Text Functions

FIND

Returns the position of a string of text within another string.You can also define where to begin the search. The search term can be a number or any string of characters. The search is case-sensitive.

Syntax

FIND("FindText"; "Text" [; Position])

FindText refers to the text string to be found.

Text is the text string in which the search takes place.

Position (optional) is the position in the text string from which the search starts.

Example

=FIND(76;998877665544) returns 6.

LEFT

Returns the first character or characters of a text string.

Syntax

LEFT("Text" [; Number])

Text is the text for which the initial partial words are to be determined.

Number (optional) specifies the number of characters for the start text. If this parameter is not defined, one character is returned.

Example

=LEFT("output";3) returns “out”.

LEN

Returns the length of a text string including spaces.

Syntax

LEN("Text")

Text is the text whose length is to be determined.

Example

=LEN("Good Afternoon") returns 14.

MID

Returns a segment of a text string. The parameters specify the starting position and the number of characters.

Syntax

MID("Text"; Start; Number)

Text is the text containing the characters to extract.

Start is the position of the first character in the text to extract.

Number specifies the number of characters in the part of the text.

Example

=MID("office";2;2) returns ff.

RIGHT

Returns the last character or characters of a text string.

Syntax

RIGHT("Text" [; Number])

Text is the text of which the right part is to be determined.

Number (optional) is the number of characters from the right part of the text. If this parameter is not defined, one character is returned.

Example

=RIGHT("Sun";2) returns un.

SEARCH

Returns the position of a text segment within a character string. You can set the start of the search as an option. The search text can be a number or any sequence of characters. The search is not case-sensitive.

Syntax

SEARCH("FindText"; "Text" [; Position])

FindText is the text string to be searched for.

Text is the text string in which the search will take place.

Position (optional) is the position in the text from which the search is to start.

Example

=SEARCH(54;998877665544) returns 10.

SUBSTITUTE

Substitutes new text for old text in a string.

Syntax
SUBSTITUTE ("Text"; "SearchText"; "NewText" [; Occurrence])

Text is the text string in which text segments are to be exchanged.

SearchText is the text segment that is to be replaced (a number of times).

NewText is the text string that is to replace the text segment.

Occurrence (optional) indicates which occurrence of the search text is to be replaced. If this parameter is omitted the search text is replaced throughout.

Example
=SUBSTITUTE("123123123";"3";"abc") returns 12abc12abc12abc.

=SUBSTITUTE("123123123";"3";"abc";2) returns 12312abc123.

=SUBSTITUTE(“123123123”;”3”;””) returns 121212

Date Functions

DATE

This function calculates a date specified by year, month, day and displays it in the cell's formatting. The default format of a cell containing the DATE function is the date format, but you can format the cells with any other number format.

Syntax
DATE(Year; Month; Day)

Year is an integer between 1583 and 9957 **Month** is an integer indicating the month.

Day is an integer indicating the day of the month.

Example
=DATE(1998;07;28) yields 28/07/19980 if the cell format setting is DD/MM/YYYY.

DATEDIF

This function returns the number of whole days, months or years between Start date and End date.

Syntax

DATEDIF(Start date; End date; Interval)

Start date is the date from when the calculation is carried out.

End date is the date until the calculation is carried out. End date must be later, than Start date.

Interval is a string, accepted values are "d", "m", "y", "ym", "md" or "yd".

When entering dates as part of formulas, slashes or dashes used as date separators are interpreted as arithmetic operators. Therefore, dates entered in this format are not recognized as dates and result in erroneous calculations. To keep dates from being interpreted as parts of formulas use the DATE function, for example, DATE(1954;7;20), or place the date in quotation marks and use the ISO 8601 notation, for example, "1954-07-20". Avoid using locale dependent date formats such as "07/20/54", the calculation may produce errors if the document is loaded under different locale settings.

Value for "Interval"	Return value
"d"	Number of whole days between Start date and End date.
"m"	Number of whole months between Start date and End date.
"y"	Number of whole years between Start date and End date.
"ym"	Number of whole months when subtracting years from the difference of Start date and End date.
"md"	Number of whole days when subtracting years and months from the difference of Start date and End date.
"yd"	Number of whole days when subtracting years from the difference of Start date and End date.

Example

Birthday calculation. A man was born on 1974-04-17. Today is 2012-06-13.

=DATEDIF("1974-04-17";"2012-06-13";"y") yields 38.

=DATEDIF("1974-04-17";"2012-06-13";"ym") yields 1.

=DATEDIF("1974-04-17";"2012-06-13";"md") yields 27.

So he is 38 years, 1 month and 27 days old.

=DATEDIF(DATE(1974,4,17);"2012-06-13";"m") yields 457, he has been living for 457 months.

=DATEDIF("1974-04-17";"2012-06-13";"d") yields 13937, he has been living for 13937 days.

=DATEDIF("1974-04-17";DATE(2012;06;13);"yd") yields 57, his birthday was 57 days ago.

Logical Functions

AND

Returns TRUE if all arguments are TRUE. If one of the elements is FALSE, this function returns FALSE.

The arguments are either logical expressions themselves (TRUE, 1<5, 2+3=7, B8<10) that return logical values, or arrays (A1:C3) containing logical values.

Syntax

AND(Logical 1 [; Logical 2 [; … [; Logical 255]]])

Logical 1, Logical 2, … , Logical 255 are boolean values, references to cells or to cell ranges of logical values.

This function ignores any text or empty cell within a data range.

Example

The logical values of entries 12<13; 14>12 and 7<6 are to be checked:

=AND(12<13;14>12;7<6) returns FALSE.

=AND(FALSE();TRUE()) returns FALSE.

IF

Specifies a logical test to be performed.

Syntax

IF(Test [; [ThenValue] [; [OtherwiseValue]]])

Test is any value or expression that can be TRUE or FALSE.

ThenValue (optional) is the value that is returned if the logical test is TRUE.

OtherwiseValue (optional) is the value that is returned if the logical test is FALSE.

In the LibreOffice Calc functions, parameters marked as "optional" can be left out only when no parameter follows. For example, in a function with four parameters, where the last two parameters are marked as "optional", you can leave out parameter 4 or parameters 3 and 4, but you cannot leave out parameter 3 alone.

Example

=IF(A1>5;100;"too small") If the value in A1 is greater than 5, the value 100 is returned; otherwise, the text too small is returned.

=IF(A1>5;;"too small") If the value in A1 is greater than 5, the value 0 is returned because empty parameters are considered to be 0; otherwise, the text too small is returned.

=IF(A1>5;100;) If the value in A1 is less than 5, the value 0 is returned because the empty **OtherwiseValue** is interpreted as 0; otherwise 100 is returned.

OR

Returns TRUE if at least one argument is TRUE. This function returns the value FALSE if all the arguments have the logical value FALSE.

The arguments are either logical expressions themselves (TRUE, 1<5, 2+3=7, B8<10) that return logical values, or arrays (A1:C3) containing logical values.

Syntax

OR(Logical 1 [; Logical 2 [; … [; Logical 255]]])

Logical 1, Logical 2, … , Logical 255 are boolean values, references to cells or to cell ranges of logical values.

Example

The logical values of entries 12<11; 13>22 and 45=45 are to be checked.

=OR(12<11;13>22;45=45) returns TRUE.

=OR(FALSE();TRUE()) returns TRUE.

VLOOKUP Functions

VLOOKUP

Vertical search with reference to adjacent cells to the right. This function checks if a specific value is contained in the first column of an array. The function then returns the value in the same row of the column named by **Index**. If the **Sorted** parameter is omitted or set to TRUE or one, it is assumed that the data is sorted in ascending order. In this case, if the exact **Lookup** is not found, the last value that is smaller than the criterion will be returned. If **sorted** is set to FALSE or zero, an exact match must be found, otherwise the error **Error: Value Not Available** will be the result. Thus, with a value of zero the data does not need to be sorted in ascending order.

When using functions where one or more arguments are search criteria strings that represents a regular expression, the first attempt is to convert the string criteria to numbers. For example, ".0" will convert to 0.0 and so on. If successful, the match will not be a regular expression match but a numeric match. However, when switching to a locale where the decimal separator is not the dot makes the regular expression conversion work. To force the evaluation of the regular expression instead of a numeric expression, use some expression that cannot be misread as numeric, such as ".[0]" or ".\0" or "(?i).0".

Syntax

=VLOOKUP(Lookup; Array; Index [; SortedRangeLookup])

Lookup is the value of any type looked for in the first column of the array.

Array is the reference, which is to comprise at least as many columns as the number passed in Index argument.

Index is the number of the column in the array that contains the value to be returned. The first column has the number 1.

SortedRangeLookup is an optional parameter that indicates whether the first column in the array contains range boundaries instead of plain values. In this mode, the lookup returns the value in the row with first column having value equal to or less than **Lookup**. E.g., it could contain dates when some tax value had been changed, and so the values represent starting dates of a period when a specific tax value was effective. Thus, searching for a date that is absent in the first array column, but falls between some existing boundary dates, would give the lower of them, allowing to find out the data being effective to the searched date. Enter the Boolean value FALSE or zero if the first column is not a range boundary list. When this parameter is TRUE or not given, the first column in the array **must be sorted in ascending order**. Sorted columns can be searched much faster and the function always returns a value, even if the search value was not matched exactly, if it is greater than the lowest value of the sorted list. In unsorted lists, the search value must be matched exactly. Otherwise the function will return #N/A with message: **Error: Value Not Available**.Example

You want to enter the number of a dish on the menu in cell A1, and the name of the dish is to appear as text in the neighbouring cell (B1) immediately. The Number to Name assignment is contained in the D1:E100 array. D1 contains 100, E1 contains the name Vegetable Soup, and so forth, for 100 menu items. The numbers in column D are sorted in ascending order; thus, the optional Sorted parameter is not necessary.

Enter the following formula in B1:

=VLOOKUP(A1;D1:E100;2)

As soon as you enter a number in A1 B1 will show the corresponding text contained in the second column of reference D1:E100. Entering a non-existent number displays the text with the next number down. To prevent this, enter FALSE as the last parameter in the formula so that an error message is generated when a non-existent number is entered.

The same with VLOOKUP in other sheet named Sheet2

=VLOOKUP(A1;**‘Sheet2’**.D1:E100;2)