocks must have equal size), otherwise an error message will be returned.

ResBlock: Only the top left corner is important here as the block size results from block1 and block2 (and certainly the operation) - the size of ResBlock will be ignored.

Examples:

see example sheet "Arrays.TCD".

Related Commands:

MMUL , MADD

1.21 PASTE([Block])

PASTE([Block])

Inserts the block that has been copied to the buffer by CUT or COPY at the current position/block or at the position given by Block. If a block is marked (or specified), it will be filled by either repeating or truncating the buffered data.

Block: Determines the block (or a cell), where the buffer is to be inserted. If it is missing, the current cursor position or the current block will be used.

If called without parameters, this command corresponds to the <Edit-Paste> menu item.

Note: If you want to insert the formatting or the contents only, you must use PASTEDATA.

Related Commands:

CUT , COPY , PASTEDATA

1.22 PASTEDATA(Mode;[Block])

PASTEDATA(Mode;[Block])

Inserts the block that has been copied to the buffer by CUT or COPY at the current position/block or at the position given by Block. If a block is marked (or specified), only this will be filled (the copied contents will be either repeated or truncated).

Mode: Determines which part of the copied contents is to be inserted:

0: Paste the formatting only.

1: Paste values only (the formulas will be removed).

2: Paste values and formulas - the formatting will be ignored.

Block: Determines the block (or a cell), where the buffer is to be inserted. If it is missing, the current cursor position or the current block will be used.

If called without parameters, this command corresponds to the menu-item <Edit-Paste only>.

Note: If you wish to paste the entire contents, you have to use the PASTE command.

Related Commands:

CUT , COPY , PASTE

1.23 REFERENCESHIFT(X;Y;Flag;Block)

REFERENCESHIFT(X;Y;Flag;Block)

Moves all references of the formulas within block by a specified offset.

X: Number of columns by which the formulas are to be moved. (Positive values: to the right, negative values: to the left.)

Y: Number of rows by which the formulas are to be moved. (Positive values: to the bottom, negative values: to the top.)

Flag: Determines which formulas are to be moved:

0 all references (default, if this parameter is omitted)

1 relative references

2 absolute references

Block: The block in which the formulas are to be checked and changed. If this parameter is omitted the currently marked block will be used.

If called without parameters this corresponds to <Command-References-Shift>. You will find more details in the description of that menu item.

1.24 REFERENCETYPE(Flag[;Block])

REFERENCETYPE(Flag[;Block])

Changes all references (e.g. C5 or C5) of the formulas within block and corresponding to Flag:

0 change all references to absolute (C5)

1 changes all references to relative (C5)

2 swap all references (default, if this parameter is omitted)

Block: The block in which the formulas are to be checked and changed. If this parameter is omitted, the currently marked block will be used.

If called without parameters this corresponds to <Command-References-Rel<->Abs>>. More details can be found in that menu item's description.

1.25 REMOVE(Data;[Block])

REMOVE(Data;[Block])

Removes all information of the current or specified block and moves the rest to the left/top, or removes entire rows/columns as follows:

Data: Determines how the removal should be carried out:

0: The whole column will be deleted

1: The whole row will be deleted

2: The block will be deleted and the rest (to the right of the block) will be shifted to the left.

3: The block will be deleted and the rest (below the block) will be shifted upwards.

4-7: as for 0-3 but these do not adjust cell references (in functions, macros and names)!

All other values produce an error message!

Block: Determines the block which should be affected by REMOVE. If this information is missing, the current block or cell is used.

If called without parameters, this command corresponds to the <Edit- Remove cells> menu-item - the appropriate selection-window will be displayed.

Related Commands:

CUT , CLEAR , ADD

1.26 SERIES(Type;Increment;Columns;Range)

SERIES(Type;Increment;Columns;Range)

Type: Specifies the type and the method for the creation of a data-series:

0: Arithmetical (default, if this parameter is omitted)

1: Geometric

2: Day

3: Weekday

4: Month

5: Year

Increment: Indicates the value by which the start value is incremented.

Columns: The series will be created by column if Columns is TRUE (or non-zero). If set to 0 or FALSE the series will be created row by row.

Range: Determines the range, which is to be filled by SERIES. If this information is missing, the current block will be used.

Called without parameters, this command corresponds to the <Data-Create Series> menu item - the appropriate selection-window will be displayed.

1.27 SORT(Ascending;Direction;Cell;Block)

SORT(Ascending;Direction;Cell;Block)

This command sorts a specified range based on the following options:

Ascending: Sorts the data-sets in ascending order if this parameter is non-zero or omitted. The value zero or FALSE causes sorting in descending order.

Direction: Sorts the data-range according to rows (parameter non-zero or omitted) or columns (parameter set to zero or FALSE)

Cell: Determines the key by which which rows/columns (depending on Direction) the sort is to be done. TurboCalc sorts according to the first row/column, if this parameter is left out.

Block: Determines the block which is to be sorted. If this information is missing, the current block/cell will be used.

If called without parameters, this command corresponds to the menu-item <Data-Sort Block> - the appropriate selection window will be displayed.

1.28 TRANSPOSE([Block])

TRANSPOSE([Block])

Transposes the current or specified block, that means that the cells will be mirrored at the diagonal from the top left to the bottom on the right. (the elements on the diagonal remain unchanged).

This corresponds to the usual mathematical method of transposing a matrix.

Block: Determines the block which is to be transposed. If this information is missing, then the current block will be used.

Note: The block must have square dimensions (that means height = width) - otherwise transposing is not possible and an error message will be returned.

If called without parameters, this command corresponds to the <Command-Transpose> menu item.

1.29 Formatting Commands

Formatting Commands

ALIGNMENT([Hor];[Vert];[Block])

BOX(Left;Right;Top;Bottom;[Block])

CHANGESTYLE(Num;[Block])

COLORS([Color1];[Color2];[block])

COLUMNTITLE(Title;[Cell])

COLUMNWIDTH(Width;[Block])

DEFAULTROWHEIGHT(Height)

FONT([Num];(Characterset];[Block])

FRAME(Left;Right;Top;Bottom;[Block])

FREEZE(Cell)

HIDE(Row;[Block])

NOTE(Note;[Cell])

NUMERICFORMAT(Format;[Block])

PATTERN(Number;[Block])

PROTECTION([Write];[Formula];[Block])

ROWHEIGHT(Height;Block)

ROWTITLE(Title;[Cell])

SHOW(Row;[Block])

STDFONT(Character set)

1.30 ALIGNMENT([Hor];[Vert];[Block])

ALIGNMENT([Hor];[Vert];[Block])

Determines the alignment of the current or specified block as follows:

Hor: 0=default, 1=left, 2=right, 3=centered (all other values or omitting of the parameter do not change the horizontal alignment).

Vert: 0=default, 1=top, 2=middle, 3=bottom (all other values or omitting of the parameter do not change the vertical alignment).

Block: Determines the block in which formatting is to be changed. If this information is missing, the current block will be used.

If called without parameters, this command corresponds to the menu-item <Format-Alignment> - the appropriate selection-window will be displayed.

Examples:

ALIGNMENT(3) centers all cells of the current block

ALIGNMENT(;1;A1:A10) The contents of cell A1 to A10 will be set to "top".

ALIGNMENT() a window appears, where the user may select the formatting.

Related Commands:

FONT , NUMERICFORMAT , COLORS ; FRAME

1.31 BOX(Left;Right;Top;Bottom;[Block])

BOX(Left;Right;Top;Bottom;[Block])

Determines the frame around the current or specified block as follows:

Left, Right, Top, Bottom: Specifies the frame thickness for the respective edge

0: off

1: thin

2: medium

3: thick

If one (or more) of these is omitted, the respective edge(s) will not change.

Block: Determines the block whose frame is to be changed. If this information is missing, the current block will be used.

This function corresponds to FRAME, but only the relevant formatting at the edge of the block will be changed.

Examples:

BOX(;;3;3;A10:C20) Block A10:C20 receives a top and bottom margin (the rest remains unchanged)

BOX(1;1;1;1;A1:A10) Cells A1 to A10 receive a thin frame on all sides.

Related Commands:

FRAME , COLORS , FONT , CHANGESTYLE , ALIGNMENT , NUMERICFORMAT

1.32 CHANGESTYLE(Num;[Block])

CHANGESTYLE(Num;[Block])

Changes the text style of the current or specified block as follows:

Num: Determines the style, which is to be applied for the current text: 1=underlined, 2=bold, 4=italic - combinations are possible by adding the appropriate features (e.g.: bold, italic and underlined = 2+4+1 = 7).

Note: In contrast to FONT, which sets a character-set, CHANGESTYLE toggles the characteristics of any font. If the style feature was active before, it will be switched off, otherwise switched on.

Block: Determines the block in which formatting is to be changed. If this information is missing, the current block will be used.

Examples:

CHANGESTYLE(3) Changes the style features bold and underlined of the current block.

CHANGESTYLE(5) switches off underlined again and switches on italic (so: bold and italic).

{b}Related Commands:

FONT , ALIGNMENT , NUMERICFORMAT

1.33 COLORS([Color1];[Color2];[block])

COLORS([Color1];[Color2];[block])

Determines the text- and background color of the current or specified block as follows:

Color1: Determines the text color, 0=standard, 1,2,3...are further colors, refer to <Format-Colors> (-1 or omission of the parameter leaves the color unchanged).

Color2: Determines the background color with the same options as Color1.

Block: Determines the block in which colors are to be changed. If this parameter is missing, the current block will be used.

If called without parameters, this command corresponds to the <Format-Colors> menu item - the appropriate selection-window will be displayed.

Examples:

COLOR(3;A1:A10) - contents of the cells A1 to A10 become blue (ref. to standard colors)

COLOR(;3) determines a new background color for the current cell.

Related Commands:

FONT , CHANGESTYLE , ALIGNMENT , NUMERICFORMAT , FRAME

1.34 COLUMNTITLE(Title;[Cell])

COLUMNTITLE(Title;[Cell])

Defines a Title for the column of the current or specified cell.

Title: Text to be assigned to the column title. If omitted, the existing title will be removed and the default (A, B, C, ...) re-assigned.

Cell: Specifies the columns for which the title is to be changed. If omitted, the current cell determines the column. If a block is specified, then the left-most column of the block is used.

If no parameters are specified, then this correspond to the <Format-Column-Title> menu option and a selection window will appear. (See the introduction on Macro Commands about this!)

Example:

COLUMNTITLE("Test") Assigns the title "Test" to the current column.

COLUMNTITLE(;C1) Removes the title from column C.

COLUMNTITLE("";C1) Blanks out the title on column C (a blank title).

Related Commands:

ROWTITLE

1.35 COLUMNWIDTH(Width;[Block])

COLUMNWIDTH(Width;[Block])

Determines the column width of the current or specified block as follows:

Width: The width of the column in 1/8 characters - if the value is too big or too small, it will be ignored! (Except 0, which selects the default width!)

Block: Determines the block for which the width is to be changed. If this information is missing, the current block will be used.

If called without parameters, this command corresponds to the <Format-Column Width> menu item - the appropriate selection-window will be displayed. (See the introduction on Macro Commands regarding this!)

Examples:

COLUMNWIDTH(80) - the current columns (i.e. all selected columns) are set to a width of 10 characters

COLUMNWIDTH(0;A1) - column A adjusts to the default width.

COLUMNWIDTH() - opens a window where the user can set the preferred width.

Related Commands:

ROWHEIGHT

1.36 DEFAULTCOLUMNWIDTH(Width)

DEFAULTCOLUMNWIDTH(Width)

Defines the width of all columns which have not had their width set explicitly.

Width: Determines the width in 1/8 characters; decimal values are permitted.

If no width is specified, then this is equivalent to the <Format-Column-Default Width> menu option - and a selection window will appear. (See the introduction on Macro Commands about this!)

Related Commands:

DEFAULTROWHEIGHT , COLUMNWIDTH

1.37 DEFAULTROWHEIGHT(Height)

DEFAULTROWHEIGHT(Height)

Defines the height of all rows which have not had their height set explicitly.

Height: Determines the height in characters; decimal values are permitted.

If no height is specified, then this is equivalent to the <Format-Row-Default Height> menu option - and a selection window will appear. (See the introduction on Macro Commands about this!)

Related Commands:

DEFAULTCOLUMNWIDTH , ROWHEIGHT

1.38 FONT([Num];(Characterset];[Block])

FONT([Num];(Characterset];[Block])

Determines the text style and the font of the current or specified block as follows:

Num: Determines the style, in which the text is to appear. 0=normal, 1=underlined, 2=bold, 4=italic - furthermore, combinations can be achieved by adding the appropriate styles (e.g.: bold, italic and underlined = 2+4+1 = 7) (-1 or omission of the parameter keeps the previous formatting)

Characterset: Specifies, which character set is to be used. This is text, which consists of the character-set name (with or without the filename-extension ".FONT") and a slash followed by the size (e.g. "topaz/8" or "Helvetica.font/13"). If this parameter is missing, the current character set will not be changed. Double quotation marks "" selects the standard character-set.

Block: Determines the block in which the font is to be changed. If this information is missing, the current block will be used.

If called without parameters, this command corresponds to the <Format-Font> menu item - the appropriate selection window will be displayed.

Examples:

FONT(3) reformats the text of the current blocks to bold and underlined.

FONT(;"Times/13";A1:A10) the contents of cell A1 to A10 will be redered in the "Times" character set, size 13 - the style remains unchanged.

FONT() opens a window, where the user can set formatting.

Related Commands:

COLORS , FRAME , CHANGESTYLE , ALIGNMENT , NUMERICFORMAT

1.39 FRAME(Left;Right;Top;Bottom;[Block])

FRAME(Left;Right;Top;Bottom;[Block])

Specifies a frame for the current or specified block.

Left, Right, Top, Bottom: specify the frame thickness for the respective edge:

0= off, 1= thin, 2= medium, 3= thick

If a parameter is omitted (or -1) the respective edge will not be changed.

Block: Specifies the block, which is to be framed. If this is missing, the current block will be used.

If called without parameters, this command corresponds to the <Format-Frame> menu item - the appropriate selection window will be displayed. (See the introduction on Macro Commands for important notes about this!)

Example:

FRAME(;;3;3;A10) - create a top and bottom frame to cell A10.

FRAME(1;1;1;1;A1:A10) - cells A1 to A10 get a thin frame on all sides.

Related Commands:

BOX , COLORS , FONT , CHANGESTYLE , ALIGNMENT , NUMERICFORMAT

1.40 FREEZE(Cell)

FREEZE(Cell)

Freezes all rows above Cell and all columns left to Cell.

Cell: Is the cell according to which the title range is set. If the current range is to be used, then specify CURRENTRANGE as the parameter.

If called without parameters, this command corresponds to <Format-Freeze>. Further details can be found there.

Note: Up to version 3.5, TurboCalc used the curent cursor position if no parameter was specified. To duplicate this behaviour, use the definition given in the third example below:

Examples:

FREEZE(C2) - column A and B and row 1 are frozen and thus constitute a kind of title.

FREEZE(A1) - unfreezes all previous frozen columns/rows.

FREEZE(CURRENTRANGE()) - freezes the range above and to the left of the current cell.

1.41 NOTE(Note;[Cell])

NOTE(Note;[Cell])

Attaches a Note to a cell.

Note: Text to be attached. An empty string will delete any existing note.

Cell: determines the cell to which the note is to be attached. The current cell is used if none is specified. Should a block be specified, then top-most, left cell is used.

If no parameters are given, then this corresponds to the <Format-Cell Note> menu option and the corresponding selection window appears.

Example:

NOTE("Test") Attaches the note "Test" to the current cell.

NOTE("";C1) Removes the note from cell C1.

1.42 HIDE(Row;[Block])

HIDE(Row;[Block])

Hides the current column or row:

Row: Determines whether the specified row or column should no longer be displayed:

0: Column, 1: Row

Block: Determines the block to which the command is to apply (depending on the value of Row only the row or column will be used). If this information is missing, only the current block or cell will be used.

If called without block-parameter, this command corresponds to the menu-items <Format-Column-Hide> or <Format-Row-Hide> - the appropriate selection-windows will be displayed.

Examples:

HIDE(0) hides the current column.

HIDE(1;C2:C3) hides rows 2 and 3.

1.43 NUMERICFORMAT(Format;[Block])

NUMERICFORMAT(Format;[Block])

Determines the numeric format of the current or specified block as follows:

Format: 0=standard, number 1 to 56 represent the respective numeric format (beginning with 0= standard) as follows:

Note: Starting with version 3.5 some new formats (41 to 56) have been added to TurboCalc. For compatibility reasons they have been added at the end.

0 standard

1 0

2 0.0

...

7 0.000000

8 0.000

9 0,000.00

10 000

11 0000

12 00000

13 0.0E+00

...

18 0.000000E+00

19 0%

...

25 0.000000%

26 $0

27 $0.00

28 $0.000

29 $0,000.00

30 DD.MM.YYYY

31 DD.MM.YY

32 DD.MMM.YYYY

33 DD.MMM YY

34 DD.MMM

35 MMM.YYYYY

36 MMM.YY

37 hh:mm:ss

38 hh:mm

39 h:mm:ss

40 h:mm

The new formats (TurboCalc3.5)

41 000000

42 0000000

43 00000000

44 000000000

45 0000000000

46 DD M YYYY

47 DD M YY

48 M YYYY

49 M YY

50 DD M

51 h:mm:ss am/pm

52 h:mm am/pm

53 [h]

54 [h]:mm

55 [m]

56 [m]:ss

Block: Determines the block for which formatting is to be changed. If this information is missing, the current block or cell will be used.

If called without parameters, this command corresponds to the <Format-Numeric Format> menu item - the appropriate selection window will be displayed.

Examples:

NUMERICFORMAT(0) the current cells are presented in standard format.

NUMERICFORMAT(2;A1:A10) sets the format "0.00" for cells A1 to A10

NUMERICFORMAT() opens a window, where the user can set preferred formatting.

Related Commands:

FONT , ALIGNMENT , COLORS , FRAME , BOX

1.44 PATTERN(Number;[Block])

PATTERN(Number;[Block])

Determines the background pattern of the current or specified block.

The Number parameter ranges from 0 to 15 and determines one of 15 possible background patterns.

The optional Block parameter determines the range whose background pattern is to be changed.

Without parameters this command corresponds to <Format-Pattern>.

Related Commands:

COLORS

1.45 PROTECTION([Write];[Formula];[Block])

PROTECTION([Write];[Formula];[Block])

Determines the protection of the current or specified block as follows:

Write: Determines if write access is to be granted for these cells. (0=yes, 1=no, -1=do not change.)

Formula: Determines if formulas within these cells are to be displayed or hidden (0=display, 1=hide, -1=do not change)

Block: Determines the block for which access privileges are to be changed. If this information is missing, the current block will be used.

If called without parameters, this command corresponds to the menu-item <Format-Protection> - the appropriate selection-window will be displayed.

Example:

PROTECTION(0) allow editing of the current cells, regardless of protection applied to the sheet.

PROTECTION(;1;A1:A10) prevents the display of formulas within the block A1:A10.

PROTECTION() opens a window, where the user can set the required protection.

Related Commands:

FONT , ALIGNMENT , NUMERICFORMAT , COLORS , FRAME , BOX

1.46 ROWHEIGHT(Height;Block)

ROWHEIGHT(Height;Block)

Determines the row height of the current or specified block as follows:

Height: Determines the height of the row according to the height of the current font, decimal values are allowed as well - if the value is too big or too small, it will be ignored! (Except 0, which selects the standard height!)

Block: Determines the block for which the height is to be changed. If this information is missing, the current block will be used.

If called without parameters, this command corresponds to the menu-item <Format-Row Height> - the appropriate selection-window will be displayed.

Example:

ROWHEIGHT(4) the current rows (that means all partly marked rows) are set to a height of 4 times the font height.

ROWHEIGHT(1.5;A1) row 1 receives a height of 1.5 font heights.

ROWHEIGHT() opens a window, where the user can set the required height.

Related Commands:

COLUMNWIDTH

1.47 ROWTITLE(Title;[Cell])

ROWTITLE(Title;[Cell])

Sets the title for the current row or the row specified by Cell.

Title: Text string containing the title to be assigned. If the parameter is omitted, then any existing title will be removed and the default (1, 2, 3, ...) is restored.

Cell: determines the row to which the title is to be assigned. If this is missing, then the current cell will be used. Should a block be specied, then the top-most row of the block is used.

If no parameter is given, then this is equivalent to the <Format-Row-Title> menu option - the appropriate selection window will be displayed. (See the introductory section on Macro commands about this!)

Examples:

ROWTITLE("Test") assigns the title to the current row.

ROWTITLE(;A7) removes the title on row 7.

ROWTITLE("";A7) blanks out the title on row 7

Related Commands:

COLUMNTITLE

1.48 SHOW(Row;[Block])

SHOW(Row;[Block])

Redisplays hidden rows or columns:

Row: Determines if TurboCalc should stop hiding the specified row or column:

0: Column

1: Row

Block: Determines the block for which the command is apply (depending on the value of Row only the row or column will be used). If this information is missing, only the current block or cell will be used.

If called without a second parameter, this command corresponds to the <Format-Show-Column> or <Format-Show-Row> menu items - the appropriate selection-windows will be displayed.

Examples:

SHOW(0) redisplays the current column.

SHOW(1;C2:C3) redisplays the rows 2 and 3.

Related Commands:

HIDE

1.49 STDFONT(Font)

STDFONT(Font)

This function determines the standard font for the current sheet.

Font: is text specifying the font (name/size, e.g. "Helvetica/13"). If this parameter is not specified, a selection window will appear.

If called without parameters, this command corresponds to the "Font" option on <Sheet-Settings> - the appropriate selection window will appear.

1.50 Cursor Control

COLUMN(Column)
CURRENTCELL()
CURSORDOWN(Num)
CURSORLEFT(Num)
CURSORRIGHT(Num)
CURSORUP(Num)
FIND(Text;Part;Case;Columns;Range)
GOTOCOLUMN(Column)
GOTOLINE(Line)
LASTCOLUMN()
LASTROW()
LINE(Line)
PING(Index)
PONG(Index)
SELECT([Block])
SELECTTOFRONT([Block])
SELECTWAIT()

1.51 COLUMN(Column)

COLUMN(Column)

Moves the cursor within current sheet column columns to the left or right (1 means one column to the right).

Related Commands:

GOTOCOLUMN , GOTOLINE , LINE , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.52 CURRENTCELL()

CURRENTCELL()

Shifts the visible part of the screen, so that the current cell or - block becomes visible (if it is already visible, this command has no effect).

This command corresponds to the <Command-Show Active Cell> menu item.

Related Commands:

SELECT

1.53 CURSORDOWN(Num)

CURSORDOWN(Num)

Moves the cursor by Num steps to the appropriate direction. If Num is missing, the cursor will be moved by one field.

Num: Can be any (positive) number. If it is missing, 1 will be assumed.

Related Commands:

LINE , COLUMN , GOTOLINE , GOTOCOLUMN , SELECT , CURSORLEFT , CURSORRIGHT , CURSORUP

1.54 FIND(Text;Part;Case;Columns;Range)

FIND(Text;Part;Case;Columns;Range)

Searches for the next appearance of text and locates the cursor in this cell.

Text: The text which is to be found. If this parameter is missing, the text entered before will be used, if there is not yet text available, an error message will appear. (See also the third example below)

Part: Determines, whether the cell must contain the exact text (0 or FALSE), or whether a partial match is sufficient (TRUE or non-zero). (If this parameter is omitted, TRUE will be assumed).

Case: Determines case sensitivity. Zero or FALSE means that even capitalization must match. TRUE or non-zero means that a simple match of the letters is sufficient. If this parameter is omitted, TRUE will be assumed.

Columns: Determines whether the text is to be searched by column (0 or FALSE) or by row (TRUE or non-zero) . If this parameter is omitted, a search by row will be assumed.

Range: If a cell is specified, it determines the starting position of search.

If a range is specified, the search will be started in the top left corner of it and the procedure is limited to this range.

If called without parameters, this command corresponds to the <Command-Find> menu-item.

In Macro Mode: If you use this command in the macro mode note that the cell in which this command is located receives the value TRUE if the search operation has been successful. If the search has failed, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see "return values of commands" at the beginning of the command overview).

Example:

FIND("hello")

searches for the string "hello", case is not significant and the text may also occur as a part of a string. (e.g. "Hello Alex").

FIND ("hello";;0)

Ditto, but now the word must be written in lower case letters ("hello Alex").

FIND(;)

resumes the last FIND process.

1.55 GOTOCOLUMN(Column)

GOTOCOLUMN(Column)

Moves the cursor within the current sheet to the specified column (1 represents the column at the extreme left of the sheet).

Related Commands:

GOTOLINE , COLUMN , LINE , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.56 GOTOLINE(Line)

GOTOLINE(Line)

Moves the cursor within the current sheet to the specified line (1 represents the line on the top of the sheet).

Related Commands:

GOTOCOLUMN , COLUMN , LINE , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.57 LASTCOLUMN()

LASTCOLUMN()

Moves the cursor, starting at its current position, to the last column which contains data (in this row).

Related Commands:

GOTOCOLUMN , GOTOLINE , LINE , LASTROW , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.58 LASTROW()

LASTROW()

Moves the cursor, starting at its current position, to the last row which contains data (in the current column)

Related Commands:

COLUMN , LASTCOLUMN , LINE , GOTOLINE , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.59 CURSORLEFT(Num)

CURSORLEFT(Num)

Moves the cursor by Num steps to the appropriate direction. If Num is missing, the cursor will be moved by one field.

Num: Can be any (positive) number. If it is missing, 1 will be assumed.

Related Commands:

LINE , COLUMN , GOTOLINE , GOTOCOLUMN , SELECT , CURSORRIGHT , CURSORUP , CURSORDOWN

1.60 LINE(Line)

LINE(Line)

Moves the cursor within the current sheet line rows up or down (1 means one row downwards).

Related Commands:

COLUMN , GOTOLINE , GOTOCOLUMN , CURSORUP , CURSORDOWN , CURSORLEFT , CURSORRIGHT , SELECT

1.61 CURSORRIGHT(Num)

CURSORRIGHT(Num)

Moves the cursor by Numsteps to the appropriate direction. If Num is missing, the cursor will be moved by one field.

Num: Can be any (positive) number. If it is missing, 1 will be assumed.

Related Commands:

LINE , COLUMN , GOTOLINE , GOTOCOLUMN , SELECT , CURSORLEFT , CURSORUP , CURSORDOWN

1.62 PING(Index)

PING(Index)

Stores the current cursor or block position (including sheet reference) in Index.

At some later time, the same position may be resumed by using PONG.

The position is remembered until TurboCalc exits, unless it has been over-written by a subsequent PING using the same Index.

Index: An integer from 0 to 9 (inclusive) which "place" is to be used to store the position. A previously stored position will be over-written.

Example:

see under PONG

Related Commands:

PONG

1.63 PONG(Index)

PONG(Index)

Returns the cursor to the position previously stored using PING.

Index: An integer from 0 to 9 (inclusive) which "place" is to be used to restore the position from a previous PING.

Note: The corresponding sheet will be brought to the front. Execution can be sped up within complex macros by using NOREFRESH to disable this function.

.

Example:

=PING(0) store current position in Index 0

=SELECT(C7) move to cell C7

=... miscellaneous operations

=PONG(0) return to previous position

Related Commands:

PING

1.64 SELECT([Block])

SELECT([Block])

Position the cursor as specified by Block (if it is a single cell) or marks the block and locates the cell cursor in the top, left corner of the block. If this range is not visible at the moment you call this command, the screen will be scrolled automatically.

Block: Determines the new cursor position. It can be a cell reference or a block (or, of course, a name, e.g. DATABASE)

Notes:

* The cursor can be positioned directly on a specific sheet by use of the AT (@) prefix for the block specification.

* An alternative method of switching to another sheet and displaying this in the foreground is the SELECTTOFRONT command.

* The synonyms SELECTCELL or SELECTRANGE have been introduced for ARexx as SELECT is a reserved key-word for ARexx, which would require it to always be placed in quotation marks.

* If called without parameters this command corresponds to the <Command-Goto> menu-item - the appropriate selection-window will be displayed. (See further notes about this in the introduction to Macro Commands!) This occurs asynchronously, so this command is not of practical use when awaiting user input. SELECTWAIT should be used instead.

Related Commands:

CURRENTCELL , SELECTWAIT , SELECTTOFRONT

1.65 SELECTTOFRONT([Block])

SELECTTOFRONT([Block])

This corresponds to SELECT, but the sheet containing the selected cell or block is brought to the front.

Positions the cursor as specified by Block (if it is a single cell) or

marks the block and locates the cell cursor in the top, left corner of the block. If this range is not visible at the moment you call this command, the screen will be scrolled automatically.

Block: Determines the new cursor position. It can be a cell reference or a block (or, of course, a name, e.g. DATABASE)

Notes:

* The cursor can be positioned directly on a specific sheet by use of the AT (@) prefix for the block specification.

* The selected sheet will be brought into foreground automatically. NOREFRESH may be used To speed up operations in complex macros.

* The synonyms SELECTCELL or SELECTRANGE have been introduced for ARexx as SELECT is a reserved key-word for ARexx, which would require it to always be placed in quotation marks.

* If called without parameters this command corresponds to the <Command-Goto> menu-item - the appropriate selection-window will be displayed. (See further notes about this in the introduction to Macro Commands!) This occurs asynchronously, so this command is not of practical use when awaiting user input. SELECTWAIT should be used instead.

Related Commands:

CURRENTCELL , SELECT , SELECTWAIT

1.66 SELECTWAIT()

SELECTWAIT()

This opens a selection window for the user to enter a block to highlight, analogous to the <Command-Goto> menu option. Unlike SELECT, this command will wait for the user to make a selection or to cancel the selection.

This command is therefore suited to allow a user to select a block on which subsequent macro commands are to operate.

Note: The selected sheet will be brought into foreground automatically. NOREFRESH may be used To speed up operations in complex macros.

In Macro Mode: The cell containing this command will be assigned a value of TRUE in macro mode if >OK< was pressed. If >Cancel< was selected, the cell is assigned a value of FALSE. This can be interrogated using IFGOTO.

In ARexx mode: >OK< doesn't return an error; otherwise ABORT is returned.

Related Commands:

SELECT

1.67 CURSORUP(Num)

CURSORUP(Num)

Moves the cursor by Num steps to the appropriate direction. If Num is missing, the cursor will be moved by one field.

Num: Can be any (positive) number. If it is missing, 1 will be assumed.

Related Commands:

LINE , COLUMN , GOTOLINE , GOTOCOLUMN , SELECT , CURSORLEFT , CURSORRIGHT , CURSORDOWN

1.68 Input Commands

Input Commands

AMIGAGUIDE(File;Command)

BEEP()

CHELP(Num;Link)

DELAY(Time)

DIALOGUE(Dialogue;Hook)

DLGRESUME(Flag)

FILEREQUEST([File];[Title];[Cell])

HELP(Num;File)

INPUT(Text[;Title];[Cell])

LISTREQUEST([Title];[Block])

MESSAGE(Text[;Title])

PUT(Contents[;Cell])

REQPARA(X;Y;Oktext;Aborttext;Time)

REQUEST(Text[;Title])

1.69 AMIGAGUIDE(File;Command)

AMIGAGUIDE(File;Command)

This provides for the display (certain parts) of AmigaGuide files and is used to show the AmigaGuide help of TurboCalc.

File: Is the name of the AmigaGuide file. You will either have to specify the complete path or to specify it relative to the directory in which TurboCalc exists. If this parameter is missing, "TurboCalc.guide" will be used.

Command: Is a command which is to be sent to the AmigaGuide viewer. The following commands are possible:

ALINK <name> Loads the paragraph <name> in a new window.

Note: In version 39 of the 'amigaguide.library' a new window will not be opened. (This constitutes a documented error of that version).

LINK <name> Loads the paragraph <name>.

RX <macro> Executes an ARexx-macro.

RXS <cmd> Executes an ARexx-string; e.g.:

'ADDRESS COMMAND DISPLAY <picture name>'

'ADDRESS COMMAND MORE <doc>'

CLOSE Closes the window (preferably, one which was opened with ALINK).

QUIT Closes the current AmigaGuide file.

LINK <name> will be mainly used to display a specific part of the AmigaGuide file. (<name> is the part located at @ENDNODE

The display takes place in the background (asynchronously). You can continue working with TurboCalc as you used to. Further, AMIGAGUIDE commands will also be possible. If the same file name is indicated, no new AmigaGuide viewer be loaded, but the indicated command will be transferred to the existing viewer. If a file name is used which had not been used before, a new viewer (i.e. a new window) will be started.

Note: amigaguide.library must be available to display the information, otherwise an error message will appear.

Example:

=AMIGAGUIDE("TurboCalc.guide")

Opens a window with the "contents" (main) of TurboCalc.guide.

=AMIGAGUIDE("TurboCalc.guide";"LINK save")

Displays the content of the paragraph (node) "save" in the window opened before.

=AMIGAGUIDE("Test.guide")

Opens another AmigaGuide file and displays "main".

Related Commands:

CHELP , HELP

1.70 BEEP()

BEEP()

Produces a short flash on the screen together with an audible signal. The audio-signal is limited to OS2.0 and higher (both can be switched on/off via Sound in Workbench Preferences).

BEEP is useful to indicate an error, if the user did something wrong.

1.71 CHELP(Num;Link)

CHELP(Num;Link)

Num: The number of the AmigaGuide help file (0=TurboCalc, 1=Index, 2=Menu, 3=Functions, 4=Macros, 5=Settings, 6=Tutorial, 7=Descriptions, depending on the current language).

Link: Is the name of the "node" within the help file, which is to be displayed. If omitted, then the "MAIN" node will be displayed.

Related Commands:

AMIGAGUIDE , HELP

1.72 DELAY(Time)

DELAY(Time)

Suspends macro-execution for the length of the Time parameter. This command is useful in connection with the start of external Amiga-DOS commands via the RUN-command.

Time: specifies the delay in 1/50 seconds. The maximum value is 30*50=1500, which corresponds to 30 seconds. This restriction was introduced to avoid prolonged blocking of TurboCalc execution. If you need more time, you can execute several DELAY commands in succession. If you omit the parameter time, the default delay of one second (i.e. the value 50) is assumed.

Note: The wait-state of TurboCalc can be interrupted in macro-mode only (with the menu-item <Macro - Stop Macro>, which is checked every 2 seconds). If DELAY was started via ARexx or directly (<F10>), you cannot interrupt it. You have to wait until the specified time has passed (which is the reason for the delay time restriction of this command).

1.73 DIALOGUE(Dialogue;Hook)

DIALOGUE(Dialogue;Hook)

Display a user dialogue (within a macro) which allows you to execute more complex requests (more complex than INPUT for example).

Dialogue:is a cell reference to the top left corner of a range containing the dialogue description (see below).

Hook: With this parameter you can determine a routine which will be called if the requester is changed (e.g. if a boolean gadget is clicked). The requester remains open in this case. (An application example is for example, the AutoFormat-Macro, in which - after having selected the corresponding formatting - this one will be executed immediately without having to close the requester).

Before calling the hook-routine the dialogue fields with the results will be updated.

The hook-routine must be terminated with DLGRESUME to allow further interaction. No other dialogue may be called from within this hook-routine.

Note: The hook-routine should be short and error-free.

See also the example macro `Macro_Dialogue2´ as well as `Macro_AutoFormat´.

A window will be opened containing the dialogue elements (e.g. text, gadgets, etc.). The macro will only be continued after a click on OK or on one of the other gadgets.

If this command is called from within a macro, the cell containing the macro command, will receive the return value (see REQUEST).

Clicking on OK (or <Return>) results in 0,

clicking on Abort (or <ESC>) results in -1,

clicking on another gadget (GAD-field) results in the corresponding parameter, see below!

Dialogue Range:

The following sheet contains 8 fields with the following inputs:

pos0 pos1 pos2 pos3 pos4 pos5 pos6 pos7

command X Y width height text para data

The entire row will not always be used, the unused portions being ignored. You will find details in the description of the commands.

X/Y/width/height: Determines the position and size of the elements. All specifications (positive) refer to 1/8 width or height of the character set (thus dialogues which are independent of the character set can be created).

Note: If you work with Topaz/8 (or with another 8 point character set), the values correspond exactly to the screen "pixels".

Width/height can be omitted (or be set to zero). Then the default width and height will be adopted. This is recommended in most cases.

Negative values (or 0) can be used for X/Y. Then this will indicate the modifications concerning the values previously specified. (Note: The change always goes to bottom left! Mixed values (pos/neg) can also be specified for X/Y) (See example 1).

Please note that 0 is interpreted as "negative" and means "not to change the position!" If you want to indicate the "first position top left" directly, you will have to use 1!

Command constitutes a number with a command:

0 END This ends the dialogue sheet! If text appears at the `text´ position, this text will be used as window-title!.

1 TEXT Inserts text (of pos5) to the X/Y position.

2 GAD Inserts a gadget with text (of pos5). Para (pos6) indicates the return value when clicking on this gadget (this will also close the dialogue).

3 BOOL Creates a bool with text (of pos5). Data (pos7) receives the select mode (0=off). This will be written back when ending the dialogue (if the return value is not negative).

4 CHOICE Similar to BOOL, but here you can select between various possibilities. The parameter mask (pos6) is used here to specify which of the other CHOICE fields will automatically be deselected when selecting this field: Mask is calculated as sum of 2^i, i represents the positions (from 1 onwards) of the CHOICE fields in this sheet (Note: TEXT will not be counted!!!)

5 STRING Creates a text field for the input of text. Default and return value in data (pos7). Para (pos6) contains the maximum width (in characters) of the text (300 characters maximum). This can be larger than the width of the text field. It will be scrolled!

6 MULT Not yet implemented in this version.

7 LIST Not yet implemented in this version.

8 FRAME A frame will be inserted (with width/height). Text (pos5) appears in the upper frame line.

9 SEPLINE A line will be inserted over the whole width at the current Y position.

10 OKABORT Inserts a sepline and two gadgets with the titles <OK> and <Abort>. This was inserted as a kind of "macro", because it occurs frequently.

Warning: Parameter data will only be changed in case the return value is greater than 0 (thus not in the case of abort!). Furthermore no "Recalculate" and no "Refresh" of the data will be executed in this case. But because this command will (almost) always be executed from a macro, this is no great consequence (to the contrary it will rather accelerate the macro). Requesting the data from the macro is still possible (and also "Recalculate" from the macro).

Example 1 (negative coordinates)

A B C D E F

1 8 8 0 0 This is text.

1 -32 -16 0 0 more text

2 0 -16 0 0 GAD1

2 -32 0 0 0 GAD2

0 Window-Title!

The first text is written at the position 8/8 and (2 rows) below, and 4 characters to the left the second text will be placed. Further 2 rows below, the gadget "GAD1" will be inserted with default width and then the gadget "GAD2" will be inserted in the same row with a distance of 4 characters.

Further examples may be found in the TurboCalc examples directory.

Title:

A B C D E

F

... (other commands)

0 SampleTitle

Compare this to the sample macros.

Related Commands:

DLGRESUME

1.74 DLGRESUME(Flag)

DLGRESUME(Flag)

This must be the last command of the DIALOGUE hook-routine.

Flag determines whether or not the dialogue-window should be closed:

0 The dialogue is closed and the macro execution is continued with the command after DLGRESUME. (If necessary, one can branch from there to the command after DIALOGUE.)

1 The dialogue is continued normally. This is the default which is assumed if the parameter is omitted.

NOte: If no dialogue is active at the time of DLGRESUME, then it is simply ignored.

Related Commands:

DIALOGUE

1.75 FILEREQUEST([File];[Title];[Cell])

FILEREQUEST([File];[Title];[Cell])

Opens a window with a file requestor. Macro execution is suspended until the user selects a file name, or cancels the operation.

This command is therefore suited for interactive use when the user is required to enter a file name to be used by subsequent macro commands.

File: A file name used as the default for the requester.

Title: Determines the title of the requester window. If omitted, then a default will be used.

Cell: Identifies the cell which is to receive the file name when selected. If omitted, the current cell of the current sheet is used. (i.e. Usually the window from which the macro was called, and not the macro window!)

In Macro Mode: The cell containing this command will be assigned a value of TRUE if the command completes successfully. If >Cancel< was selected, or the window closed using the close gadget, then a value of FALSE is assigned. This can be interrogated subsequently using IFGOTO.

In ARexx Mode: If >OK< was clicked, then no error is returned, otherwise the ABORT error.

1.76 HELP(Num;File)

HELP(Num;File)

Opens a text file (normally but not necessarily a help-file) and displays it:

Num: is the number of the help-text within the file (see below)

File: is the file to open. If this parameter is omitted, "TurboCalc.HELP" will be used instead.

The file is searched for in the current directory, in the TurboCalc directory (the one, in which TurboCalc itself is located) and in the S: directory.

File structure: the help-file is a pure "ASCII" text, where the lines are terminated with CR (that means <Return>, code 10). Lines should not exceed more than 60 characters, the rest will be truncated.

To combine different "text" into one file (to avoid an unnecessary number of files), the hash mark ("#") has been introduced as a separator. It has to be the first character in a line, after the CR. These "text" may be addressed with the parameter "num" (0 is the text up to the first "#", then follows 1...).

Note: This command can be used to show any text file. It is not limited to the TurboCalc help-file only (see second example).

Note: As of version 3, HELP without a second parameter (i.e. HELP(Num)) will attempt to open an AmigaGuide file. See also CHELP. (Both commands are equivalent without a second parameter.)

Examples:

HELP(0) shows the first help text. (From Version 3: this is the MAIN entry of TurboCalc.guide.)

HELP(0;"dfO:text")

shows the entire file "dfO:text" (up to the first # at the beginning of a line, but this does not usually occur in text!)

Related Commands:

AMIGAGUIDE , CHELP

1.77 INPUT(Text[;Title];[Cell])

INPUT(Text[;Title];[Cell])

Opens a window and displays Text in this window. Title determines the window title. The window has >OK< and >Abort< gadgets and a text gadget, which is provided for user input. A click on >OK< or pressing >Return< transfers this input to the current or specified cell.

Text: is normal text (in quotation marks or given as a text formula). It will be printed within the window - if it is longer than one line, it will be folded automatically (the tilde "~" can be used as a hyphenation hint: It will not be printed, except at the end of a line, where it changes into a dash "-". The "paragraph P" (ASCII-Code 182, produced by <ALT>+<P>) means "end of line". After this code, the text is continued on the next line).

Title: Determines the window title. A default title appears if you omit this parameter.

Cell: Determines the cell to receive the result after >OK<. If you omit this parameter, the current cell in the current window will be used (which is the window from which the macro is called and not the macro window with its source code).

Note: If you want to include cell contents in the text, use the function TEXT(cell) and concatenation by +. For further information, also refer to the third example of MESSAGE.

Note: For changing the window position etc. see RQPARA.

Note: If you use this command in the ARexx mode and if the specified text or cell contains blanks, make sure to put them in quotations marks. You can either enclose the whole command in inverted commas or the text in inverted commas and quotation marks: e.g. ’ "This is a text" ’.

In Macro Mode: If you use this command in macro mode, note that the cell in which this command is located receives the value TRUE, if you press the >OK< gadget. If you click on >Abort< or close the window with the close gadget in the top left corner, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example below).

In ARexx Mode: No error message is produced on >OK<, otherwise the error ABORT will be returned.

The current cell received the enetered data in both cases. This is processed according to the normal input rules. Therefore, numbers, dates, times booleans and text (including >"123< to enter numbers as text) may be entered.

Examples:

INPUT("Please enter your name";"Your Name";A1)

Opens a window with the title "Your Name" and displays "Please enter your name". The text gadget can now be filled with the user’s name. >OK< puts it to cell A1, >Abort< discards any input.

The following program excerpt implements a name requestor:

If the user does not enter valid text (that means: no alphanumeric data) or passes by with >Abort<, the macro main-routine is skipped. Otherwise, the message "Hello name" is displayed.

(Note the "#A10" notation for checking the cell A10 in the macro sheet (and not in the current cell)

A10 =INPUT("Please enter your name";"Personal data";A1)

A11 =IFGOTO(NOT(#A10 AND ISSTRING(A1));A20)

A12 =MESSAGE("Hello "+A1)

... main routine ...

A20 =RETURN()

/*ARexx-Example:*/

"Options Results

ADDRESS TCALC

INPUT ’"Please enter your name"’ ’"Your Name"’ C5

GETVALUE C5

SAY "Hello" Result

This example asks for the name, writes it to C5 and then displays "hello" + name.

(Please remember the single and double quotation marks, which enclose the text "Please enter your name" and "Your Name", because they contain blanks).

Instead of the INPUT line above, the following expression would also be possible (everything enclosed in single quotation marks):

’INPUT "Please enter name" "Your name" C5’

1.78 LISTREQUEST([Title];[Block])

LISTREQUEST([Title];[Block])

Opens a window containing a list of text elements in (the first column of) Block. Macro execution is suspended until the user selects an item from the list or cancels.

This command is therefore suited to allow a user to make an interactive selection to be used by subsequent commands.

Title: Determines the title of the window. If omitted, a default will be used.

Block: Determines the list entries. All cells from the first column of the block will be used. If no block is specified, then the current block of the current window is used. (This usually means the window from which the macro was called; not the window containing the macro!)

In Macro Mode: The current cell receives the number of the selected entry (0=first). If the selection was cancelled by the user, the cell will be assigned a value of -1.

1.79 MESSAGE(Text[;Title])

MESSAGE(Text[;Title])

Opens a window and displays Text in this window. Title determines the window’s title. The window has a >More< gadget. A click on this gadget or closing the window resumes macro execution.

Text: is normal text (in quotation marks or given as a text formula). It will be printed within the window - if it is longer than one line, it will be separated automatically (the tilde "~" can be used as a hyphenation hint: It will not be printed, except at the end of a line, where it changes into a dash "-". The "paragraph P" (ASCII-Code 182, produced by <ALT>+<P>) means "end of line". After this code, the text is continued on the next line).

Title: Determines the window’s title. A default title appears if you omit this parameter.

Note: If you want to include cell contents in the text, use the function TEXT(cell) and concatenation using "+", e.g. "contents of cell A1 is"+TEXT(A1)+"!". For further information, also refer to the third example).

Note: Use REQPARA to alter the window position, etc.

Examples:

MESSAGE("This is a test";"title")

Opens a window with the title "title" and displays "This is a test".

MESSAGE("Alongwordwith~separator").

The tilde determines at which point the text is separated by a hyphen.

MESSAGE("cell A1: "+TEXT(A1)+CHAR(182)+" Cell A2:"+TEXT(A2)+CHAR(182)+" A1+A2:"+TEXT(A1+A2)

CHAR(182) is the "paragraph P" (can also be entered directly by <Alt>-<P>), which forces TurboCalc to continue the text in a new line.

1.80 PUT(Contents[;Cell])

PUT(Contents[;Cell])

Writes the parameter contents to the specified or current cell. This corresponds to the direct input to the cell.

Contents: the input for the specified cell. It is treated like the direct input by using the keyboard - therefore, date, time, booleans and formulas can also be entered. The contents must be enclosed in quotation marks (If the text or the formula itself contains single quotation marks, you have to substitute them by two of the same kind (see below)).

Numbers and booleans can be entered directly as a parameter (without quotation marks), which avoids an unnecessary conversion into text. The same is valid for date and time values. Please note that these cannot be entered directly in formulas (only with VALUE, DATEVALUE...).

Cell: Determines the cell which is to be filled with contents. If it is missing, the current cursor position will be used.

Examples:

PUT("12";A1)

PUT(12;A1)

Both commands do the same. They put the numerical value 12 to cell A1

PUT(A1+1;A1)

Increases the contents of cell A1 by 1 (very useful in connection with IFGOTO, see there)

PUT("=A1+1";A2)

Puts the formula =A1+1 into cell A2

PUT("He said:""Hello""";A1)

Writes the text "He said:"Hello"" to cell A1 (With two quotation marks!!!).

PUT("""12";A1)

Writes the alphanumerical value 12 to cell A1 (that means the text "12, so that it is not interpreted as a number).

1.81 REQPARA(X;Y;Oktext;Aborttext;Time)

REQPARA(X;Y;Oktext;Aborttext;Time)

Determines the position, gadgets and the display time of the next requester (with MESSAGE, REQUEST, INPUT).

This command must be executed before the appropriate requester command.

X, Y: Indicates the X or Y coordinate of the window. Positive values give pixels beginning at the top left corner of the window. -1 centers the window in the respective direction (this will also be used if the parameter is omitted).

Oktext: This is the text which is to appear in the gadget "OK" (bottom left) or with messages in the gadget "CONTINUE" (centre). If this text is omitted or if "" is indicated, the default text will be used.

Aborttext: As per Oktext, but for "ABORT". It will be ignored for MESSAGE".

Note: The gadget always has the same width, so adjust the text correspondingly.

Time: Determines the time after which the window will be closed automatically. The number corresponds roughly to the time in seconds, 0 or omitting the parameter corresponds to the standard (i.e. no automatic closing). In case of the automatic closing a click on "abort" is simulated.

Note: The time will only be counted as long as the window is active. Moreover the time is not exact, but only an approximate value. These restrictions are no impediment to the intended purpose of the function, i.e. to briefly show a message.

Warning: REQPARA always refers to the next message/requester window. If an error message is displayed instead of MESSAGE, REQUEST ..., the current settings will also be used. But this will usually not cause great difficulties.

Example:

=REQPARA(;;"Cool";;20)

=MESSAGE("Hello")

The message "Hello" appears - the window will automatically be closed after a click on "Cool" (new name of the gadget "Continue") or after 20 seconds.

=REQPARA(0;0)

=MESSAGE("Hello")

The message "Hello" appears and the window will be opened in the top left corner of the screen.

1.82 REQUEST(Text[;Title])

REQUEST(Text[;Title])

Opens a window and displays Text in this window. Title determines the window's title. The window has >OK< and >Abort< gadgets.

Text: is normal text (in quotation marks or as a text formula). It will be printed within the window - if it is longer than one line, it will be separated automatically (the tilde "~" can be used as a hyphenation hint: It will not be printed, except at the end of a line, where it changes into a dash "-". The "paragraph P" (ASCII-Code 182, produced by <ALT>+<P>) means "end of line". After this code, the text is continued in the next line).

Title: Determines the window title. A default title appears if you omit this parameter.

Note: If you want to include cell contents in the text, use the function TEXT (cell) and the concatenation by "+", e.g. "contents of cell A1 is"+TEXT(A1)+"!". For further information, also refer to the third example of MESSAGE.

Note: For changing the window position etc. see REQPARA.

Note: If you use this command in the ARexx mode and if the specified text or cell contains blanks, make sure to put them in quotations marks. You can either enclose the whole command in inverted commas or the text in inverted commas and quotation marks: e.g. ' "This is text" '.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if you press the >OK< gadget. If you click on >Abort< or close the window with the close gadget in the top left corner, the cell receives the value FALSE. This can subsequently be tested with IFGOTO (see example below).

In ARexx Mode: For >OK< no error message is produced, otherwise the error ABORT will be returned.

Example:

REQUEST("This is a test";"Test-Window")

Opens a window with the title "Test-Window" and displays "this is a test". You can click on >OK< or pass by with >Abort<.

The following program extract implements a confirm requester:

(Note the "#A10" for checking the cell A10 in the macro sheet, and not in the current cell)

A10 =REQUEST("Are you sure ?";"Test-Macro";)

A11 =IFGOTO(NOT(#A10);A20)

... main routine ...

A20 =RETURN()

1.83 Load / Save

Load / Save

CLIPREAD([Block])

CLIPWRITE([Block])

CSVINSERT([Block];[Name];[Separator])

CSVLOAD([Name]);[Separator])

CSVSAVE([Name]);[Separator])

CSVSAVEBLOCK([Block];[Name];[Separator])

LOAD([Name])

LOADCONFIG()

PROCALCINSERT([Block];[Name])

PROCALCLOAD([Name])

SAVE([Name])

SAVEAS([Name])

SAVEBLOCK([Block];[Name])

SAVECONFIG()

SYLKINSERT([Block];[Name])

SYLKLOAD([Name])

SYLKSAVE([Name])

SYLKSAVEBLOCK([Block];[Name])

TCDINSERT([Block];[Name])

1.84 CLIPREAD([Block])

CLIPREAD([Block])

Reads from the clipboard and inserts the data into Block. If a cell is specified for Block, the data (as far as existing) will be continued to bottom right, otherwise at most the range indicated as block will be filled. If block is omitted, the currently marked block will be used.

The settings for the clipboard access can be determined with CLIPBOARD().

1.85 CLIPWRITE([Block])

CLIPWRITE([Block])

Writes the specified Block into the clipboard. If this parameter is missing, the current block will be used.

The settings for the clipboard access can be determined with CLIPBOARD().

1.86 CSVINSERT([Block];[Name];[Separator])

CSVINSERT([Block];[Name];[Separator])

Inserts the sheet Name at the position Block. The sheet has to be in CSV format - for further information refer to the appendix "Sheet Formats".

Block: Determines the top left corner where the data insertion is to start. If it is omitted, the insertion begins at the current cursor position.

If called without parameters, this command corresponds to the menu-item <Edit -Insert File - CSV >.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully. If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be tested subsequently with IFGOTO. For more details, see CSVLOAD, which is imilar, except for the first parameter.

Related Commands:

TCDINSERT , SYLKINSERT , PROCALCINSERT

1.87 CSVLOAD([Name];[Separator])

CSVLOAD([Name];[Separator])

Opens a new sheet (unless the current sheet is new and still empty) and loads the sheet Name. The sheet has to be in CSV format - for further information refer to the appendix "Sheet Formats".

Name: Specifies the name of the file to load. Specify the complete path (e.g. "DHO:TurboCalc/sheets/test.CSV"), if possible.

Separator: Specifies the character with which the CSV-format separates the different cell contents. If this information is missing, the semicolon ";" will be taken as default value. The character has to be put in quotation marks, e.g. "," or ";" or " " (blank). "T" represents the tab character, "S" is the blank (space).

Note: If only one parameter is specified, this will usually be the name. Exception: If the name consists of one character only, it will be interpreted as a separator and the requester for file-selection appears.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE, if loading was completed successfully. If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example under DBFIND). If called without parameters, this command corresponds to the <Project - Import -CSV> menu item.

Related Commands:

LOAD , SYLKLOAD , PROCALCLOAD

1.88 CSVSAVE([Name];[Separator])

CSVSAVE([Name];[Separator])

Saves the current sheet in CSV format in Name - for further information about the file format refer to the appendix "Sheet Formats".

Name: Specifies the name of the file in which the sheet is to be saved. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test
if possible.

Separator: Specifies the character with which the CSV-format separates the different cell contents. If this information is missing, the semicolon ";" will be assumed. The character has to be put in quotation marks, e.g. "," or ";" or " " (blank). "T" represents the tab character, "S" is the blank (space).

Note: If only one parameter is specified, this will usually be the name. Exception: If the name consists of one character only, it will be interpreted as a separator and the requester for file-selection appears.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if saving was completed successfully. If an error occurred during the process or the user aborted the save, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example under DBFIND).

If called without parameters, this command corresponds to the <Project - Export -CSV> menu item.

Related Commands:

SAVE , SAVEAS , SYLKSAVE

1.89 CSVSAVEBLOCK([Block];[Name];[Separator])

CSVSAVEBLOCK([Block];[Name];[Separator])

Saves the current or specified block of the current sheet as Name in CSV-format (see appendix "Sheet Formats"). If this name is missing, a file requester for name-selection will be displayed.

Block: Determines the block to save.

Name: Specifies the name of the file in which the sheet is to be saved. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.
if possible.

Separator: Specifies the character with which the CSV-format separates the different cell contents. If this information is missing the semicolon ";" will be used as the default value. The character has to be put in quotation marks, e.g. "," or ";" or " " (blank). "T" represents the tab character, "S" is the blank (space).

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if saving was completed successfully. If an error occurred during the process or the user aborted the save, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example under DBFIND).

If called without parameters, this command corresponds to the <Edit -Save Block as - CSV > menu item.

Related Commands:

SAVE , SAVEAS , CSVSAVE , SYLKSAVE

1.90 LOAD([Name])

LOAD([Name])

Opens a new sheet (unless the current sheet is new and still empty) and loads the sheet Name. The sheet must be in TurboCalc standard format.

Name: Specifies the name of the file to load. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.TCD"), if possible. A file requester will appear if this parameter is omitted.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully. If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project-Open> menu item.

Related Commands:

CSVLOAD , SYLKLOAD , PROCALCLOAD

1.91 LOADCONFIG([File])

LOADCONFIG([File])

This corresponds to the <Options-Config-Load> menu item and searches for the file "S:TurboCalc.CFG". If the file was found, the current settings are changed accordingly.

File: allows for the specification of an alternate configuration file. If File is specified as "", a file requester appears to allow selection of a configuration file. (If this parameter is omitted, then the default name as noted above, will be used.)

1.92 PROCALCINSERT([Block];[Name])

PROCALCINSERT([Block];[Name])

Inserts the named sheet at the position block. The sheet has to be in ProfessionalCalc file format - for further information refer to the appendix "Sheet Formats".

Block: Determines the top left corner where the data insertion is to start. If it is omitted, the insertion begins at the current cursor position.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Edit-Insert File-ProCalc> menu item.

For details see command PROCALCLOAD (which is identical, except of the first parameter).

Related Commands:

TCDINSERT , CSVINSERT , SYLKINSERT

1.93 PROCALCLOAD([Name])

PROCALCLOAD([Name])

Opens a new sheet (unless the current sheet is new and still empty) and loads the sheet name. The sheet has to be in the ProfessionalCalc file-format - for further information refer to the appendix "Sheet Formats".

Name: Specifies the name of the file to load. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.PCF"), if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project-Import-ProCalc> menu item.

Related Commands:

LOAD , CSVLOAD , SYLKLOAD

1.94 SAVE([Name])

SAVE([Name])

Saves the current sheet in TurboCalc standard format - for further information refer to the appendix "Sheet Formats".

Name: Specifies the name of the file to save. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.TCD"), if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project-Save> menu item.

Related Commands:

SAVEAS , CSVSAVE , SYLKSAVE

1.95 SAVEAS([Name])

SAVEAS([Name])

Saves the sheet in standard TurboCalc-format. In contrast to the SAVE command, this command always opens a file-requester for name-selection. If the parameter name is given, this will be taken as the default save-name, otherwise the name of the current sheet will be used.

Name: Specifies the default name for the file requester. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.TCD"), if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project-Saveas> menu item.

Related Commands:

SAVE , CSVSAVE , SYLKSAVE

1.96 SAVEBLOCK([Block];[Name])

SAVEBLOCK([Block];[Name])

Saves the current or specified Block of the current sheet as Name in standard TurboCalc-format (see appendix, "Sheet Formats"). If this name is missing, a file requester for name-selection will be displayed.

Block: Determines the block to save.

Name: Specifies the name of the file under which the sheet is to be saved. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/
if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if saving was completed successfully. If an error occurred during the process or the user aborted the save, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example under DBFIND).

If called without parameters, this command corresponds to the <Edit - Save Block as -TurboCalc> menu-item.

Related Commands:

SAVE , SAVEAS , CSVSAVE , SYLKSAVE

1.97 SAVECONFIG([File])

SAVECONFIG([File])

This corresponds to the <Options-Config-Save> menu item and saves the current settings as "S:TurboCalc.CFG" - this file is loaded after every start of TurboCalc or <Project-New> and the corresponding settings are adopted.

The current cell contents are not saved with this command.

File allows the specification of an alternate configuration file. If File is "", then a file requeter is presented to allow the user to select a file name. (If this parameter is omitted, then the default configuration file name as shown above, will be used.)

1.98 SYLKINSERT([Block];[Name])

SYLKINSERT([Block];[Name])

Inserts the sheet Name at the position Block. The sheet must be in SYLK-format - for further information refer to the appendix "Sheet Formats".

Block: Determines the top left corner, where the data insertion should start. If it is omitted, the insertion begins at the current cursor position.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Edit-Insert File-SYLK> menu item.

For details see command SYLKLOAD (which is identical, except of the first parameter).

Related Commands:

TCDINSERT , CSVINSERT , PROCALCINSERT

1.99 SYLKLOAD([Name])

SYLKLOAD([Name])

Opens a new sheet (unless the current sheet is new and still empty) and loads the sheet Name. The sheet has to be in SYLK-format - for further information refer to the appendix "Sheet Formats".

Name: Specifies the name of the file to load. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/test.SLK"), if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

Called without parameters, this command corresponds to the <Project - Import from -SYLK> menu item.

Related Commands:

LOAD , CSVLOAD , PROCALCLOAD

1.100 SYLKSAVE([Name])

SYLKSAVE([Name])

Saves the current sheet as Name in SYLK-format (see appendix, "Sheet Formats"). If this name is missing, a file requester for name-selection will be displayed.

Name: Specifies the name of the file under which the sheet is to be saved. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/
if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project - Export to -SYLK> menu item.

Related Commands:

SAVE , SAVEAS , CSVSAVE ,

1.101 SYLKSAVEBLOCK([Block];[Name])

SYLKSAVEBLOCK([Block];[Name])

Saves the current or specified block of the current sheet as Name{ui} in SYLK-format (see appendix, "Sheet Formats"). If this name is missing, a file requester for name-selection will be displayed.

Block: Determines the block to save.

Name: Specifies the name of the file under which the sheet is to be saved. Please specify the complete path (e.g. "DHO:TurboCalc/sheets/ if possible.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Project - Save Block as -SYLK> menu item.

Related Commands:

SAVE , SAVEAS , CSVSAVE , SYLKSAVE

1.102 TCDINSERT([Block];[Name])

TCDINSERT([Block];[Name])

Inserts the sheet Name at the position Block. The sheet must be in standard TurboCalc-format (TCD is the abbreviation of TurboCalcData)- for further information refer to the appendix "Sheet Formats".

Block: Determines the top left corner, where the data insertion should start. If it is omitted, the insertion begins at the current cursor position.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE if loading was completed successfully . If an error occurred during the process or the user aborted the load, the cell receives the value FALSE. This can be interrogated later. See "Return value of Commands" in the introduction on Commands.

If called without parameters, this command corresponds to the <Edit-Insert File-TurboCalc> menu-item.

For details see LOAD (which is identical, except of the first parameter).

Related Commands:

CSVINSERT , SYLKINSERT , PROCALCINSERT

1.103 Database Commands

Database Commands

CRITERIA([Range])

DATABASE([Range])

DBDELETE()

DBEXTRACT([Cell])

DBFIND([Cell])

DBMASK([Routine])

DBPREV(Cell)

DBSORT(Ascending;[Cell])

1.104 CRITERIA([Range])

CRITERIA([Range])

Determines the new range which serves as sort criterion (corresponds to <Data-Define criteria>).

If the parameter range is omitted, the currently marked block will be used.

Further information about the creation of search criteria can be found in the "Database" section of the manual.

Examples:

CRITERIA() - determines the current block as new search criteria.

CRITERIA(A1:C10) - the block A1 to C10 is the new search-criteria-range.

Related Commands:

DATABASE , DBFIND , DBEXTRACT , DBDELETE , DBSORT

1.105 DATABASE([Block])

DATABASE([Block])

Determines the new database range (corresponds to <Data-Define Database>).

If the parameter range is omitted, the currently marked block will be used.

For details about database ranges, refer to chapter seven "Database".

Examples:

DATABASE() - defines the current block as a new database range

DATABASE(A1:C10) - block A1:C10 is the new database range

Related Commands:

CRITERIA , DBFIND , DBEXTRACT , DBDELETE , DBSORT

1.106 DBDELETE()

DBDELETE()

Deletes all data sets which correspond to the current criteria (corresponds to <Data-Delete>).

For details, refer to the "Database" section.

Example:

DBDELETE() - deletes all lines which correspond to the criteria.

Related Commands:

DATABASE , CRITERIA , DBFIND , DBEXTRACT , DBSORT

1.107 DBEXTRACT([Cell])

DBEXTRACT([Cell])

Copies all data sets which correspond to the current criteria (corresponds to <Data-Extract>).

Cell determines the position to which the data is to be copied. Please make sure that there is enough space left around the copy-position, so that nothing will be overwritten by chance! If the parameter cell is omitted, inserting will be performed from current cursor-position onwards.

For details, refer to the "Database" section.

Example:

DBEXTRACT(A100) copies all matching datasets to A100.

Related Commands:

DATABASE , CRITERIA , DBFIND , DBDELETE , DBSORT

1.108 DBFIND([Cell])

DBFIND([Cell])

Searches for the next match of the database (corresponds to <Data-Find>).

Cell determines the start position of the search-process (A1 represents: from the beginning).

If the parameter cell is missing, the current cursor position serves as start position.

In Macro Mode: If you use this command in macro mode, the cell in which this command is located receives the value TRUE, if the search was completed successfully . If an error occurred during the process, the cell receives the value FALSE. This can be tested subsequently with IFGOTO (see example below).

TurboCalc won't open a window to indicate that the search has failed in macro mode. If you want to provide the user with this information, you have to program such a window in your macro.

For details, refer the "Database" section.

Examples:

DBFIND(A1) searches for the first column to match the current criteria.

DBFIND() resumes this process, after the first matching item has been found (TurboCalc stops searching after the first match).

A10 =DBFIND(A1)

A11 =IFGOTO(#A10;A14)

A12 =MESSAGE("No matching items found !")

A13 =GOTO(A30)

A14 ... main routine ...

...

A30 =RETURN()

Related Commands:

DATABASE , CRITERIA , DBEXTRACT , DBDELETE , DBSORT

1.109 DBMASK([Routine])

DBMASK([Routine])

Opens a window to allow the viewing, editing and addition of data records (corresponds to the <Data-Mask> menu item). See the "Database" section for more details.

The window is opened asynchronously. i.e. As the user enters data, macro commands continue execution. Synchrnous operation can be forced using the following parameter.

Routine: indicates the cell containing an address from which DBMASK continues execution (i.e. when the user clocks on "Close").

* set DBMASK with the parameter pointing to the cell after the next.

* command after DBMASK: RETURN()

* cell after next contains the command to be executed after DBMASK completes.

Example:

A10 =DBMASKE(A12)

A11 =RETURN

A12 ...

Related Commands:

DATABASE , CRITERIA , DBFIND , DBEXTRACT , DBSORT

1.110 DBPREV(Cell)

DBPREV(Cell)

This command works in the same way as DBFIND, but the search direction is from bottom to top!

Cell: Determines the position at which the search is to be started. If this specification is missing, the current position of the cursor will be used. If this specification is outside the database range, the start will be in the last database row.

DBPREV returns the following value (i.e. the cell in which the macro command is set has the following content):

TRUE: a row has been found, this is marked.

FALSE: no match

Note: In contrast to DBFIND, DBPREV never displays a message (e.g. "no data range found"), not even in interactive mode.

1.111 DBSORT(Ascending;[Cell])

DBSORT(Ascending;[Cell])

Sorts all data sets in ascending (or descending) order according to the current column (corresponds to <Data-Sort Range>):

Ascending: Sorts the data-sets in ascending order if this parameter is not 0 or omitted. The value zero or FALSE causes sorting in descending order.

Cell determines the column according to which the sort will be performed. If the parameter cell is missing, the sorting will be performed according to the current cursor position. (If the column, which is used for the sorting process, lies outside a database range, the DBSORT command will be ignored!)

For details, refer to the "Database" section.

Example:

DBSORT(A1) -sorts according to the first column (if the database range starts in column A).

Related Commands:

DATABASE , CRITERIA , DBFIND , DBEXTRACT , DBDELETE

1.112 Options

Options

AUTOSAVE(Flag)

AUTOCORRECT()

AUTOFILL()

CLIPBOARD(Unit;Separators;Quotation marked)

DEFPUBSCREEN(Screen)

DISPLAY(Title;Grid;Toolbar;Formulas;Zero)

FORMFEED(Flag)

GLOBALFLAGS()

KEY(Key;Command)

LOCALE(NF1;NF2;DF;Currency;CPrefix;CSuffix;Inch)

OPENFLAGS(NoAutoMacro;Hide;Password)

PAPERFORMAT()

POSTSCRIPT()

PRINTFORMAT(LM;RM;UM;BM;Style;Header;H-Text;Footer;F-Text;Title;Grid)

PRINTRANGE(Activate;[Range])

PROTECTFLAGS()

PSFONTLIST()

REFRESH(Mode)

SHANGHAI(Mode)

SETTINGS()

SETTINGS.PREVIEW()

SETTINGS.RECALC()

SETTINGS.SCREEN()

SETTINGS.UNDO()

SHEETFLAGS(Maxwidth;Maxheight;Calculation;Return;Direction;Icons)

SHEETSETTINGS()

SMARTREFRESH(Flag)

TOOLBAR()

1.113 AUTOCORRECT()

AUTOCORRECT()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.114 AUTOFILL()

AUTOFILL()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.115 AUTOSAVE(Flag)

AUTOSAVE(Flag)

This corresponds to <Sheet-Global Settings> "Saving" and calls the window for parameter setting.

If flag is set, the current sheet will instead, be saved.

1.116 CLIPBOARD(Unit;Separators;Quotation marked)

CLIPBOARD(Unit;Separators;Quotation marked)

Here you can determine the parameters for the clipboard. If there is no parameter specified, this will correspond to "Clipboard" under <Sheet-Global Settings>. A detailed description of parameters can be found in that section.

Unit: Number from 0 to 255.(Default:0)

Separator: Kind of separator, number from 0 to 4 (Std: 0)

0 tab

1 comma

2 semicolon

3 blank

4 several blanks

Quotation marks: Number from 0 to 2: (Std: 0)

0 never without quotation marks

1 quotation marks only if necessary

2 always quotation marks

If a parameter is omitted or specified as -1, the current setting will not be changed. The clipboard will then be called with the commands CLIPREAD and CLIPWRITE.

Examples:

CLIPBOARD(1)

changes to clipboard No. 1 - other parameters will not be changed.

CLIPBOARD(;0;0)

Tab, never quotation marks - the clipboard unit will not be changed.

1.117 DEFPUBSCREEN(Screen)

DEFPUBSCREEN(Screen)

(from OS2.0 on)

Determines the current DefaultPublicScreen, that means the screen on which the messages as well as further windows are to appear. Further information can be found in the user manual.

Screen: Is the name of the PublicScreen (e.g. "WorkBench" for the WorkBench). If a name is not given, the TurboCalc screen will be used (if it is opened).

Examples:

DEFPUBSCREEN() selects the TurboCalc screen as new DefaultPublicScreen.

DEFPUBSCREEN("WorkBench") The workbench screen will again be the DefaultPublicScreen.

Related commands:

SHANGHAI

1.118 DISPLAY(Title;Grid;Toolbar;Formulas;Zero;CursorMode)

DISPLAY(Title;Grid;Toolbar;Formulas;Zero;CursorMode)

If called without parameters this command corresponds to the menu item <View-Display> which allows the setting of display-parameters of a TurboCalc-window.

Note: All display-options only affect the current window - they can (and must) be set separately for every window!

Title: determines if the column and row titles are to be displayed.

Grid: determines if a grid for visual cell separation is to be displayed.

Toolbar: determines if the toolbar is to be displayed.

Formulas: It determines whether the values or the formulas behind them are to be displayed.

Zero: determines if zero-values in cells is to be displayed or left out.

CursorMode: determines, whether the cursor is to be drawn as a frame or as a block.

For each parameter you can specify the following options:

0 or FALSE: parameter not activated.

1 or TRUE: parameter activated.

2 toggle. Turns the status of a parameter into its opposite (activated becomes not activated and vice versa)

-1 or omission of parameter: The setting remains unchanged.

Example:

DISPLAY(;;;2)

This command toggles between displaying values or formulas as cell contents.

1.119 FORMFEED(Flag)

FORMFEED(Flag)

This function determines, whether a printed page should be finished with the control code Formfeed (12) or some linefeeds.

Normally, TurboCalc prints out several linefeeds to finish the page according to the page's length. This allows for an unrestricted paper-format.

If you set FORMFEED to "on" you force TurboCalc to print out one single formfeed instead of the linefeeds. This requires the setting of page-length directly in your printer-setups in addition to the setting within TurboCalc.

Flag: value 0 or FALSE means: no formfeed required.

value 1 or TRUE means: formfeed required.

Note: Page-orientated printers (e.g. the ones which use the laser or inkjet technology) may require this option.

Warning: In spite of having activated this option, you have to determine the paper-format again within TurboCalc. The program needs this information to calculate the number of printable lines.

1.120 GLOBALFLAGS()

GLOBALFLAGS()

This corresponds to <Sheet-Global Settings> "@bMisc".

1.121 KEY(Key;Command)

KEY(Key;Command)

This macro allows the binding or rebinding of any keys to TurboCalc commands. Possible application is in TurboCalc.STD (for determining Short-Cuts which should always be available) or the macro commands to be able to call for example certain macro programs by pressing one key only.

Key: Is text which indicates a key (see table below).

Command: Constitutes the command (macro or ARexx command) which will be executed as soon as the key is pressed. This command can also be omitted to unbind the command from the key. (For "Standard-F2", use the command EDIT and for "Standard-F10"; COMMAND)

The key can begin with the following "qualifier key" (and end with "-"):

LSHIFT-RSHIFT-SHIFT-S-

CAPS-

CTRL-C-

LALT-RALT-ALT-A-

LAMIGA-RAMIGA-AMIGA-

L... or R.. stand for exactly the right or left key. The expressions without L or R mean that it will be sufficient to press either key. (If Lxxx as well as Rxxx is specified, this corresponds to xxx, it will then be sufficient to press one shift key).

Several qualifier-keys can be combined.

Subsequently (or in case of no qualifier-keys being used, also by themselves) one of the following keys as follows:

F1 ... F10

ESC

TAB

RETURN

DEL

BS

HELP

UP

DOWN

LEFT

RIGHT

And for the numeric block:

NUM0..NUM9

ENTER

NUMA..NUMG (the remaining keys of the numeric block).

Or a normal key can be added (e.g. ctrl-a, alt-r)

Warning: Only use normal keys (i.e. those which can be accessed without qualifier keys. Use "shift-1" to specify "!" on most keyboards.

LSHIFT-F1 is the key F1, if the left shift-key is pressed.

SHIFT-RALT-F1 Any shift-key as well as the right Alt-key must be pressed if F1 is pressed.

Examples:

KEY("F1";"EDIT) the key F1 has now the function which is normally the function of F2 (Turn on editing)

KEY("F1") recreate the default function of F1

KEY("CTRL-X";"CUT") to assign the block command to the keys Ctrl-X/C/V

KEY("CTRL-C";"COPY") (can normally be accessed by r.Amiga)

KEY("CTRL-V";"PASTE")

KEY("CTRL-A";"MACROPLAY(A1)") if Ctrl-A is pressed, the macro will be executed from position A1.

1.122 LOCALE(NF1;NF2;DF;Currency;CPrefix;CSuffix;Inch)

LOCALE(NF1;NF2;DF;Currency;CPrefix;CSuffix;Inch)

If you call this command without any parameter, it corresponds to the <Sheet-Settings> menu item and its "Locale" option, allowing setting of locale-specific parameters.

If called with parameters, you can change these options directly and thus avoid the selection requester.

NF1 (NumericFormat1): determines the separator between decimals in front and after the decimal-point:

0: comma (1,23)

1: decimal point (1.23)

NF2 (NumericFormat2): determines the separator of the thousand figures in numbers.

0: comma (1,200)

1: decimal point (1.200)

2: apostrophe (1’200)

3: dash (1-200)

4: blank (1 200)

DF: determines the date format:

0: point (30.09.93)

1: dash (30-09-93)

2: slash (30/09/93)

Currency: determines the currency symbol:

0: DM

1: Dollar

2: user definition as follows below

3: Pound

4: Yen

5: FF (Franc)

6: ÖS (Schilling)

7: SFr (Franks)

CPrefix is the text which is to be located in front of a currency amount.

CSuffix is the text which is to be placed after a of currency amount.

(One or both can be omitted as well, by setting their respective parameter to "".

Inch: TRUE (or 1) changes to "inch", FALSE (or 0) changes to "cm".

This flag is for the use in TurboCalc’s requester only. The macro-commands (e.g. PRINT) are not affected by this flag, their values must always be entered in centimeters (to ensure macro compatability).

1.123 OPENFLAGS(NoAutoMacro;Hide;Password)

OPENFLAGS(NoAutoMacro;Hide;Password)

NoAutoMacro: If this flag is set to TRUE, any AUTOMATIC-Macro will not be executed. This is useful if you want to open a sheet for editing without the macro being executed.

Hide: If this flag is selected (TRUE), the window of the sheet which is to load will not be opened, but will immediately be marked as "hidden". (It can later be displayed by <Sheet-Show Sheet>). This function can be used by macros to reload sheets in the background (almost imperceptibly to the user).

Password: Stores a 'default password' which will be used in case the sheet is password protected. This default password will be retained until the next modification. You can turn off the default password with "" (empty text). (If this default password does not match to that of the sheet, a requester will appear in which the user will be asked for the correct password.)

This can be very helpful if you want to open password-protected sheets from a macro. The user will then only have to be asked once for the password and this one can then be used for all sheets.

Note: By using this option the effect of the password protection will naturally be limited and security reduced. You should take this into account when using this command. (Example: The user inputs his password in the macro, this one will be taken as the default password. The user leaves the room for a short time. Now anybody can interrupt the macro and load any sheet protected with the same password!)

1.124 PAPERFORMAT()

PAPERFORMAT()

This corresponds to "Paper Format" (Sheet-Settings).

1.125 POSTSCRIPT()

POSTSCRIPT()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.126 PRINTFORMAT(LM;RM;UM;BM;Style;Header;H-Text;Footer;F-Text;Title;Grid)

PRINTFORMAT(LM;RM;UM;BM;Style;Header;H-Text;Footer;F-Text;Title;Grid)

This command may be used to determine the layout of printing. If you call it without any parameters, it corresponds to the <Sheet-Settings> menu's <Print Layout> option.

LM: left margin in centimeters.

RM: right margin in centimeters.

UM: upper margin in centimeters.

BM: bottom margin in centimeters.

Style: 0=Pica, 1=Elite, 2=Condensed

Header: Determines if a header is to be printed. TRUE or non-zero values mean "Yes" whereas any other value suppresses the header.

Headtext: is the text, which is to be printed as header if the parameter "Header" is TRUE. The text must be quoted.

You can also replace this parameter by a formula, which returns text (e.g. LEFT...).

Footer: Determines if a footer is to be printed. TRUE or non-zero values mean "Yes" whereas any other value suppresses the footer.

Foottext: is the text which is to be printed as footer if the parameter "Footer" is TRUE. The text must be quoted.

You can also replace this parameter by a formula, which returns a text (e.g. LEFT...).

Title: Determines if row or column titles are to appear on the printout. TRUE or non-zero means "Yes" while any other value suppresses the titles.

Grid: Determines if a grid is to be printed to separate cells. TRUE or non-zero values mean "Yes" while any other value suppresses the grid.

Examples:

PRINTFORMAT(2;2;2;2;0;TRUE;"Header";FALSE;"Footer";FALSE;FALSE)

Margins 2 cm all round.

Style: Pica.

Header on (text:"Headline").

Footer off (but text defined as:"Footer").

No titles nor grid.

PRINTFORMAT(;;;;;;;TRUE)

Adds the footer and leaves the rest unchanged.

PRINTFORMAT(;;;;;;;;"page%P")

Changes the footer into the text "page" followed by the current page number.

1.127 PRINTRANGE(Activate;[Range])

PRINTRANGE(Activate;[Range])

Determines the range which is to be printed. Corresponds to the <Sheet-Settings menu's <Print Range>.

Activate: Determines if the printrange-option is to be switched on or off. 0 or FALSE turns off the printrange and earmarks the entire sheet for print-out.

Range: Determines the range which is to be printed (if the parameter Activate has been set to a non-zero value). If this parameter is omitted, the current block will be marked.

Examples:

PRINTRANGE() marks the current block for printing.

PRINTRANGE(0) switches off printrange.

PRINTRANGE(;A1:C5)

PRINTRANGE(TRUE;A1:C5) - both determine block A1 to C5 as the new printrange.

1.128 PROTECTFLAGS(Active;Password)

PROTECTFLAGS(Active;Password)

This command allows for setting protection attributes for the current sheet.

Active:

0 or FALSE de-activates protection.

1 or TRUE activates protection.

-1 toggles protection.

Password: If the status can only be changed by providing a password, then it must be given here as text. If no passowrd is required, then it may be omitted -- it would be ignored anyway.

If no password was given when required, or the wrong password was given, then the error ERR_PROTECTED (41) will be returned.

Without parameters, this command is equivalent to "Protection Flags" (<Sheet-Settings>).

Note: The other options can only be set via the settings window.

1.129 PSFONTLIST()

PSFONTLIST()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.130 REFRESH(Mode)

REFRESH(Mode)

Toggles between screen-refresh on or off, or forces a new refresh.

Mode:

0 (or FALSE) switches screen refresh off.

1 (or TRUE) switches screen refresh on.

The value 2 forces a refresh. This is advisable if the refresh has been previously switched off and is now set to "on" again. TurboCalc does not automatically do a refresh in this case, so you have to add this function manually.

Note: This function turns off screen refresh as well as automatic formula calculation. This characteristic speeds up macro execution, as it avoids the screen-redraw after every input (see second note). If you need the current cell results a RECALC must be performed.

Note: This command only affects macro and ARexx commands. All normal inputs or command calls cause TurboCalc to refresh the screen immediately.

In spite of this, the refresh should be switched on at the end of a every macro with REFRESH(TRUE), followed by REFRESH(2) to get the current and correct cell contents.

Example:

Following macro extract accelerates macro-execution:

A10 =REFRESH(0)

... insert the routine be accelerated here.

A20 =REFRESH(1)

A21 =RECALC()

A22 =REFRESH(2)

A23 =RETURN()

1.131 SETTINGS()

SETTINGS()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.132 SETTINGS.PREVIEW()

SETTINGS.PREVIEW()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.133 SETTINGS.RECALC()

SETTINGS.RECALC()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.134 SETTINGS.SCREEN()

SETTINGS.SCREEN()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.135 SETTINGS.UNDO()

SETTINGS.UNDO()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.136 SHANGHAI(Mode)

SHANGHAI(Mode)

(for OS2.0 and higher!)

Switches the so-called Shanghai-mode on or off:

Mode: Is the sum of the following two settings:

1: SHANGHAI (all new opened workbench-windows appear on the TurboCalc screen)

2: POPPUBSCREEN (determines that the TurboCalc screen is put to front whenever a requester appears on the TurboCalc screen.)

SHANGHAI(3) activates both options.

1.137 SHEETFLAGS(Maxwidth;Maxheight;Calculation;Return;Direction;Icons)

SHEETFLAGS(Maxwidth;Maxheight;Calculation;Return;Direction;Icons)

Determines the behaviour of each individual sheet. If called without any parameter, it corresponds to menu item <Sheet-Settings>.

Maxwidth: maximum number of columns in the sheet (between 50 and 18728 cells!).

Maxheight: maximum number of rows in the sheet (between 50 and 9999999 cells!).

Calculation: Determines the mode of recalculation (automatic/manual). TRUE or non-zero switches on automatic recalculation after any change within the sheet. The value 0 or FALSE switches to manual recalculation.

Return: Determines, if the cursor is to be moved after pressing <Return>. TRUE or non-zero means "Yes" while any other value suppresses cursor-movement after <Return>.

Direction: Specifies the direction to which the cursor is to be moved: 0=down, 1=up, 2=left, 3=right.

Icons: Determines if an icon should be created when the sheet is saved. TRUE or non-zero means "Yes" while any other value suppresses the creation of icons.

1.138 SHEETSETTINGS()

SHEETSETTINGS

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.139 SMARTREFRESH(Flag)

SMARTREFRESH(Flag)

Switches SmartRefresh on or off. If it is activated, all windows will be saved temporarily in intermediate memory ("backing store") as soon as they are overlaid by other windows or graphic objects. This process reduces the number of necessary screen-redraws and speeds up TurboCalc execution accordingly. The disadvantage to this is the increased memory requirement for the backing store.

Flag:

0: SmartRefresh is turned off completely.

1: SmartRefresh is enabled for sheet-windows.

2: SmartRefresh is enabled for chart-windows.

3: SmartRefresh is enabled for all windows.

If called without parameters, this command corresponds to the <Sheet-Global Settings> menu item's "Screen Refresh".

1.140 TOOLBAR()

TOOLBAR()

This macro command opens the corresponding settings window. Use this command with ADDMENUITEM or KEY to set up a menu structure or to bind keys as required.

1.141 Menu Commands

Menu Commands

ADDMENUITEM(Name;Command;[Menu;Item])

ADDMENUSUB(Name;Command;[Menu;Item;Sub])

ADDMENUTITLE(Name;[Menu])

CHARTMENU()

DELMENUITEM(Menu;Item)

DELMENUSUB(Menu;Item;Sub)

DELMENUTITLE(Menu)

NEWMENU()

SHOWMENU()

1.142 ADDMENUITEM(Name;Command;[Menu;Item])

ADDMENUITEM(Name;Command;[Menu;Item])

Inserts the new menu item at the position Menu, Item and assigns the Command to it.

Name: Is a string which is to be displayed as the menu item name.

Command: Is the command to be executed by selection of the respective menu item. You may use any valid TurboCalc macro or ARexx command.

Menu: Is the menu number to which the menu item is to be added (0=start, ...). If title is missing, it will be automatically set to the end of the menu list. By replacing this parameter by blank text ("") you specify this menu-item as "not-selectable".

Item: Is the menu-item number to which the menu item is to be added (0=start, 1=after first position...). If "item" is missing it will be set to the end of the item list.

For further information and a more detailed example, please refer to NEWMENU.

Example:

ADDMENUITEM("Macro 1";"MACROPLAY(Macro1)";1)

This command adds the menu item "Macro 1" to the end of the Edit-menu. If you select this menu item, a macro named "Macro 1" will be started immediately!

ADDMENUITEM("~")

If the tilde ("~") is specified as text, it corresponds to a separator. This can be used to separate menu groups from each other.

1.143 ADDMENUSUB(Name;Command;[Menu;Item;Sub])

ADDMENUSUB(Name;Command;[Menu;Item;Sub])

Inserts the new sub-menu item name at the position menu, item, sub and assigns the command to it.

Name: Is a string, which is to be displayed as sub-menu item text.

Command: is the command which is to be executed when the sub-item is selected. Any valid TurboCalc macro or ARexx command may be used.

Menu: Is the menu number to which the sub-menu item is to be added (0=start, ...). If menu is missing, it will be set to the end of the menu list.

By replacing this parameter by blank text ("") you specify this sub-menu item as "not-selectable".

Item: Is the menu-item number to which the sub-menu item is to be added (0=start, ...). If item is missing it will be set to the end of the item list.

Sub: Is the sub-menu item number to which the submenu-item is to be added (0=start, ...). If sub is missing, it will be set to the end of the sub-item list.

For further information and a more detailed example, please refer to NEWMENU.

1.144 ADDMENUTITLE(Name;[Menu])

ADDMENUTITLE(Name;[Menu])

Inserts the new menu name at the position Menu.

Name: Is a string, which is to be displayed as the menu title.

Menu: Is the menu number to which the menu item is to be added (0=start, ...). If Menu is missing, it will be set to the end of the menu list.

For further information and a more detailed example, please refer to NEWMENU.

1.145 CHARTMENU()

CHARTMENU()

The currently-defined menu is established as the new chart menu. NEWMENU is executed automatically. This command is used to establish menus for individual chart windows.

1.146 DELMENUITEM(Menu;Item)

DELMENUITEM(Menu;Item)

Deletes the menu item at the position Title, Item.

Menu: Specifies the menu number which is to lose the menu item.

Item: specifies the item which is to be deleted (0=first...)

Example:

DELMENUITEM(0;1) deletes the menu item <Project - Open> in the standard menus.

1.147 DELMENUSUB(Menu;Item;Sub)

DELMENUSUB(Menu;Item;Sub)

Deletes the menu item at the position title, item, sub.

Menu: Specifies the menu number which is to lose the sub-menu-item.

Item: Specifies the menu-item number which is to lose the sub-menu-item...).

Sub: The sub item which is to be deleted (0=first...)

Example:

DELMENUSUB(0;5;0) deletes the menu item <Project - import -Procalc> in the standard menus. Project - import -Procalc> in the standard menus.

1.148 DELMENUTITLE(Menu)

DELMENUTITLE(Menu)

Deletes the menu title at the position Title.

Menu: Specifies the menu number which is to be deleted. (0=first, ...)

Example:

DELMENUTITLE(0) deletes the entire <Project> menu in the standard menus.

1.149 NEWMENU()

NEWMENU()

Deletes the current menu and allows the creation of a complete new menu with ADDMENUTITLE, ADDMENUITEM and ADMMENUSUB

Note: Normally, menus are changed immediately after a NEWMENU... or DELETEMENUTITLE... . This can be very time-consuming when you create a completely new menu. Therefore, the "Renewal" of the menu structure is switched off after NEWMENU until the first SHOWMENU is executed. Therefore, SHOWMENU must always be the last command of a menu definition.

(If you subsequently want to add or remove some menu items, the use of SHOWMENU is not mandatory, unless NEWMENU has been called previously).

Example:

NEWMENU

ADDMENUTITLE file

ADDMENUITEM open... LOAD

ADDMENUITEM "Save as..." SAVEAS

ADDMENUITEM END!!! QUIT

SHOWMENU

You find a further example in the file TurboCalc.STD2 - it contains the macro commands for the standard TurboCalc menu.

1.150 SHOWMENU()

SHOWMENU()

Shows the modified (or new) menu after the automatic display has been switched off by NEWMENU.

This command should be located at the end of any menu declaration in script files.

1.151 Macro Control

Macro Control

BLOCKVARIABLE(Name;Block)

CLOSESHEET(Now)

DELETEVARIABLE(Name)

EXECUTE(File;Parameter;[Window])

IFFPRINT(File;Width;Height;Depth;Block)

FOLDER.ADDSHEET()

FOLDER.INSERTSHEET([Name])

FOLDER.REMOVESHEET()

FOLDER.RECALC()

FOLDER.RENAMESHEET()

FOLDER.SAVESHEET([Name])

FOLDER.SHOWSHEET()

MACROPLAY(Cell)

NEWSHEET(Name)

PRINT(Printer;File;NLQ;Range;Page1;Page2;LPI;Colored;Break;Size;Width;Height;OTypeFF)

QUIT([Flag])

RECALC([Mode])

RUN(File;Parameter;[Window])

RX(File;Port)

RXSYNC(File;Port)

SELECTSHEET(Name[;Windownumber])

SHEETHIDE(Sheetname;Windownumber)

SHEETSHOW(Sheetname;Windownumber)

START(Filename)

UNCHANGED()

VARIABLE(Name;Value)

1.152 BLOCKVARIABLE(Name;Block)

BLOCKVARIABLE(Name;Block)

Inserts a name in the variable list. The name can be subsequently used in formulas. For further details, refer to the section on "Names".

Name: is the (text) name of the block. (If possible, use names without blanks or other special characters).

Block: is the block which is to be given the name. If you omit this parameter, the current block will be used.

Note: To allocate names to numerical or alphanumeric values, please use the VARIABLE command.

Examples:

BLOCKVARIABLE("test";A1:C5)

allocates the name "test" to the block A1 to C5. This block can now be addressed with its reference A1:C5 or the name "test".

BLOCKVARIABLE("test")

the current block can now be addressed with the name "test".

1.153 CLOSESHEET(Now)

CLOSESHEET(Now)

Closes the current sheet (and asks about saving before quitting the sheet). If this command is called from the last opened sheet, TurboCalc will be terminated.

Now: determines if TurboCalc should open a requester before quitting.

0: no requester required.

1: or omission of the parameter: opens a requester and asks about saving before quitting the sheet.

Note: Please note that any other sheet can be declared as the "active" one after using this command. If you like to enter further commands always use the SELECTSHEET command next.

1.154 DELETEVARIABLE(Name)

DELETEVARIABLE(Name)

Deletes the specified Name from the name list. For further details, please refer to the VARIABLE command and "Names".

Name: is a string, which specifies the name to delete.

Example:

DELETEVARIABLE("VAT")

Deletes the name VAT, but not the attached cell and its contents. If this variable is still used in a formula, you will propably get an error when recalculating.

1.155 EXECUTE(File;Parameter;[Window])

EXECUTE(File;Parameter;[Window])

Starts an external AmigaDOS batch file and waits until it is finished.

It corresponds to the RUN command but in this case the command is executed (OS2.0 and higher allows the use of RUN only with the set script flag).

For further details and parameters, please refer to RUN.

Example:

EXECUTE("s:test")

starts the script file "s:test" consisting of AmigaDOS commands.

1.156 FOLDER.ADDSHEET()

FOLDER.ADDSHEET()

Inserts a new sheet into the current folder (at the end) and activates it. Thos corresponds to the <Sheet-New Sheet> menu item.

1.157 FOLDER.INSERTSHEET([Name])

FOLDER.INSERTSHEET([Name])

Inserts the sheet Name into the current folder. If Name is omitted, then a file requester appears to allow selection of the file. Ths corresponds to the <Edit-Insert Data-TurboCalc Sheet>>

1.158 FOLDER.REMOVESHEET()

FOLDER.REMOVESHEET()

Removes the current sheet from the current folder. This corresponds to the <Sheet-Remove Sheet> menu item.

1.159 FOLDER.RECALC()

FOLDER.RECALC()

Causes recalulation of all sheets in the current folder. The current sheet is recalculated first; then the other sheets, starting with the first of the folder.

See also RECALC.

1.160 FOLDER.RENAMESHEET()

FOLDER.RENAMESHEET()

Opens a window allowing the current sheet to be assigned a new name. Coresponds to the <Sheet-Rename Sheet> menu item.

1.161 FOLDER.SAVESHEET([Name])

FOLDER.SAVESHEET([Name])"

Saves the current sheet as Name. if the parametr is omitted, then a file requester appears allowing selection of a file name. This corresponds to the <Project-Export to-TurboCalc Sheet> menu item.

1.162 FOLDER.SHOWSHEET()

FOLDER.SHOWSHEET()

Opens a window listing all sheets of the current folder; allowing selection of a new sheet. Corresponds to the <Sheet-Show Sheet> menu item.

1.163 IFFPRINT(File;Width;Height;Depth;Block)

IFFPRINT(File;Width;Height;Depth;Block)

With this command, a part of a sheet can be saved as an IFF image. This allows (automatic) export to other programs.

File: Is the name of the file in which the image is to be saved. If the parameter is omitted, the previously specified name (specified either in this command, in PRINT or in the print-requester) will be used. If ""is indicated, a file name will be asked for (by the file requester). (Note: Macro execution will be suspended until the name is netered or the abort key is pressed).

Width/Height/Depth: The image format. If omitted, the previous setting will be kept.

Block: Determines the block which is to be saved. If this parameter is missing, the currently selected block will be used.

(Note: This does not normally correspond to the print range! If you want this, specify the block as PRINTRANGE, see last example.)

Examples:

IFFPRINT("ram:test";640;512;A1:D20)

Saves the range A1:D20 an image with the name ram:test and the size 640*512*3 (3=8 colors).

IFFPRINT("ram:test";;;;PRINTRANGE)

Saves the current print range as ram:test with the default size (i.e. the previous size). If no print range is determined, i.e. the variable PRINTRANGE is not defined, an error message will appear.

1.164 MACROPLAY(Cell)

MACROPLAY(Cell)

Starts the macro-execution beginning with the Cell given as parameter. If called without any parameters, this command corresponds to the menu-item <Macro-Play> and the appropriate selection-window will be opened.

Cell: is any cell specification or name according to the TurboCalc standard. You can even refer to other sheets using the AT-function (or @).

If this command is executed from ARexx, ARexx is forced to wait until the macro is finished.

Note: This command is not intended for calling subroutines from within a macro. Use the command CALL for that purpose.

1.165 NEWSHEET(Name)

NEWSHEET(Name)

Opens a new sheet and its corresponding window. The sheet is named as specified by the parameter Name or a default name if the parameter was omitted.

Name: Is the (string) name for the new sheet (this command does not replace the LOAD command. If you specify a pre-existing name (e.g. on diskette or harddisk), this file will not be loaded by NEWSHEET).

If the parameter name is omitted, the sheet will be named "Sheet1". TurboCalc will ask for the desired filename before the first save of the sheet.

1.166 PRINT(Printer;File;NLQ;Range;Page1;Page2;LPI;Colored;Break;Size;Width;Heigl

PRINT(Printer;File;NLQ;Range;Page1;Page2;LPI;Colored;Break;Size;Width;Height;OType;FF)

Printer: Determines if the output is to be sent to the printer (TRUE or non-zero) or to a file (0 or FALSE).

File: is the (text) file name for print redirection to a file.

NLQ: Determines the print-out quality: TRUE or non-zero switches to NearLetterQuality, the value 0 or FALSE prints the sheet in draft quality.

Range: TRUE (or non-zero) sends the entire sheet to the printer, FALSE (or 0) only the following pages:

Page1, Page2: specifies the pages to print.

LPI: switches between 6 lpi (TRUE or non-zero) or 8 lpi (FALSE or 0) (lpi= lines per inch).

Colored: toggles between color (TRUE or non-zero) or black/white (FALSE or 0) print.

Break: Determines if TurboCalc should pause after every page (FALSE or 0) or not (TRUE or non-zero).

Size: Determines the paper size:

0: A4 normal (21*29.7)

1: A4 fanfold (21*72)

2: A3 normal (29.7*42)

3: user defined:

Width, Height: Determines the paper width and height (in centimeters) for the "user defined" page dimensions.

Note: For compatiblity you have to enter the values in centimeters even if you have selected "inch" at "Locale" in <Sheet-Settings> . You may use "centimeter=inch/2.54" (or even enter this formula as the parameter).

Omitted parameters will not be changed.

If called without any parameters, this command corresponds to the menu-item <Project - Print> and the appropriate selection-window will be opened.

Further print parameters can be adjusted with the PRINTOPTIONS command.

OType: Defines the type of output.

0; Text

1: Graphics

2: Postscript

FF: determines if the page should be completed with a formfeed control character:

0: No FF at end (only line-feeds)

1: FF at the end of each page.

2/3: like 0/1 but: Stop immediately after printing, without ejecting the page.

Warnings:

* This option has been included for very special cases and should only be used if you know exactly what you're doing!

* Only works properly if no footer has been defined.

* Only sensible for single-page output (but applies to all pages)!

* The next printout won't have correct page alignment: Either set this anew, or set the formfeed to on (1, not 3!).

* This option only applies to one print even for reasons of sanity! Flags will be reset after printing.

Parameters which are omitted will not be changed. If no parameters are given, this command corresponds to the Project-Print> menu item and the corresponding window appears.

Example:

PRINT(;)

Prints the current sheet as set before; no parameters are changed.

PRINT()

shows the window for the print-options selection (corresponds to Project-Print>)

PRINT(0;"ram:print.out")

Prints the current sheet to the file "ram:print.out" . the remaining parameters stay unchanged.

PRINT(0;"ram:print.out";;0;2;3)

ditto, but limits the print-out to page 2 and 3.

1.167 QUIT([Flag])

QUIT([Flag])

This command ends a TurboCalc session.

If it is called within an ARexx script, this script receives the message OK (that means error value 0) first to terminate it normally. TurboCalc should not be addresses via ARexx after this command.

Flag: Determines if a confirm-requester is to be opened before quitting:

0 or FALSE: no requester (even if unsaved sheets are still in memory).

1 or TRUE: requester, if unsaved sheets are still in memory.

If this parameter is not specified, TRUE will be assumed (corresponding to the menu item <Project - Quit>.

1.168 RECALC([Mode])

RECALC([Mode])

If you call this command without a parameter, you force TurboCalc to recalculate the entire sheet (corresponds to <Commands - Recalculate>). This is useful if the automatic calculation is set to "off".

If the parameter Mode is given, the automatic calculation can be switched on (TRUE or not null) or off (0 or FALSE).

Note: The automatic calculation can be switched on or off with SHEETOPTIONS too (but this command requires further parameters). Because of the frequent use of the disabled automatic calculation (e.g. to speed up macros), this command has been added to the command set to simplify things.

Examples:

RECALC()

Recalculates all formulas and updates the sheet.

RECALC(0)

Switches off the automatic calculation.

1.169 RUN(File;Parameter;[Window])

RUN(File;Parameter;[Window])

Starts an external program from TurboCalc and waits until it is finished.

File: is the name of the file to be started. Please specify the full file path or pre-set it, so that the appropriate file can be found without problems.

Parameter: is the parameter text to be passed to the program. If there are several parameters, simply separate them with blanks (as usual) and write them in a single string.

Window: specifies the name of the window which is to be opened before starting the program. If this information is missing, the following default will be assumed: "CON:0/0/0/640/200/TurboCalc/CLOSE/WAIT/AUTO". This command causes OS2.0 and higher to open a window only if it is actually needed. Furthermore, the window stays preserved until you touch the closing gadget.

(1.3 users receive the message "Please press return" after execution. TurboCalc waits until this message is acknowledged.)

Please make sure that the window specification is correct, otherwise the execution of this command will not be possible.

Examples:

RUN("dir";"dfO:") shows the directory of drive dfO:

Print in the background:

=PRINT(0;"ram:printer.out")

=RUN("RUN";">nil: copy ram:printer.out prt: QUIET")

This macro prints into the file "ram:printer.out" and starts the "copy" command in the background via RUN, which sends the file to the printer.

Thus, you are able to continue your work during actual printing.

1.170 RX(File;Port)

RX(File;Port)

External, asynchronous Rexx-Scripts are started with this command.

File is the (text) file name.

Port selects the Rexx-Port to which the file/command is sent. If this parameter is ommited, "REXX" will be used (the name for the ARexx-Port). This can be used to send commands directly to other applications.

Note: if Port is specified, then the File parameter should be called 'Command' as this most often indicates the command (in contrast to the use without Port wherein a file is specified).

The return value of the script (OK/Error/Warning) will be written in the corresponding cell i.e. the cell in which holds the macro command (when this cammnd is used in a macro).

Note: Unlike RXSYNC, RX does not wait for the script to complete; the macro continues. Thus, no return value of the script will be available. However, RX can be used to start scripts which address TurboCalc's ARexx port.

Examples:

RX("Example.rexx") executes the Arexx-Script Example.rexx

RX"(Example.rexx","REXX") ditto

RX("command";"PORT") sends command to the application with the ARexx-Port PORT

RX('"address PORT command'") ditto, but this time command is redirected via the normal ARexx-Port

1.171 RXSYNC(File;Port)

RXSYNC(File;Port)

This commands allows for synchronous execution of ARexx scripts.

File: is text containng the name of the file.

Port: is the the Port to which 'File' is to be sent. If this parameter is ommited, "REXX" will be used (the name for the ARexx-Port). This can be used to send commands directly to other applications.

The return value of the script (OK/Error/Warning) will be written in the corresponding cell i.e. the cell in which holds the macro command (when this cammnd is used in a macro).

Note: if Port is specified, then the File parameter should be called 'Command' as this most often indicates the command (in contrast to the use without Port wherein a file is specified).

Note: RXSYNC executes the script synchronously; i.e. TurboCalc waits for the script to complete. No macro commands can be executed at ths time. Therefore, it's also not possible to send ARexx message to TurboCalc from the called script. If you need to do this, use RX instead.

1.172 SELECTSHEET(Name[;Windownumber])

SELECTSHEET(Name[;Windownumber])

This command selects a sheet as the new Macro/ARexx-sheet. All further commands and cell references will only affect this sheet.

This command does not activate a particular window automatically; for this purpose use the ACTIVATEWINDOW command.

Name: specifies the name of the sheet (the name, which is displayed in the window title but without "-2..."). If the sheet-name ends in ".TCD", this can be omitted. Furthermore, all path information can be left out.

TurboCalc accepts the following names for the example file "DHO:TurboCalc/sheets/example.TCD":

"example", "example.TCD", "TurboCalc/example"...

Windownumber: specifies the number of the window (that means the "-XXX"- part of the window title). The numbering starts with 1. If the parameter is omitted or too large, the first window of this sheet will be taken.

In Macro Mode: If you use this command in the macro mode note that the cell in which this command is located receives the value TRUE if the selected sheet was found. If the process has failed, the cell receives the value FALSE. This can be tested subsequently.

Note: Use the SHEETNAME function to obtain the name of the window (e.g. to buffer the current window temporarily).

Examples:

SELECTSHEET("sheet1")

Selects the sheet, which is normally opened after start-up.

SELECTSHEET("sheet1";2)

Selects a second split window of the same name as above produced with <Options-New Sheet Window>.

1.173 SHEETHIDE(Sheetname;Windownumber)

SHEETHIDE(Sheetname;Windownumber)

Hides the specified or current window. If you have only one open window left, the call of this command will produce an error message. TurboCalc expects at least one open window before you can use SHEETHIDE (so open or create a new sheet, if necessary).

Name: specifies the name of the sheet (the name, which is displayed in the window title but without "-2..."). If the sheet-name ends in ".TCD", this can be omitted. Furthermore, all path information can be left out.

TurboCalc accepts the following names for the example file "DHO:TurboCalc/sheets/example.TCD":

"example", "example.TCD", "TurboCalc/example"...

Windownumber: specifies the number of the window (that means the "-XXX"- part of the window title). The numbering starts with 1. If the parameter is omitted or too large, the first window of this sheet will be used.

If no parameter is given, the current window will be hidden. (This corresponds to <View- Hide Sheet-Window>).

In Macro Mode: If you use this command in macro mode, note that the cell in which this command is located receives the value TRUE if the selected sheet was found. If the process has failed, the cell receives the value FALSE. This can be tested subsequently.

Note: This function is useful for special macros: A macro adds a new menu item and hides itself after execution. Thus, the macro-sheet does not disturb normal proceeding but the function can be easily called via the menu or macro-requester.

1.174 SHEETSHOW(Sheetname;Windownumber)

SHEETSHOW(Sheetname;Windownumber)

Brings the specified sheet to the front and activates it.

It also redisplays the desired window if it has been previously hidden with SHEETHIDE.

Name: specifies the name of the sheet (the name, which is displayed in the window title but without "-2..."). If the sheet-name ends in ".TCD", this can be omitted. Furthermore, all path information can be left out.

TurboCalc accepts the following names for the example file "DHO:TurboCalc/sheets/example.TCD":

"example", "example.TCD", "TurboCalc/example"...

Windownumber: specifies the number of the window (that means the "-XXX"- part of the window title). The numbering starts with 1. If the parameter is omitted or too large, the first window of this sheet will be taken.

In Macro Mode: If you use this command in macro mode, note that the cell in which this command is located receives the value TRUE if the selected sheet was found. If the process has failed, the cell receives the value FALSE. This can be tested subsequently.

Note: This command does not activate the specified window, use SELECTSHEET for that purpose.

If you call this command without any parameters, a selection window will be opened. This corresponds to the menu item <View - Show Sheet-Window>.

1.175 START(Filename)

START(Filename)

This command calls an external TurboCalc command file. It consists of macro commands, ordered line by line (without quotation marks). The commands can either be separated by brackets (macro style), or by blanks (ARexx convention).

The commands GOTO, IFGOTO, MIF, CALL or FOR are not to be used in such a script file!

This is an optional method of writing macro programs. It is suitable for longer programs (with the disadvantage that branching and loop constructions are not possible).

Note: In order to be able to better locate errors, TurboCalc indicates - when an error has occurred - the error cell. In this case the column (the letter) can be ignored, the row corresponds to the row of the file (Example: A100: error in row 100).

Note: You can also use the specific sequence $$ as replacement for the parameter "filename", which causes TurboCalc to execute the internal script-file (which is run in the start-up phase). It creates the basic environment of TurboCalc (the menus and the screen). This file can be found on the original disk under TurboCalc.STD2 and can be edited like any other TurboCalc sheet.

For further information, refer to "Macro" "Executing a macro while starting TurboCalc".

Example:

START("ram:test")

with the following file "ram:test":

SELECT a1

PUT 3

SELECT A1:A10

SERIES(1;3)

1.176 UNCHANGED()

UNCHANGED()

This command marks the current sheet as "unchanged", which avoids the confirm request for saving the sheet when you try to close it or to leave TurboCalc.

Note: This is useful for macros that modify their sheets (e.g. to save intermediate calculation results).

1.177 VARIABLE(Name;Value)

VARIABLE(Name;Value)

Inserts a name in the name list. The name can be used subsequently in formulas. See "Names" for further details.

Name: is the (text) name of the variable. (If possible, use name definitions without blanks or other special characters).

Value: is the value which is to be named. It may be a number, a boolean value or text.

Text can also start with an equals sign "=" and thus introduce a formula or a reference/range, see examples below.

Note: If a block is to assigned a name, then this can be either by assigning text (e.g. "=A1:C5") of by using the BLOCKVARIABLE command.

If called without any parameters, this command corresponds to the menu-item <Commands - Define Names> and the appropriate selection-window will be opened.

Note: The first character of the name may be a "#".

This indicates that the variable is isolated to the current macro sheet, and not the work sheet. (This corresponds to the "#" function of the same name.) This can be used to assign temporary variable storage in the macro sheet.

Examples:

VARIABLE("VAT";0.15)

Allocates the value 0,15 to the name VAT. This name can now be used as a constant in formulas, e.g. 200*(1+VAT). In case of a change of the cell "VAT" and a following <Commands - Recalculation> the formula will be changed as well.

VARIABLE("cell1";"=C5")

Cell C5 receives the name "cell1" and can now be addressed under this name in any allowed TurboCalc expression (e.g. in formulas).

VARIABLE("Test";"=CELL(-1;-1)*2")

Allocates a formula to the name "Test" (here: take the cell left above the calling cell and double its contents).

1.178 Screen Control

Screen Control

ACTIVATEWINDOW()

CHANGECOLOR(Color;Red;Green;Blue)

CHANGEWINDOW(X;Y;Width;Height)

ICONIFY()

MOVEWINDOW(X;Y)

OLDCOLORS()

POSWINDOW()

POSWINDOW2()

SCREEN(Width;Height;Depth;Mode)

SETFONT(Font;Mode)

SIZEWINDOW(Width;Height)

STDCOLORS()

WINDOWTOBACK()

WINDOWTOFRONT()

ZOOM(ZoomX[;ZoomY])

1.179 ACTIVATEWINDOW()

ACTIVATEWINDOW()

Activates the current window.

1.180 CHANGECOLOR(Color;Red;Green;Blue)

CHANGECOLOR(Color;Red;Green;Blue)

This command is used to adjust the TurboCalc screen colors (or the workbench colors, if the TurboCalc windows are located on its screen):

Color: determines the color number which is to be set (0=backgroundcolor, 1=color 1...).

Red: determines the proportion of red in the color on a scale from 0 to 15 (0= no red, 15= only red).

Green: determines the proportion of green in the color on a scale from 0 to 15 (0= no green, 15= only green).

Blue: determines the proportion of blue in the color on a scale from 0 to 15 (0= no blue, 15= only blue).

If you call this command without any parameter, a selection-window will be opened where the screen colors may be set.

Note: The color requester can be opened to use for settings (<Sheet-Sheet Settings- Colors>).

The color numbers in the requester start with 1, therefore you have to subtract 1, if you want to use the information as parameters - the red/green/blue-proportions can remain unchanged.

Examples:

CHANGECOLOR(0;0;0;0)

sets the background color to black

CHANGECOLOR(0;15;15;15)

and to white

CHANGECOLOR(1;15;0;0)

sets the character color to red, assuming the standard color allocation.

1.181 CHANGEWINDOW(X;Y;Width;Height)

CHANGEWINDOW(X;Y;Width;Height)

Moves the current window on the screen and changes its size:

X,Y: specify the new coordinates.

Width, Height: determine the new dimensions of the window.

This is a combination of MOVEWINDOW and WINDOWSIZE.

1.182 ICONIFY()

ICONIFY()

This command corresponds to the menu item <Project - Iconify> and puts TurboCalc into a "standby mode". All windows are closed; only a small "memo window" (or from OS2.0 on an icon) is displayed. TurboCalc is restored to its previous state by clicking on this window (or icon)..

1.183 MOVEWINDOW(X;Y)

MOVEWINDOW(X;Y)

Moves the current window on the screen:

X,Y: specify the new coordinates.

1.184 OLDCOLORS()

OLDCOLORS()

Sets the screen colors as they were set when TurboCalc was started. This command is useful if you have made (temporary) color changes (e.g. from within a macro) and you wish to return to the old settings.

1.185 POSWINDOW()

POSWINDOW()

Corresponds to <View - Arrange Windows- show all>.

1.186 POSWINDOW2()

POSWINDOW2()

Corresponds to <View - Arrange Windows- standard>.

1.187 SCREEN(Width;Height;Depth;Mode)

SCREEN(Width;Height;Depth;Mode)

Determines if TurboCalc is to open its own screen, or if its windows are to appear on the workbench screen. This command also determines the appearance of the screen:

Width: determines the width of the screen (at least 600!). -1 or omission of the parameter corresponds to the width of workbench screen. The value 0 means: do not open custom screen, use workbench screen (all remaining parameters are ignored.)

Height: determines the height of the screen (at least 200!). -1 or omission of the parameter corresponds to the height of the workbench screen.

Depth: determines the number of bitplanes, which determines the number of colors. -1 or omission of the parameter uses workbench colors. 1 corresponds to two colors, 2= 4colors, 3= 8 colors and so on.

Mode: determines the screen mode in which the screen is to be opened (only for OS2.0 and higher). This parameter is only necessary for "unusual" formats. Regular formats are recognized by their screen dimensions.

This mode can either be entered as a number (screen-ID) or as a string. For further details, please refer to "Screen Modes" in the appendix.

If you call this command without any parameter, a selection-window will be opened where the screen mode may be set. This corresponds to the menu item <Sheet-Global Settings> and <Screen>.

Examples:

SCREEN() opens a window to determine the screen.

SCREEN(;) opens a screen with the dimensions and colors of the workbench.

SCREEN(0) closes a previously opened screen and puts all windows on the workbench.

SCREEN(640;256;3) opens a 640*256 pixel screen with 8 colors.

SCREEN(640;470;2"VP") opens a 640*480 pixel VGA-productivity screen.

1.188 SETFONT(Font;Mode)

SETFONT(Font;Mode)

This command sets the global font (corresponding to "Fonts" under the <Sheet - Global Settings> menu item).

Font: is the name of existing font (format: name/size, e.g. "Times/13"). If this parameter is omitted, a selection window with a list of all available fonts will appear.

Mode: this parameter determines which part of TurboCalc is to use the new font.

0 (or omission): selects the font for all menus and all windows.

1: font only for the current window

2: font for menus.

3: standard font for the current sheet (and corresponds to STDFONT)

Note: You can only select the mode 0 via the TurboCalc menus. Different fonts for different windows cannot be set via the menu. Please, refer to the examples below.

1st Example:

=ADDMENUITEM("Menufont";"SETFONT("""";2)";5)

This macro command adds the menu item <Menufont> at the end of the menu <Options>. If you select this new item, you can determine a new font for the menus.

2nd Example:

ADDMENUITEM WindowFont "SETFONT # 1"

(extracted from the TurboCalc.STD startup file)

Adds the menu item named "WindowFont" for selecting a new window-font.

1.189 SIZEWINDOW(Width;Height)

SIZEWINDOW(Width;Height)

Changes the size of the current window on the screen:

Width, Height: determine the new dimensions of the window.

1.190 STDCOLORS()

STDCOLORS()

Sets the standard TurboCalc colors. This is useful if you have previously adjusted your colors or for 1.3-users.

1.191 WINDOWTOBACK()

WINDOWTOBACK()

Puts the current window to the back on the screen.

1.192 WINDOWTOFRONT()

WINDOWTOFRONT()

Puts the current window to the front on the screen.

1.193 ZOOM(ZoomX[;ZoomY])

ZOOM(ZoomX[;ZoomY])

This command allows you to zoom in (or out) on the curent view (corresponds to <View-Zoom>) with the corresponding selection window appearing if no parameters are given.

ZoomX: Defines the magnification in the X direction as a percentage. 100 corresponds to the original size, with 200 causing the sheet to appear twice as large.

ZoomY: If specified, the magnification in the X- and Y- direction may be different. Otherwise ZoomX applies in both directions. (Note: Fonts are scaled according to ZoomY; thus if ZoomX and ZoomYdiffer, the fonts will appear distorted.)

1.194 Miscellaneous Commands

Miscellaneous Commands

Many commands presented in this chapter have, strictly speaking, little meaning in macro programming.

They have been included in the command set of TurboCalc due to the their application in the menu definitions,

e.g. ADDMENUITEM ("Record Macro";"RECORD")

ABOUT()

COMMAND()

CHARTSHOW()

EDIT()

FILEINFO()

INSERTFORMULA()

INSERTMACRO()

INSERTNAME()

NEWWINDOW()

OBJECTCLASS (Name)

OBJECT(Name;Type;Text;Value1;Value2;Block)

OBJECTPARA(Name;Macro;MacroActivation;Color;Frame)

OBJECTPOS(Name;X;Y;Width;Height)

OBJECTSELECT(Name)

PREVIEW()

RECORD()

REDO()

SPECIALFLAGS(Flags)

STOPRECORD()

SYSINFO()

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

UNDO()

1.195 ABOUT()

ABOUT()

Corresponds to menu item <Project - About> and shows a window with information about the author and his program.

1.196 COMMAND()

COMMAND()

Corresponds to an <F10> key press (in the standard keymap). This command is of no use in the normal macro mode, it was added to combine this function with a key combination or to define a corresponding menu item.

Example:

KEY("F1";"COMMAND")

1.197 CHARTSHOW()

CHARTSHOW()

Corresponds to menu item <Data - Show Chart...>.

1.198 EDIT()

EDIT()

Activates the input row, thus corresponding to <F2> (in the standard keymap). This command is of no use in the normal macro mode, it was added to combine this function with a key combination or to define a corresponding menu item.

Example:

KEY("F1";EDIT")

1.199 FILEINFO()

FILEINFO()

Corresponds to the <Help-File Info> menu option and show information about the current working folder (e.g. Name, creation date, author).

1.200 INSERTFORMULA()

INSERTFORMULA()

This corresponds to menu item <Command-Paste-Formula> . It opens a window, where the user can select a valid formula to be inserted into the current cell.

1.201 INSERTMACRO()

INSERTMACRO()

This corresponds to menu item <Command-Paste-Macro>. It opens a window, where the user can select a valid macro-command to be inserted into the current cell.

1.202 INSERTNAME()

INSERTNAME()

This corresponds to menu item <Command-Paste-Name>. It opens a window, where the user can select a defined name to be inserted into the current cell.

1.203 NEWWINDOW()

NEWWINDOW()

Corresponds to menu item <View - New View>.

1.204 OBJECTCLASS (Name)

OBJECTCLASS (Name)

Allows for the binding of object classes after initial startup. When TurboCalc starts it binds all object classes in TCLibs. (e.g. those in LIBS:TCLibs or PROGDIR:TCLibs).

Name: Is the name of the object class, if necessary with path.

Example:

OBJECTCLASS("dh0:TurboCalc/object_extra.tclib")

1.205 OBJECT(Name;Type;Text;Value1;Value2;Block)

OBJECT(Name;Type;Text;Value1;Value2;Block)

Creates a new object, see <Data-Create Object>. If no parameter is given, this menu item will also be called.

Name: The name of the object. The name can be used for OBJECTSELECT as well as OBJECTPARA.

Type: Specification as ObjectID, which can be found out with the help of the function OBJID from the corresponding object type short form.

The object classes have the following short forms:

TEXT text class

GEL graphical elements

IFF IFF-pictures

CHRT charts

LOGO external example class

Text: Text which will be passed to the object as a parameter. See below.

Value1, Value2: Two values which will be passed to the object as parameters.

Block: Determines the position/size of the object.

Object parameters: Text, value1 and value2 will be passed to the object and have - depending on the object type - different meanings. Thus it is possible to adjust the appearance when creating an object by macro:

Text object: Text: is the actual text, Value1/2: formatting

Drawing object: Value1: type, Value2: formatting

Picture object: Text: picture name

Parameters which are not used can be omitted or be replaced by any values ("" or 0 recommended).

Example:

OBJECT("Test";OBJID("TEXT");"This is a test-text object")

creates a text object at the current cursor position.

1.206 OBJECTPARA(Name;Macro;MacroActivation;Color;Frame;Background;Protectio

OBJECTPARA(Name;Macro;MacroActivation;Color;Frame;Background;Protection)

Determines the overall parameters of an object.

Name: Is the name of the object. If this name is missing, the currently selected object will be used. If there is no object selected an error message will appear.

Macro: Macro text which is to be executed when clicking with the mouse.

MacroActivation: Determines whether the macro text is to be executed when clicking with the mouse. (0/FALSE:Macro will not be executed, 1/TRUE: Macro will be executed).

Color: Determines the background color of the macro (0: no background color, 1: background color 0, 2: background color 1, ...)

Frame: Specifies the frame value. (0: no frame, 1: simple frame, 2: normal 3D frame, 3: "inverse" 3D frame)

Background: determines if the object is behind the cells (-1 or omitted; no change, 0: No (i.e. in foreground), 1: Yes (i.e. in the background))

Protection: Determines if the object is protected (-1 or omitted; no change, 0: protected, 1: not protected)

Parameters can be omitted, then the defined setting will not be changed.

If no parameter or only the name is specified, the selection window will appear for the setting of parameters (as with mouse double-click) for the specified or active object.

1.207 OBJECTPOS(Name;X;Y;Width;Height)

OBJECTPOS(Name;X;Y;Width;Height)

Changes the position and/or size of the active object or the named object.

Name: Is the name of the object. If it is missing, the currently selected object will be used. If there is no object selected an error message will appear.

X,Y,Width,Height: The corresponding parameter omitted or -1 will not change the setting. These specifications do not constitute cell specifications (objects do not necessarily need to begin exactly at the cell margins), but a smaller measure.

1.208 OBJECTSELECT(Name)

OBJECTSELECT(Name)

Selects the object Name. (Corresponds to a click on the object with the mouse.)

Name: The name of the object.

1.209 PREVIEW()

PREVIEW()

Calls Preview, corresponds to <Project-Preview>.

1.210 RECORD()

RECORD()

Corresponds to menu item <Macro - Record>.

This command should not be used in a macro for its only purpose is to facilitate menu creation (call it after an ADDMENUITEM only).

1.211 REDO()

REDO()

Corresponds to <Edit-Redo>.

1.212 SPECIALFLAGS(Flags)

SPECIALFLAGS(Flags)

This macro allows the setting of special flags for the current sheet. At present, there is only one flag:

Flags:

0 open normally

1 Load sheet in background - no window is shown.

Note: Other flags are reserved for future use. Only use 0 or 1!

This flag is used by the auto-open sheets which set up the menus and therefore do not require a window. Your own macros can also be loaded 'in background'.

1.213 STOPRECORD()

STOPRECORD()

Corresponds to menu item <Macro - Stop Recording>. This command should not be used in a macro as its only purpose is to facilitate menu creation.

1.214 SYSINFO()

SYSINFO()

μ

Corresponds to menu item <Help - Info> and displays the current system status.

1.215 TCMACRO(TCLib;Offset;Num1;Num2;Text)

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

This command allows binding and calling of external macro commands. TurboCalc can be readily be expanded by addition of function by this means. TurboCalc libraries are used are used for this purpose. You will find further details in the chapter "Projects".

TCLib: This is the name of the TurboCalc library (as text).

Offset: Specifies the command number (thus more than one command may exist in one library).

Num1, Num2, Text: Constitute the possible parameters for the command (Num1 and Num2 are numerical values, text is a text string).

The exact function as well as the parameter definition can be found in the description of the appropriate TC library. If you want to create a library yourself, you will find details in the above mentioned section about TCLibs.

1.216 UNDO()

UNDO()

Corresponds to <Edit-Undo>.

1.217 Special Macro Commands

Special Macro Commands

This chapter contains the remaining "macro only" commands. They control macro execution and allow ifor branching and subroutines. They cannot be used as ARexx commands.

CALL(Cell)

FOR(Variable;Begin;End;Step)

FORRANGE(Variable;Block;Flags)

GOTO(Cell)

IFGOTO(Condition;Cell)

LOOP()

MACRO(...)

MELSE()

MENDIF()

MIF(Condition)

NEXT()

ONERROR(Cell)

RETURN([Cell])

STEP([Flag])

UNTIL(Condition)

WHILE(Condition)

1.218 CALL(Cell)

CALL(Cell)

Transfers macro execution to the specified Cell where the macro will be continued. TurboCalc saves the origin cell of the CALL command to resume execution there, after the first RETURN in the part the macro branched to before.

The command is similar to GOTO but is intended to introduce subroutines (GOTO does not save the last position and thus cannot return to its cell of origin).

Cell: is a valid cell reference (e.g. C10) or a cell name (e.g. loop), which specifies the position at which to continue execution.

Note: Normally, cell references refer to the current sheet (and not to the macro-sheet). But as you would mostly call cells within the macro sheet, the cell reference within given refers to the macro sheet, regardless of which one is active. The CELL function also specifies the position relative to the cell where the macro command is located (and not to the current cursor position!).

If you need to branch into another sheet, this can be done with the use of the AT-reference (e.g. @DesiredMacro;C5).

1.219 FOR(Variable;Begin;End;Step)

FOR(Variable;Begin;End;Step)

Provides a FOR...NEXT loop which is part of almost every programming language. The rows following through to NEXT are executed several times.

Variable: This is text(!), which specifies the name of the variable which is set to the appropriate count value. Thus the current loop value can be interrogated within the loop. See the VARIABLE command for details about "variable". This parameter can also be omitted or be specified as "". Then a variable will not be defined.

Begin: Initial value for the loop

End : End value for the loop.

Step: Step size for the loop. This value will be added to the current value (at the beginning: initial value). It can also be negative. If omitted, 1 will be assumed.

For every NEXT the current value will be incremented by step and then it will be used as follows:

Step>=0: If the current value<=end value, flow of execution continues at the row after FOR.

Step<0: If the current value>end value, flow of execution continues at the row after FOR.

Otherwise, the loop will be terminated and the next command to be executed is the one after NEXT.

Note: Instead of NEXT, WHILE(condition) or UNTIL(condition) can be used. Thus the loop can also be combined with a further condition. For description see LOOP.

Note: The FOR-loop will (unlike in many other programming languages) be executed at least once - a test will only take place in NEXT (or UNTIL/WHILE).

Examples:

=FOR("abc";1;5)

=MESSAGE(text(abc))

=NEXT()

produces the numbers 1, 2, ... 5 in succession.

A1 =FOR("abc";5;1;-2)

A2 =REQUEST(text(abc))

A3 =WHILE(A2)

A4 ...

produces 5, 3, 1 in succession. The loop will be interrupted if you press abort in the requester. abs is set to "-1" in cell A4 (after the end of the loop), if an early abort did not take place in WHILE.

=FOR("";1;3)

=MESSAGE("Hello")

=NEXT()

outputs "Hello" three times - a counting variable is needed here and hence is not defined.

=FOR("";0;0;0)

...

=UNTIL(...)

corresponds to LOOP...UNTIL (even if it is a bit obtuse).

1.220 FORRANGE(Variable;Block;Flags)

FORRANGE(Variable;Block;Flags)

A loop construct similar to FOR, but the loop will be executed for every cell of the block instead of a number range:

Variable: This is text(!) which indicates the name of the variable, which is set to the appropriate cell. Thus the current cell of the block can be referenced within the loop. Concerning variables see the VARIABLE command. This parameter can also be omitted or be specified as "". A variable will then not be defined..

Block: Specifies the block which is to be "worked on" in succession. If this parameter is omitted, the currently defined block of the sheet will be used.

Flags: The sum of the following numbers:

1: The block will be worked on column by column (otherwise: row by row)

2: The block will be worked on from bottom right to top left (otherwise: top left ... bottom right).

4: At the beginning of the loop, the cell cursor will be set to the appropriate cell, otherwise the cell cursor will not be changed. In both cases the current cell can be read from the variable, if specified.

Examples:

=FORBLOCK("position";A1:C5)

=WRITE(position+1;position)

=NEXT()

Increases the cells of the range A1:C5 by 1.

=FORBLOCK("";A1:C5;4)

=COLORS(5)

=NEXT()

Gives the color red (or color 5) to the cells in the range A1:C5. The parameter 4 specifies that the cell cursor is to be set correspondingly. Therefore a cell parameter is not needed for COLORS and the name of the variable can be omitted.

=FORBLOCK("position";A1:C5)

=COLORS(5;;position)

=NEXT()

Has the same function as the macro described before, but the cell cursor does not have to be set for every loop iteration. This function is faster.

1.221 GOTO(Cell)

GOTO(Cell)

Branches to the position specified in the parameter Cell (unconditional jump). The next command will be taken from this "cell" and the macro will be continued there.

Cell: Is a valid cell reference (e.g. C10) or a cell name (e.g. loop), which specifies the position to resume the macro.

Note: Normally, cell references refer to the current sheet (and not to the macro-sheet). But as you would mostly GOTO cells within the macro sheet, the cell reference within given refers to the macro sheet, regardless of which one is active. The CELL function also specifies the position relative to the cell where the macro command is located (and not to the current cursor position!).

If you need to branch into another sheet, this can be done with the use of the AT-reference (e.g. @DesiredMacro;C5).

1.222 IFGOTO(Condition;Cell)

IFGOTO(Condition;Cell)

IFGOTO implements conditional branching. If the parameter "condition" is TRUE (or non-zero), the execution of the macro will be continued at the position given in "cell", otherwise in the next line.

Condition: any boolean value (with =,<>,<,<=,>,>=... also refer to "Formula - Booleans").

Cell: specifies the cell where the execution is to be continued, if the parameter "condition" is TRUE (for further details, see GOTO!).

GOTO is identical to the IFGOTO command and a TRUE condition.

Note: Normally, cell references refer to the current sheet (and not to the macro-sheet). But as you would mostly IFGOTO to cells within the macro sheet, the cell reference within given refers to the macro sheet, regardless of which one is active. The CELL function also specifies the position relatively to the cell where the macro command is located (and not to the current cursor position!).

If you need to branch into another sheet, this can be done with the use of the AT-reference (e.g. @DesiredMacro;C5).

Example:

IFGOTO(TRUE;C5)

corresponds to GOTO(C5), as TRUE always is!

The following program sequence executes the commands in cell A1 to A13 ten times (10 is written to cell B1 and decremented by one successively (9,8,...). Processing is repeated as long as B1 is greater than 0.).

This allows a quick loop-construction (which is of inferior quality to the original LOOP construct. Please, refer to its description for further details).

Cell Macro-Command

A10 =PUT(10;B1)

A11

A12 =PUT(B1;CELLABS(B1;3))

A13

A14 =PUT(B1-1;B1)

A15 =IFGOTO(B1>0;A11)

The command in cell A12 causes TurboCalc to set the cell with column 3 and row B1 (1 to 10) to value 1 to 10: cells B1 to B10 receive the contents 1 to 10!

1.223 LOOP()

LOOP()

LOOP together with UNTIL or WHILE constitute a loop construct, which means that a part of your routine is executed several times depending on a certain condition: The commands enclosed in LOOP and UNTIL or LOOP and WHILE are executed at least once. If the interpreter encounters the first WHILE and if the condition is TRUE (or, in case of UNTIL, FALSE), the part between LOOP and WHILE (or UNTIL) will be repeated until the end-condition is FALSE (or, in case of UNTIL, TRUE).

The entire construct works without cell references (in contrast to IFGOTO), which improves legibility, susceptibility to errors, and re-usability.

Whenever possible, use LOOP instead of IFGOTO.

Note: If you try to jump back within a loop by means of RETURN(), an error message will appear!

Example 1:

=PUT(1;A1)

= LOOP()

= PUT(A1+1;A1)

=WHILE(A1<10)

Example 2:

=PUT(1;A1)

=LOOP()

= PUT(A1+1;A1)

=UNTIL(A1=11)

Both examples write the start-value 1 to cell A1 and increment it by 1 as follows:

1. while it is less than 10.

2. until it is exactly 11.

(which leads, in this case, to an identical result)

Note: as you might have noticed, it is possible to leave blanks between the command keyword and the equals sign. Please make use of this to improve the structure (e.g. in complex loop constructs) and legibility of your routines.

Note: you can use UNTIL and WHILE together as loop conditions, but in most cases one single condition will be simpler and structurally clearer.

As mentioned before, it is possible to build up more complex loop constructs (e.g. loops within a loop).

e.g. the following Example:

=PUT(1;A1)

=PUT(1;A3)

=LOOP()

= PUT(1;A2)

=LOOP()

= PUT(A2+1;A2)"

= PUT(A3+1;A3)

=UNTIL(A2=5)

= PUT(A1+1;A1)

=UNTIL(A1=5)

This example program counts cell A1 from 1 to 5 and (while doing this) cell A2 from 1 to 5. In every step A3 is increased by 1.

1.224 MACRO(...)

MACRO(...)

It is advisable to use this command as the first command of every macro with its name set in the parameter brackets to improve the structure of the program.

This command has no effect on execution of a macro and its use is not mandatory!

The expression in brackets will be ignored - therefore you do not have to specify a valid formula or text in quotation marks.

Example:

MACRO(TestMacro)

1.225 MELSE()

MELSE()

Introduces the else-branch in the MIF-request, see there.

1.226 MENDIF()

MENDIF()

Ends a MIF-request, see there.

1.227 MIF(Condition)

MIF(Condition)

This control construct allows for conditional excecution similar to IFGOTO.:

=MIF(condition)

...

=MELSE()

...

=MENDIF()

The advantage of this construct is that a jump (GOTO) does not have to be specified, but that either (if the condition is true) the part after MIF (til MELSE or MENDIF) will be executed or omitted (if the condition is false). In the second case the part between MELSE and MENDIF (if available) will be executed.

MIF...[MELSE]...MENDIF can also be nested to several levels.

Note: Every MIF must be completed with MENDIF! The specification of MELSE (between MIF and MENDIF) is optional.

Examples:

=MIF(value(A1)=5)

= MESSAGE("Cell A1 contains 5")

=MENDIF()

Outputs the message "Cell A1 contains 5", if this is so. Otherwise this command will be skipped.

=MIF(value(A1)=5)

= MESSAGE("cell A1 contains 5")

=MELSE

= MESSAGE("cell A1 does not contain 5!")

=MENDIF

As described before, except that the text "A1 does not contain 5" appear otherwise.

1.228 NEXT()

NEXT()

Ends FOR and FORRANGE loops; see there.

1.229 ONERROR(Cell)

ONERROR(Cell)

With this command you can determine a cell to which is to be jumped to if an error occurs. The error number can be determined from the LASTERROR function. The error handling must be end with RETURN() or RETURN(cell). If another error occurs before such a RETURN, the macro execution will be terminated.

Cell: Determines the cell to be jumped to if an error occurs. If this parameter is omitted, the error routine will be turned off.

Examples:

ONERROR(X1) If an error occurs, a jump will take place to X1.

ONERROR() Turns off the error handling.

The (very simple) error routine may appear as follows:

X1 =MESSAGE("Error number "+TEXT(LASTERROR)+" occurred!")

X2 =RETURN()

(see also example file)

1.230 RETURN([Cell])

RETURN([Cell])

This should be the last command of every macro as it terminates its execution.

If you have branched to a subroutine (through CALL), it terminates the respective subroutine and returns to the first cell after the subroutine-CALL.

Cell: If you specify this parameter, the execution will be continued in this cell instead of the first cell after the subroutine-CALL.

1.231 STEP([Flag])

STEP([Flag])

Switches the single-step-mode on (if the parameter Flag is omitted or not zero) or off (in case of 0 or FALSE).

This mode causes TurboCalc to display the next command to perform and to pause a while before its execution (tracing or trace mode).

It is very useful for debugging purposes, please refer to the corresponding chapter seven "Macro-commands" as well.

1.232 UNTIL(Condition)

UNTIL(Condition)

This is one end-condition for the LOOP construct.

If the condition results in FALSE, execution continues at the first command after LOOP (continuing the loop).

If the condition results in TRUE, the next command outside the loop will be executed (leaving the loop).

For further details, please refer to LOOP and WHILE.

1.233 WHILE(Condition)

WHILE(Condition)

This is an end-condition for the LOOP construct.

If the condition results in TRUE, execution continues at the first command after LOOP (continuing the loop).

If the condition results in FALSE, the next command outside the loop will be executed (leaving the loop).

For further details, please refer to LOOP and UNTIL.

1.234 Special ARexx-Commands

Special ARexx-Commands

This section introduces special ARexx commands, which have no equivalent in the combined macro/ARexx language. You should not call the following constructs from within a macro, because most of them serve to request values and other specifications from a sheet, which is in the macro language performed by stating the cell reference, etc. directly.

GETCURSORPOS

GETFORMULA [Cell]

GETVALUE [Value]

REM ...

1.235 GETCURSORPOS

GETCURSORPOS

This command returns the current cell or block of the selected sheet and saves it under RESULT in the regular reference format (A1 or C5:H7).

Note: Make sure that you have inserted the "option results" command at the beginning of your ARexx script. This internal ARexx-command allows the return of values.

Note: This command can be used to buffer the cursor position temporarily.

Example:

/*read out old position*/

GETCURSORPOS

oldpos = RESULT

/* Here you may add your routine */

.....

/* set old position */

"SELECT" oldpos

1.236 GETFORMULA [Cell]

GETFORMULA [Cell]

This commands is used to return TurboCalc formulas and to transfer them to the ARexx script. There, they can be addressed via the RESULT variable.

Cell: can be any valid cell reference. If this information is missing, the formula of the current cell will be returned and displayed.

If the current cell does not contain a formula, the blank text "" will be returned.

Note: Make sure that you have inserted the "option results" command at the beginning of your ARexx script. This internal ARexx-command allows the return of values.

Examples:

PUT "=A1+A2"

GETFORMULA A3

SAY RESULT

reads out the formula of cell A3 (which has been defined as =A1+A2 before)

RESULT is filled with =A1+A2

"SELECT A3"

GETFORMULA

SAY RESULT

ditto, but the SELECT statement declares cell A3 as the current one so you can leave out the cell reference in the GETFORMULA command.

1.237 GETVALUE [Value]

GETVALUE [Value]

This commands is used to obtain TurboCalc values (numbers, text...) and to transfer them to the ARexx script. There, they can be adressed via the RESULT variable.

Value: can be any valid formula (a simple numerical value, cell reference, functions...), except pure strings (alphanumercial data). If this information is missing, the contents of the current cell will be returned and shown.

Note: normally, this command will be used only to read out the contents of a cell (calculations can be executed directly in ARexx). Please check the first two examples.

Note: Make sure that you have inserted the "option results" command at the beginning of your ARexx script. This internal ARexx-command allows the return of values.

Example:

GETVALUE A1

SAY RESULT

reads out the value of cell A1 and puts it to the screen.

"SELECT A1"

GETVALUE

SAY RESULT

ditto, but here GETVALUE (without parameter) reads out the current cell.

GETVALUE "5+5"

returns 10 to the ARexx variable RESULT.

GETVALUE "SUM(A1:A10)"

calculates the sum of cell A1 to A10 and transfers it to RESULT (please note the combination of ARexx and TurboCalc macro commands).

1.238 REM ...

REM ...

(also ** or --)

This command has no operational effect. The text following the key-word is treated as a pure remark. This is applicable for macro-script files only (please, refer to the START command). Within macros it is possible to write any remark as normal text.

1.239 Table of Contents

Table of Contents

ALIGNMENT([Hor];[Vert];[Block])

BOX(Left;Right;Top;Bottom;[Block])

CHANGESTYLE(Num;[Block])

COLORS([Color1];[Color2];[block])

COLUMNTITLE(Title;[Cell])

COLUMNWIDTH(Width;[Block])

DEFAULTCOLUMNWIDTH(Width)

DEFAULTROWHEIGHT(Height)

FONT([Num];(Characterset];[Block])

FRAME(Left;Right;Top;Bottom;[Block])

FREEZE(Cell)

NOTE(Note;[Cell])

HIDE(Row;[Block])

NUMERICFORMAT(Format;[Block])

PATTERN(Number;[Block])

PROTECTION([Write];[Formula];[Block])

ROWHEIGHT(Height;Block)

ROWTITLE(Title;[Cell])

SHOW(Row;[Block])

STDFONT(Font)

Cursor Control

COLUMN(Column)

CURRENTCELL()

CURSORDOWN(Num)

FIND(Text;Part;Case;Columns;Range)

GOTOCOLUMN(Column)

GOTOLINE(Line)

LASTCOLUMN()

LASTROW()

CURSORLEFT(Num)

LINE(Line)

CURSORRIGHT(Num)

PING(Index)

PONG(Index)

SELECT([Block])

SELECTTOFRONT([Block])

SELECTWAIT()

CURSORUP(Num)

Input Commands

AMIGAGUIDE(File;Command)

BEEP()

CHELP(Num;Link)

DELAY(Time)

DIALOGUE(Dialogue;Hook)

DLGRESUME(Flag)

FILEREQUEST([File];[Title];[Cell])

HELP(Num;File)

INPUT(Text[;Title];[Cell])

LISTREQUEST([Title];[Block])

MESSAGE(Text[;Title])

PUT(Contents[;Cell])

REQPARA(X;Y;Oktext;Aborttext;Time)

REQUEST(Text[;Title])

Load / Save

CLIPREAD([Block])

CLIPWRITE([Block])

CSVINSERT([Block];[Name];[Separator])

CSVLOAD([Name];[Separator])

CSVSAVE([Name];[Separator])

CSVSAVEBLOCK([Block];[Name];[Separator])

LOAD([Name])

LOADCONFIG([File])

PROCALCINSERT([Block];[Name])

PROCALCLOAD([Name])

SAVE([Name])

SAVEAS([Name])

SAVEBLOCK([Block];[Name])

SAVECONFIG([File])

SYLKINSERT([Block];[Name])

SYLKLOAD([Name])

SYLKSAVE([Name])

SYLKSAVEBLOCK([Block];[Name])

TCDINSERT([Block];[Name])

Database Commands

CRITERIA([Range])

DATABASE([Block])

DBDELETE()

DBEXTRACT([Cell])

DBFIND([Cell])

DBMASK([Routine])

DBPREV(Cell)

DBSORT(Ascending;[Cell])

Options

AUTOCORRECT()

AUTOFILL()

AUTOSAVE(Flag)

CLIPBOARD(Unit;Separators;Quotation marked)

DEFPUBSCREEN(Screen)

DISPLAY(Title;Grid;Toolbar;Formulas;Zero;CursorMode)

FORMFEED(Flag)

GLOBALFLAGS()

KEY(Key;Command)

LOCALE(NF1;NF2;DF;Currency;CPrefix;CSuffix;Inch)

OPENFLAGS(NoAutoMacro;Hide;Password)

PAPERFORMAT()

POSTSCRIPT()

PRINTFORMAT(LM;RM;UM;BM;Style;Header;H-Text;Footer;F-Text;Title;Grid)

PRINTRANGE(Activate;[Range])

PROTECTFLAGS(Active;Password)

PSFONTLIST()

REFRESH(Mode)

SETTINGS()

SETTINGS.PREVIEW()

SETTINGS.RECALC()

SETTINGS.SCREEN()

SETTINGS.UNDO()

SHANGHAI(Mode)

SHEETFLAGS(Maxwidth;Maxheight;Calculation;Return;Direction;Icons)

SHEETSETTINGS()

SMARTREFRESH(Flag)

TOOLBAR()

Menu Commands

ADDMENUITEM(Name;Command;[Menu;Item])

ADDMENUSUB(Name;Command;[Menu;Item;Sub])

ADDMENUTITLE(Name;[Menu])

CHARTMENU()

DELMENUITEM(Menu;Item)

DELMENUSUB(Menu;Item;Sub)

DELMENUTITLE(Menu)

NEWMENU()

SHOWMENU()

Macro Control

BLOCKVARIABLE(Name;Block)

CLOSESHEET(Now)

DELETEVARIABLE(Name)

EXECUTE(File;Parameter;[Window])

FOLDER.ADDSHEET()

FOLDER.INSERTSHEET([Name])

FOLDER.REMOVESHEET()

FOLDER.RECALC()

FOLDER.RENAMESHEET()

FOLDER.SAVESHEET([Name])

FOLDER.SHOWSHEET()

IFFPRINT(File;Width;Height;Depth;Block)

MACROPLAY(Cell)

NEWSHEET(Name)

PRINT(Printer;File;NLQ;Range;Page1;Page2;LPI;Colored;Break;Size;Width;Height;OType;FF)

QUIT([Flag])

RECALC([Mode])

RUN(File;Parameter;[Window])

RX(File;Port)

RXSYNC(File;Port)

SELECTSHEET(Name[;Windownumber])

SHEETHIDE(Sheetname;Windownumber)

SHEETSHOW(Sheetname;Windownumber)

START(Filename)

UNCHANGED()

VARIABLE(Name;Value)

Screen Control

ACTIVATEWINDOW()

CHANGECOLOR(Color;Red;Green;Blue)

CHANGEWINDOW(X;Y;Width;Height)

ICONIFY()

MOVEWINDOW(X;Y)

OLDCOLORS()

POSWINDOW()

POSWINDOW2()

SCREEN(Width;Height;Depth;Mode)

SETFONT(Font;Mode)

SIZEWINDOW(Width;Height)

STDCOLORS()

WINDOWTOBACK()

WINDOWTOFRONT()

ZOOM(ZoomX[;ZoomY])

Miscellaneous Commands

ABOUT()

COMMAND()

CHARTSHOW()

EDIT()

FILEINFO()

INSERTFORMULA()

INSERTMACRO()

INSERTNAME()

NEWWINDOW()

OBJECTCLASS (Name)

OBJECT(Name;Type;Text;Value1;Value2;Block)

OBJECTPARA(Name;Macro;MacroActivation;Color;Frame;Background;Protection)

OBJECTPOS(Name;X;Y;Width;Height)

OBJECTSELECT(Name)

PREVIEW()

RECORD()

REDO()

SPECIALFLAGS(Flags)

STOPRECORD()

SYSINFO()

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

UNDO()

Special Macro Commands

CALL(Cell)

FOR(Variable;Begin;End;Step)

FORRANGE(Variable;Block;Flags)

GOTO(Cell)

IFGOTO(Condition;Cell)

LOOP()

MACRO(...)

1.240 Index

CHANGESTYLE(Num;[Block])

CHANGEWINDOW(X;Y;Width;Height)

CHARTMENU()

CHARTSHOW()

CHELP(Num;Link)

CLEAR(Data;Block])

CLIPBOARD(Unit;Separators;Quotation marked)

CLIPREAD([Block])

CLIPWRITE([Block])

CLOSESHEET(Now)

COLORS([Color1];[Color2];[block])

COLUMN(Column)

COLUMNTITLE(Title;[Cell])

COLUMNWIDTH(Width;[Block])

COMMAND()

COPY([Block])

CRITERIA([Range])

CSVINSERT([Block];[Name];[Separator])

CSVLOAD([Name];[Separator])

CSVSAVE([Name];[Separator])

CSVSAVEBLOCK([Block];[Name];[Separator])

CURRENTCELL()

Cursor Control

CURSORDOWN(Num)

CURSORLEFT(Num)

CURSORRIGHT(Num)

CURSORUP(Num)

CUT([Block])

D

Database Commands

DATABASE([Block])

DBDELETE()

DBEXTRACT([Cell])

DBFIND([Cell])

DBMASK([Routine])

DBPREV(Cell)

DBSORT(Ascending;[Cell])

DEFAULTCOLUMNWIDTH(Width)

DEFAULTROWHEIGHT(Height)

DEFPUBSCREEN(Screen)

DELAY(Time)

DELETEVARIABLE(Name)

DELMENUITEM(Menu;Item)

DELMENUSUB(Menu;Item;Sub)

DELMENUTITLE(Menu)

DIALOGUE(Dialogue;Hook)

DISPLAY(Title;Grid;Toolbar;Formulas;Zero;CursorMode)

DLGRESUME(Flag)

E

EDIT()

EXECUTE(File;Parameter;[Window])

F

FILEINFO()

FILEREQUEST([File];[Title];[Cell])

FILL(Mode;[Block])

FIND(Text;Part;Case;Columns;Range)

FOLDER.ADDSHEET()

FOLDER.INSERTSHEET([Name])

FOLDER.RECALC()

FOLDER.REMOVESHEET()

FOLDER.RENAMESHEET()

FOLDER.SAVESHEET([Name])

FOLDER.SHOWSHEET()

FONT([Num];(Characterset];[Block])

FOR(Variable;Begin;End;Step)

Formatting Commands

FORMFEED(Flag)

FORRANGE(Variable;Block;Flags)

FRAME(Left;Right;Top;Bottom;[Block])

FREEZE(Cell)

G

General

GETCURSORPOS

GETFORMULA [Cell]

GETVALUE [Value]

GLOBALFLAGS()

GOTO(Cell)

GOTOCOLUMN(Column)

GOTOLINE(Line)

H

HELP(Num;File)

HIDE(Row;[Block])

I

ICONIFY()

IFFPRINT(File;Width;Height;Depth;Block)

IFGOTO(Condition;Cell)

Index

Input Commands

INPUT(Text[;Title];[Cell])

INSERTFORMULA()

INSERTMACRO()

INSERTNAME()

K

KEY(Key;Command)

L

LANGUAGE(Mode;Block)

LASTCOLUMN()

LASTROW()

LINE(Line)

LISTREQUEST([Title];[Block])

Load / Save

LOAD([Name])

LOADCONFIG([File])

LOCALE(NF1;NF2;DF;Currency;CPrefix;CSuffix;Inch)

LOOP()

M

Macro Control

MACRO(...)

MACROPLAY(Cell)

MADD(Block1;Block2;ResBlock)

MELSE()

MENDIF()

Menu Commands

Menu Commands

MESSAGE(Text[;Title])

MIF(Condition)

Miscellaneous Commands

SETTINGS.RECALC()

SETTINGS.SCREEN()

SETTINGS.UNDO()

SHANGHAI(Mode)

SHEETFLAGS(Maxwidth;Maxheight;Calculation;Return;Direction;Icons)

SHEETHIDE(Sheetname;Windownumber)

SHEETSETTINGS()

SHEETSHOW(Sheetname;Windownumber)

SHOW(Row;[Block])

SHOWMENU()

SIZEWINDOW(Width;Height)

SMARTREFRESH(Flag)

SORT(Ascending;Direction;Cell;Block)

Special ARexx-Commands

Special Macro Commands

SPECIALFLAGS(Flags)

START(Filename)

STDCOLORS()

STDFONT(Font)

STEP([Flag])

STOPRECORD()

SYLKINSERT([Block];[Name])

SYLKLOAD([Name])

SYLKSAVE([Name])

SYLKSAVEBLOCK([Block];[Name])

SYSINFO()

T

TCDINSERT([Block];[Name])

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

Text

TOOLBAR()

TRANSPOSE([Block])

U

UNCHANGED()

UNDO()

UNTIL(Condition)

V

VARIABLE(Name;Value)

W

WHILE(Condition)

WINDOWTOBACK()

WINDOWTOFRONT()

Z

ZOOM(ZoomX[;ZoomY])