Workbook A-Z Function Reference

This document (© Visual Components Inc.) provides an alphabetical reference for the worksheet functions used in the FilmStar/Scantraq Workbook. The 32-bit version may include additional functions.

ABS	**INDEX**	**PROPER**
ACOS	**INDIRECT**	**PV**
ACOSH	**INT**	**RAND**
ADDRESS	**IPMT**	**RATE**
AND	**IRR**	**REPLACE**
ASIN	**ISBLANK**	**REPT**
ASINH	**ISERR**	**RIGHT**
ATAN	**ISERROR**	**ROUND**
ATAN2	**ISLOGICAL**	**ROW**
ATANH	**ISNA**	**ROWS**
AVERAGE	**ISNONTEXT**	**SEARCH**
CALL	**ISNUMBER**	**SECOND**
CEILING	**ISREF**	**SIGN**
CHAR	**ISTEXT**	**SIN**
CHOOSE	**LEFT**	**SINH**
CLEAN	**LEN**	**SLN**
CODE	**LN**	**SQRT**
COLUMN	**LOG**	**STDEV**
COLUMNS	**LOG10**	**STDEVP**
COS	**LOOKUP**	**SUBSTITUTE**
COSH	**LOWER**	**SUM**
COUNT	**MATCH**	**SUMSQ**
COUNTA	**MAX**	**SYD**
DATE	**MID**	**T**
DATEVALUE	**MIN**	**TAN**
DAY	**MINUTE**	**TANH**
DB	**MIRR**	**TEXT**
DDB	**MOD**	**TIME**
DOLLAR	**MONTH**	**TIMEVALUE**
ERROR.TYPE	**N**	**TODAY**
EVEN	**NA**	**TRIM**
EXACT	**NOT**	**TRUE**
EXP	**NOW**	**TRUNC**
FACT	**NPER**	**TYPE**
FALSE	**NPV**	**UPPER**
FIND	**ODD**	**VALUE**
FIXED	**OFFSET**	**VAR**
FLOOR	**OR**	**VARP**
FV	**PI**	**VDB**
HLOOKUP	**PMT**	**VLOOKUP**
HOUR	**PPMT**	**WEEKDAY**
IF	**PRODUCT**	**YEAR**

ABS

Description Returns the absolute value of a number.

Syntax **ABS**(*number*)

number is any integer.

Remarks An absolute value does not display a positive or negative sign.

See Also **SIGN** function

Examples
`ABS(-1)` returns 1
`ABS(1)` returns 1

ACOS

Description Returns the arc cosine of a number.

Syntax **ACOS**(*number*)

number is the cosine of the angle. The cosine can range from 1 to -1.

Remarks The resulting angle is returned in radians (from 0 to π).

See Also **COS** and **PI** functions

Examples
`ACOS(.5)` returns 1.05
`ACOS(-.2)` returns 1.77

ACOSH

Description Returns the inverse hyperbolic cosine of a number.

Syntax **ACOSH**(*number*)

number is any number equal to or greater than 1.

See Also **ASINH**, **ATANH**, and **COSH** functions

Examples
`ACOSH(1.2)` returns .62
`ACOSH(3)` returns 1.76

ADDRESS

Description Creates a cell address as text.

Syntax **ADDRESS**(*row*, *column*, *ref_type* [, *a1*] [, *sheet*])

row is the row number for the cell address.

column is the column number for the cell address.

ref_type is the cell reference type. The following table lists the values for this argument.

Argument	Reference type
1	Absolute
2	Absolute row, relative column
3	Relative row, absolute column

4 Relative

a1 is the reference format. This argument must be TRUE() to represent an A1 reference format; Formula One does not support the R1C1 reference format.

sheet is the name of an external spreadsheet. Omitting this argument assumes that the reference exists in the current spreadsheet.

See Also **COLUMN**, **OFFSET**, and **ROW** functions

Examples `ADDRESS(5, 6, 1)` returns "F5"
`ADDRESS(5, 6, 4, TRUE(), "SALES.VTS")` returns "SALES.VTS!F5"

AND

Description Returns True if all arguments are true; returns False if at least one argument is false.

Syntax **AND**(*logical_list*)

logical_list is a list of conditions separated by commas. You can include as many as 30 conditions in the list. The list can contain logical values or a reference to a range containing logical values. Text and empty cells are ignored. If there are no logical values in the list, #VALUE! is returned.

See Also **IF** , **NOT**, and **OR** functions

Examples `AND(1+1=2, 5+5=10)` returns True because both arguments are true.
`AND(TRUE(), FALSE())` returns False

ASIN

Description Returns the arcsine of a number.

Syntax **ASIN**(*number*)

number is the sine of the resulting angle, ranging from -1 to 1.

Remarks The resulting angle is returned in radians (ranging from $-\pi/2$ to $\pi/2$).

See Also **ASINH**, **PI** , and **SIN** functions

Examples `ASIN(-1)` returns -1.57
`ASIN(.4)` returns .41

ASINH

Description Returns the inverse hyperbolic sine of a number.

Syntax ASINH(*number*)

number is any number.

See Also **ACOSH**, **ASIN**, **ATANH**, and **SINH** functions

Examples ASINH(5.3) returns 2.37

ASINH(-4) returns -2.09

ATAN

Description Returns the arctangent of a number.

Syntax **ATAN**(*number*)

number is the tangent of the angle.

Remarks The resulting angle is returned in radians, ranging from -π/2 τo π/2. To find the result in degrees, multiply the result by 180/PI().

See Also **ATAN2**, **ATANH**, **PI** , and **TAN** functions

Examples
`ATAN(3.5)` returns 1.29
`ATAN(-4)` returns -1.33

ATAN2

Description Returns the arctangent of the specified coordinates.

Syntax **ATAN2**(*x*, *y*)

x is the x coordinate.

y is the y coordinate.

Remarks The arctangent is the angle from the x axis to a line with end points at the origin (0, 0) and a point with the given coordinates (x, y). The angle is returned in radians, ranging from -π τo π, excluding -π.

See Also **ATAN**, **ATANH**, **PI** , and **TAN** functions

Examples
`ATAN2(3, 6)` returns 1.11
`ATAN2(-1, .1)` returns 3.04

ATANH

Description Returns the inverse hyperbolic tangent of a number.

Syntax **ATANH**(*number*)

number is a number between -1 and 1, excluding -1 and 1.

See Also **ACOS**, **ASINH**, and **TANH** functions

Examples
`ATANH(.5)` returns .55
`ATANH(-.25)` returns -.26

AVERAGE

Description Returns the average of the supplied numbers. The result of **AVERAGE** is also known as the arithmetic mean.

Syntax **AVERAGE**(*number_list*)

number_list is a list of numbers separated by commas. As many as 30 numbers can be included in the list, and the list can contain numbers or a reference to a range that contains numbers. Text, logical expressions, or empty cells in a referenced range are ignored. All numeric values (including 0) are used.

See Also **MIN** and **MAX** functions

Examples `AVERAGE(5, 6, 8, 14)` returns 8.25
`AVERAGE(C15:C17)` returns 134; C15:C17 contains 24,144, and 234

CALL

Description Calls a custom function in a dynamic linked library (DLL).

Syntax **CALL**(*file_name*, *func_name*, *data_type*, *argument_list*)

file_name is the name of the DLL that contains the custom function. The file name should be provided as a quoted text string. You can also provide the path for the file.

func_name is the name of the custom function to be called from the DLL. The function name should be provided as a quoted text string.

argument_list is the list of arguments supplied to the custom function.

data_type is the data type, as a quoted text string, of the arguments and return value of the custom function. The following table lists the data type codes that can be used for this argument.

Data type	Description	Pass by	C declaration
A	Logicial (False =0, True =1)	Value	`short int`
B	IEEE 8-byte floating point number	Value	`double`
C	Null-terminated string (255 characters maximum)	Reference	`char*`
D	Byte-counted string (first byte contains string length; 255 characters maximum)	Reference	`unsigned char *`
E	IEEE 8-byte floating point number	Reference	`double*`
F	Null-terminated string (255 characters maximum)	Reference	`char*`
G	Byte-counted string (first byte contains string length; 255 characters maximum)	Reference	`unsigned char*`
H	Unsigned 2-byte integer	Value	`unsigned short int`
I	Signed 2-byte integer	Value	`short int`
J	Signed 4-byte integer	Value	`long int`
L	Logical (False=0, True =1)	Reference	`short int*`
M	Signed 2-byte integer	Reference	`short int*`
N	Signed 4-byte integer	Reference	`long int*`

Remarks For declarations made in C, it is assumed that your compiler defaults to 8-byte doubles, 2-byte short integers, and 4-byte long integers. In the Windows programming environment, all pointers should be far pointers.

Pascal calling conventions are used for all functions called from DLLs. For most C compilers, you must add the --Pascal keyword to the function declaration.

If the return value for your custom function uses a pass-by-reference data type, a null pointer can be passed as the return value. The null pointer is interpreted as the #NUM! error value.

For F and G data types, a custom function can modify an allocated string buffer. If the return value type code is F or G, the value returned by the function is ignored. The list of function arguments is searched for the first data type that corresponds to the return value type. The current contents of the allocated string buffer is taken for the return value. 256 bytes is allocated for the argument; therefore, a function can return a larger string than it receives.

You can use a single digit (n), with a value from 1 to 9, as the code for data_type. The variable in the location pointed to by the nth argument is modified instead of the return value; this process is referred to as modifying in place. The nth argument must be a pass-by-reference data type. In addition, you must declare the function void. For most C compilers, you can add the Void keyword to the function declaration.

Example `CALL("\VTFORM1\DEMO4\CUSTFUNC.DLL", "Quotient", "BBB", 3, 2)`

CEILING

Description Rounds a number up to the nearest multiple of a specified significance.

Syntax **CEILING**(*number*, *significance*)

number is the value to round.

significance is the multiple to which to round.

Remarks Regardless of the sign of the number, the value is rounded up, away from zero. If number is an exact multiple of significance, no rounding occurs.

If number or significance is non-numeric, #VALUE! is returned. When the arguments have opposite signs, #NUM! is returned.

See Also **EVEN**, **FLOOR**, **INT**, **ODD**, **ROUND**, and **TRUNC** functions

Examples `CEILING(1.23459, .05)` returns 1.25
`CEILING(-148.24, -2)` returns -150

CHAR

Description Returns a character that corresponds to the supplied ANSI code.

Syntax **CHAR**(*number*)

number is a value between 1 and 255 that specifies an ANSI character.

Remarks The character and associated numeric code are defined by Windows in the ANSI character set.

See Also **CODE** function

Examples `CHAR(70)` returns F
`CHAR(35)` returns #

CHOOSE

Description Returns a value from a list of numbers based on the index number supplied.

Syntax **CHOOSE**(*index*, *item_list*)

index is a number that refers to an item in item_list.

- *index* can be a cell reference. index can also be a formula that returns any value from 1 to 29.
- If *index* is less than 1 or greater than the number of items in item_list, #VALUE! is returned.
- If *index* is a fractional number, it is truncated to an integer.

item_list is a list of numbers, formulas, or text separated by commas. This argument can also be a range reference. You can specify as many as 29 items in the list.

See Also **INDEX** function

Examples `CHOOSE(2,"Q1", "Q2", "Q3", "Q4")` returns "Q2"

`AVERAGE(CHOOSE(1, A1:A10, B1:B10, C1:C10))` returns the average of the contents of range A1:A10.

CLEAN

Description Removes all non-printable characters from the supplied text.

Syntax **CLEAN**(*text*)

text is any worksheet information.

Remarks Text that is imported from another environment may require this function.

See Also **CHAR** and **TRIM** functions

Example `CLEAN("Payments " & CHAR(8) & "Due")` returns Payments Due because the character returned by CHAR(8) is non-printable.

CODE

Description Returns a numeric code representing the first character of the supplied string.

Syntax **CODE**(*text*)

text is any string.

Remarks The numeric code and associated string are defined in your computer's character set. The character set used by Windows is the ANSI character set.

See Also **CHAR** function

Examples `CODE("A")` returns 65
`CODE("b")` returns 98

COLUMN

Description Returns the column number of the supplied reference.

Syntax **COLUMN**(*reference*)

reference is a reference to a cell or range. Omitting the argument returns the number of the column in which **COLUMN** is placed.

See Also **COLUMNS** and **ROW** functions

Examples `COLUMN(B3)` returns 2
`COLUMN()` returns 4 if the function is entered in cell D2.

COLUMNS

Description Returns the number of columns in a range reference.

Syntax **COLUMNS**(*range*)

range is a reference to a range of cells.

See Also **COLUMN** and **ROWS** functions

Example `COLUMNS(A1:D5)` returns 4

COS

Description Returns the cosine of an angle.

Syntax **COS**(*number*)

number is any number.

See Also **ACOS**, **ASINH**, **ATANH**, **COSH**, and **PI** functions

Examples `COS(1.444)` returns .126
`COS(5)` returns .28

COSH

Description Returns the hyperbolic cosine of a number.

Syntax **COSH**(*number*)

number is any number.

See Also **ASINH**, **ATANH**, and **COS** functions

Examples `COSH(2.10)` returns 4.14
`COSH(.24)` returns 1.03

COUNT

Description Returns the number of values in the supplied list.

Syntax **COUNT**(*value_list*)

value_list is a list of values. The list can contain as many as 30 values.

Remarks **COUNT** only numerates numbers or numerical values (e.g., logical values, dates, or text representations of dates). If you supply a range, only numbers and numerical values in the range are counted. Empty cells, logical values, text, and error values in the range are ignored.

See Also **AVERAGE**, **COUNTA**, and **SUM** functions

Examples `COUNT(5, 6, "Q2")` returns 2
`COUNT("03/06/94", "06/21/94", "10/19/94")` returns 3

COUNTA

Description Returns the number of non-blank values in the supplied list.

Syntax **COUNTA**(*expression_list*)

expression_list is a list of expressions. As many as 30 expressions can be included in the list.

Remarks **COUNTA** returns the number of cells that contain data in a range. Null values ("") are counted, but references to empty cells are ignored.

See Also **AVERAGE**, **COUNT**, **PRODUCT**, and **SUM** functions

Examples `COUNTA(32, 45, "Earnings", "")` returns 4

`COUNTA(C38:C40)` returns 0 when the specified range contains empty cells

DATE

Description Returns the serial number of the supplied date.

Syntax **DATE**(*year*, *month*, *day*)

year is a number from 1900 to 2078. If year is between 1920 to 2019, you can specify two digits to represent the year; otherwise specify all four digits.

month is a number representing the month (e.g., 12 represents December). If a number greater than 12 is supplied, the number is added to the to the first month of the specified year.

day is a number representing the day of the month. If the number you specify for day exceeds the number of days in that month, the number is added to the first day of the specified month.

See Also **DATEVALUE**, **DAY**, **MONTH**, **NOW**, **TIMEVALUE**, **TODAY**, and **YEAR** functions

Examples `DATE(94, 6, 21)` returns 34506
`DATE(99, 3, 6)` returns 36225

DATEVALUE

Description Returns the serial number of a date supplied as a text string.

Syntax **DATEVALUE**(*text*)

text is a date, in text format, between January 1, 1900, and December 31, 2078. If you omit the year, the current year is used.

See Also **NOW**, **TIMEVALUE**, and **TODAY** functions

Examples `DATEVALUE("3/6/94")` returns 34399
`DATEVALUE("12/25/95")` returns 35058

DAY

Description Returns the day of the month that corresponds to the date represented by the supplied number.

Syntax **DAY**(*serial_number*)

serial_number is a date represented as a serial number or as text (e.g., "06-21-94" or "21-Jun-94").

See Also **HOUR**, **MINUTE**, **MONTH**, **NOW, SECOND**, **TODAY**, **WEEKDAY**, and **YEAR** functions

Examples `DAY(34399)` returns 6
`DAY("06-21-94")` returns 21

DB

Description Returns the real depreciation of an asset for a specific period of time using the fixed-declining balance method.

Syntax **DB**(*cost*, *salvage*, *life*, *period* [, *months*])

cost is the initial cost of the asset.

salvage is the salvage value of the asset.

life is the number of periods in the useful life of the asset.

period is the period for which to calculate the depreciation. The time units used to determine period and life must match.

months is the number of months in the first year of the item's life. Omitting this argument assumes there are 12 months in the first year.

See Also **DDB**, **SLN**, **SYD**, and **VDB** functions

Example `DB(10000, 1000, 7, 3)` returns 1451.52

DDB

Description Returns the depreciation of an asset for a specific period of time using the double-declining balance method or a declining balance factor you supply.

Syntax **DDB**(*cost*, *salvage*, *life*, *period* [, *factor*])

cost is the initial cost of the asset.

salvage is the salvage value of the asset.

life is the number of periods in the useful life of the asset.

period is the period for which to calculate the depreciation. The time units used to determine period and life must match.

factor is the rate at which the balance declines. Omitting this argument assumes a default factor of 2, the double-declining balance factor.

Remarks The double-declining balance method uses an accelerated rate where the highest depreciation occurs in the first period, decreasing in successive periods.

All arguments for this function must be positive numbers.

See Also **DB** , **SLN**, **SYD**, and **VDB** functions

Example `DDB(10000,1000, 7, 3)` returns 1457.73

DOLLAR

Description Returns the specified number as text, using currency format and the supplied precision.

Syntax **DOLLAR**(*number* [, *precision*])

number is a number, a formula that evaluates to a number, or a reference to a cell that contains a number.

precision is a value representing the number of decimal places to the right of the decimal point. Omitting this argument assumes two decimal places.

See Also **FIXED**, **TEXT**, and **VALUE** functions

Examples `DOLLAR(1023.789)` returns "$1023.79"
`DOLLAR(495.301, -2)` returns "$500"

ERROR.TYPE

Description Returns a number corresponding to an error.

Syntax **ERROR.TYPE**(*error_ref*)

error_ref is a cell reference.

Remarks The following table lists the error text and associated error numbers returned by this function.

Number	Error text
1	#NULL!
2	#DIV/0!
3	#VALUE!
4	#REF!
5	#NAME?
6	#NUM!
7	#N/A
#N/A	Other

See Also **ISERR** and **ISERROR** functions

Example `ERROR.TYPE(A1)` returns 2 if the formula in cell A1 attempts to divide by zero.

EVEN

Description Rounds the specified number up to the nearest even integer.

Syntax **EVEN**(*number*)

number is any number, a formula that evaluates to a number, or a reference to a cell that contains a number.

See Also **CEILING**, **FLOOR**, **INT**, **ODD**, **ROUND**, and **TRUNC** functions

Examples `EVEN(2.5)` returns 4
`EVEN(2030.45)` returns 2032

EXACT

Description Compares two expressions for identical, case-sensitive matches. True is returned if the expressions are identical; False is returned if they are not.

Syntax **EXACT**(*expression1*, *expression2*)

expression1 is any text.

expression2 is any text.

See Also **LEN** and **SEARCH** functions

Examples `EXACT("Match", "Match")` returns True
`EXACT("Match", "match")` returns False

EXP

Description Returns e raised to the specified power. The constant e is 2.71828182845904 (the base of the natural logarithm).

Syntax **EXP**(*number*)

number is any number as the exponent.

See Also **LN** and **LOG** functions

Examples `EXP(2.5)` returns 12.18
`EXP(3)` returns 20.09

FACT

Description Returns the factorial of a specified number.

Syntax **FACT**(*number*)

number is any non-negative integer. If you supply a real number, **FACT** truncates the number to an integer before calculation.

See Also **PRODUCT** function

Examples `FACT(2.5)` returns 2
`FACT(6)` returns 720

FALSE

Description Returns the logical value False. This function always requires the trailing parentheses.

Syntax **FALSE**()

See Also **TRUE** function

FIND

Description Searches for a string of text within another text string and returns the character position at which the search string first occurs.

Syntax **FIND**(*search_text*, *text* [, *start_position*])

search_text is the text to find. If you specify an empty string (""), **FIND** matches the first character in text.

text is the text to be searched.

start_position is the character position in text where the search begins. The first character in text is character number 1. When you omit this argument, the default starting position is character number 1.

Remarks **FIND** is case-sensitive. You cannot use wildcard characters in the *search_text*.

See Also **EXACT**, **LEN**, **MID**, and **SEARCH** functions

Examples `FIND("time", "There's no time like the present")` returns 12
`FIND("4", "Aisle 4, Part 123-4-11", 9)` returns 19

FIXED

Description Rounds a number to the supplied precision, formats the number in decimal format, and returns the result as text.

Syntax **FIXED**(*number* [, *precision*][, *no_commas*])

number is any number.

precision is the number of digits that appear to the right of the decimal place. When this argument is omitted, a default precision of 2 is used. If you specify negative precision, number is rounded to the left of the decimal point. You can specify a precision as great as 127 digits.

no_commas determines if thousands separators (commas) are used in the result. Use 1 to exclude commas in the result. If no_commas is 0 or the argument is omitted, thousands separators are included (e.g., 1,000.00).

See Also **DOLLAR**, **ROUND**, **TEXT**, and **VALUE** functions

Examples `FIXED(2000.5, 3)` returns "2,000.500"
`FIXED(2009.5, -1, 1)` returns "2010"

FLOOR

Description Rounds a number down to the nearest multiple of a specified significance.

Syntax **FLOOR**(*number*, *significance*)

number is the value to round.

significance is the multiple to which to round.

Remarks Regardless of the sign of the number, the value is rounded down, toward zero. If number is an exact multiple of significance, no rounding occurs.

If number or significance is non-numeric, #NAME? is returned. When the arguments have opposite signs, #NUM! is returned.

See Also **CEILING**, **EVEN**, **INT**, **ODD**, **ROUND**, and **TRUNC** functions

Examples `FLOOR(1.23459, .05)` returns 1.2
`FLOOR(-148.24, -2)` returns -148

FV

Description Returns the future value of an annuity based on regular payments and a fixed interest rate.

Syntax **FV**(*interest*, *nper*, *payment* [, *pv*] [, *type*])

interest is the fixed interest rate.

nper is the number of payments in an annuity.

payment is the fixed payment made each period.

pv is the present value, or the lump sum amount, the annuity is currently worth. When you omit this argument, a present value of 0 is assumed.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

Remarks The units used for interest must match those used for *nper*. For example, if the annuity has an 8% annual interest rate over a period of 5 years, specify 8%/12 for interest and 5*12 for *nper*.

Cash paid out, such as a payment, is shown as a negative number. Cash received, such as a dividend check, is shown as a positive number.

See Also **IPMT**, **NPER**, **PMT**, **PPMT**, **PV** , and **RATE** functions

Examples `FV(5%, 8, -500)` returns 4,774.55
`FV(10%/12, 240, -700, 1)` returns 531,550.86

HLOOKUP

Description Searches the top row of a table for a value and returns the contents of a cell in that table that corresponds to the location of the search value.

Syntax **HLOOKUP**(*search_item*, *search_range*, *row_index*)

search_item is a value, text string, or reference to a cell containing a value that is matched against data in the top row of *search_range*.

search_range is a reference to the range (table) to be searched. The cells in the first row of *search_range* can contain numbers, text, or logical values. The contents of the first row must be in ascending order (e.g., -2, -1, 0, 2...A through Z, False, True). Text searches are not case-sensitive.

row_index is the row in search_range from which the matching value is returned.

- *row_index* can be a number from 1 to the number of rows in *search_range*.
- If *row_index* is less than 1, #VALUE! is returned.
- When *row_index* is greater than the number of rows in the table, #REF! is returned.

Remarks **HLOOKUP** compares the information in the top row of *search_range* to the supplied search_item. When a match is found, information located in the same column and supplied row (*row_index*) is returned.

If *search_item* cannot be found in the top row of *search_range*, the largest value that is less than *search_item* is used. When *search_item* is less than the smallest value in the first row of the *search_range*, #REF! is returned.

See Also **INDEX**, **LOOKUP**, **MATCH**, and **VLOOKUP** functions

	A	B	C	D	E
1		Midwest	Northeast	Pacific	South
2	Q1	48.23	278.21	61.97	164.80
3	Q2	163.83	22.63	161.73	183.96
4	Q3	43.96	233.56	278.16	171.98
5	Q4	245.69	167.09	245.23	163.00

Examples In the preceding worksheet:

`HLOOKUP("Northeast", B1:E5, 3)` returns 22.63
`HLOOKUP("Pacific", B1:E5, 7)` returns #REF!

HOUR

Description Returns the hour component of the specified time in 24-hour format.

Syntax **HOUR**(*serial_number*)

serial_number is the time as a serial number. The decimal portion of the number represents time as a fraction of the day.

Remarks The result is an integer ranging from 0 (12:00 AM) to 23 (11:00 PM).

See Also **DAY**, **MINUTE**, **MONTH**, **NOW**, **SECOND**, **WEEKDAY**, and **YEAR** functions

Examples `HOUR(34259.4)` returns 9
`HOUR(34619.976)` returns 23

IF

Description Tests the condition and returns the specified value.

Syntax **IF**(*condition*, *true_value*, *false_value*)

condition is any logical expression.

true_value is the value to be returned if condition evaluates to True.

false_value is the value to be returned if condition evaluates to False.

See Also **AND**, **FALSE**, **NOT**, **OR** , and **TRUE** functions

Example `IF(A1>10, "Greater", "Less")` returns Greater if the contents of A1 is greater than 10 and Less if the contents of A1 is less than 10.

INDEX

Description Returns the contents of a cell from a specified range.

Syntax **INDEX**(*reference* [, *row*] [, *column*] [, *range_number*])

reference is a reference to one or more ranges.

- If *reference* specifies more than one range, separate each reference with a comma and enclose *reference* in parentheses (e.g., (A1:C6, B7:E14, F4)).
- If each range in *reference* contains only one row or column, you can omit the row or column argument. For example, if *reference* is A1:A15, you can omit the column argument (e.g., INDEX(A1:A15, 3,, 1)).

row is the row number in *reference* from which to return data.

column is column number in *reference* from which to return data.

range_number specifies the range from which data is returned if *reference* contains more than one range. For example, if *reference* is (A1:A10, B1:B5, D14:E23), A1:A10 is *range_number* 1, B1:B5 is *range_number* 2, and D14:E23 is *range_number* 3.

Remarks If *row*, *column*, and *range_number* do not point to a cell within reference, #REF! is returned. If *row* and *column* are omitted, **INDEX** returns the range in reference specified by *range_number*.

See Also **CHOOSE**, **HLOOKUP**, **LOOKUP**, **MATCH**, and **VLOOKUP** functions

	A	B	C	D	E
1	Sales Group 1			Sales Group 2	
2	Adams	$1,225.14		Cash	$1,819.47
3	Baker	$1,415.35		Johnson	$1,733.67
4	Martinez	$1,573.57		Nelson	$1,138.23
5	Smith	$1,469.78		Randall	$1,634.58
6	White	$1,390.89		Schultz	$1,093.82

Examples In the preceding worksheet:

`INDEX(A2:B6, 2, 2)` returns $1415.35
`INDEX((A2:B6, D2:E6), 4, 2, 2)` returns $1634.58

INDIRECT

Description Returns the contents of the cell referenced by the specified cell.

Syntax **INDIRECT**(*ref_text* [, *a1*])

ref_text is a reference to a cell that references a third cell. If *ref_text* is not a valid reference, #REF! is returned.

a1 is the reference format. This argument must be TRUE() to represent an A1 reference format; Formula One does not support the R1C1 reference format.

See Also **OFFSET** function

Example `INDIRECT(C1)` returns the contents of the cell that C1 references. If C1 contains “D1”, the contents of D1 is returned by INDIRECT.

INT

Description Rounds the supplied number down to the nearest integer.

Syntax **INT**(*number*)

number is any real number.

See Also **CEILING**, **FLOOR**, **MOD**, **ROUND**, and **TRUNC** functions

Examples
`INT(10.99)` returns 10
`INT(-10.99)` returns -11

IPMT

Description Returns the interest payment of an annuity for a given period, based on regular payments and a fixed periodic interest rate.

Syntax **IPMT**(*interest*, *per*, *nper*, *pv*, [*fv*], [*type*])

interest is the fixed periodic interest rate.

per is the period for which to return the interest payment. This number must be between 1 and *nper*.

nper is the number of payments.

pv is the present value, or the lump sum amount the annuity is currently worth.

fv is the future value, or the value after all payments are made. If this argument is omitted, the future value is assumed to be 0.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

Remarks The units used for interest must match those used for *nper*. For example, if the annuity has an 8% annual interest rate over a period of 5 years, specify 8%/12 for interest and 5*12 for *nper*.

Cash paid out, such as a payment, is shown as a negative number. Cash received, such as a dividend check, is shown as a positive number.

See Also **FV** , **PMT**, **PPMT**, and **RATE** functions

Examples
`IPMT(8%/12, 2, 48, 18000)` returns -117.87
`IPMT(8%/12, 2, 48, 18000, 0, 1)` returns -117.09

IRR

Description Returns internal rate of return for a series of periodic cash flows.

Syntax **IRR**(*cash_flow* [, *guess*])

cash_flow is a reference to a range that contains values for which to calculate the internal rate of return. The values must contain at least one positive and one negative value.

- During calculation, **IRR** uses the order in which the values appear to determine the order of the cash flow.
- Text, logical values, and empty cells in the range are ignored.

guess is the estimate of the internal rate of return. If no argument is supplied, a rate of return of 10 percent is assumed.

Remarks The internal rate of return is the interest rate received for an investment consisting of payments (specified by negative numbers) and investments (specified by positive numbers).

IRR is calculated iteratively, cycling through the calculation until the result is accurate to .00001 percent. If the result cannot be found after 20 iterations, #NUM! is returned. When this occurs, supply a different value for guess.

See Also **MIRR**, **NPV**, and **RATE** functions

	A	B
1	Investment	($60,000.00)
2	1989 income	$9,590.00
3	1990 income	$10,580.00
4	1991 income	$12,790.00
5	1992 income	$15,830.00
6	1993 income	$18,930.00

Examples In the preceding worksheet:

`IRR(B1:B6)` returns 3.72%
`IRR(B1:B3, -20%)` returns -49.26%

ISBLANK

Description Determines if the specified cell is blank.

Syntax **ISBLANK**(*reference*)

reference is a reference to any cell.

Remarks If the referenced cell is blank, True is returned. False is returned if the cell is not blank.

See Also **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Example `ISBLANK(A1)` returns True if A1 is a blank cell.

ISERR

Description Determines if the specified expression returns an error value.

Syntax **ISERR**(*expression*)

expression is any expression.

Remarks If the *expression* returns any error except #N/A!, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Example `ISERR(A1)` returns True if A1 contains a formula that returns an error (e.g., #NUM!).

ISERROR

Description Determines if the specified expression returns an error value.

Syntax **ISERROR**(*expression*)

expression is any expression.

Remarks If the *expression* returns any error value (e.g., #N/A!, #VALUE!, #REF!, #DIV/0!, #NUM!, #NAME?, or #NULL!), True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Examples `ISERROR(4/0)` returns True
`ISERROR(A1)` returns False if A1 contains a formula that does not return an error.

ISLOGICAL

Description Determines if the specified expression returns a logical value.

Syntax **ISLOGICAL**(*expression*)

expression is any expression.

Remarks If the expression returns a logical value, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Example `ISLOGICAL(ISBLANK(A1))` returns True because ISBLANK returns a logical value.

ISNA

Description Determines if the specified expression returns the value not available error.

Syntax **ISNA**(*expression*)

expression is any expression.

Remarks If the *expression* returns the #N/A! error, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Example `ISNA(A1)` returns True if cell A1 contains the NA() function or returns the error value #N/A!.

ISNONTEXT

Description Determines if the specified expression is not text.

Syntax **ISNONTEXT**(*expression*)

expression is any expression.

Remarks If the *expression* returns any value that is not text, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Examples `ISNONTEXT(F3)` returns True if cell F3 contains a number or is a blank cell.
`ISNONTEXT("text")` returns False.

ISNUMBER

Description Determines if the specified expression is a number.

Syntax **ISNUMBER**(*expression*)

expression is any expression.

Remarks If the *expression* returns a number, True is returned. Otherwise, False is returned. If expression returns a number represented as text (e.g., "12"), False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISREF**, and **ISTEXT** functions

Examples `ISNUMBER(123.45)` returns True
`ISNUMBER("123")` returns False

ISREF

Description Determines if the specified expression is a range reference.

Syntax **ISREF**(*expression*)

expression is any expression.

Remarks If the *expression* returns a range reference, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, and **ISTEXT** functions

Example `ISREF(A3)` returns True

ISTEXT

Description Determines if the specified expression is text.

Syntax **ISTEXT**(*expression*)

expression is any expression.

Remarks If the *expression* returns text, True is returned. Otherwise, False is returned.

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, and **ISREF** functions

Example `ISTEXT("2nd Quarter")` returns True

LEFT

Description Returns the leftmost characters from the specified text string.

Syntax **LEFT**(*text* [, *num_chars*])

text is any text string.

num_chars is the number of characters to return. This value must be greater than or equal to zero. If num_chars is greater than the number of characters in text, the entire string is returned. Omitting this argument assumes a value of 1.

See Also **MID** and **RIGHT** functions

Examples `LEFT("2nd Quarter")` returns "2"
`LEFT("2nd Quarter", 3)` returns "2nd"

LEN

Description Returns the number of characters in the supplied text string.

Syntax **LEN**(*text*)

text in any text string. Spaces in the string are counted as characters.

See Also **EXACT** and **SEARCH** functions

Examples `LEN("3rd Quarter")` returns 11
`LEN("1-3")` returns 3

LN

Description Returns the natural logarithm (based on the constant e) of a number.

Syntax **LN**(*number*)

number is any positive real number.

Remarks **LN** is the inverse of the **EXP** function.

See Also **EXP**, **LOG**, and **LOG10** functions

Examples `LN(12.18)` returns 2.50
`LN(20.09)` returns 3.00

LOG

Description Returns the logarithm of a number to the specified base.

Syntax **LOG**(*number* [, *base*])

text is any positive real number.

base is the base of the logarithm. Omitting this argument assumes a base of 10.

See Also **EXP**, **LN**, and **LOG10** functions

Examples `LOG(1)` returns 0
`LOG(10)` returns 1

LOG10

Description Returns the base-10 logarithm of a number.

Syntax **LOG10**(*number*)

number is any positive real number.

See Also **EXP**, **LN**, and **LOG** functions

Examples
`LOG10(260)` returns 2.41
`LOG10(100)` returns 2

LOOKUP

Description Searches for a value in one range and returns the contents of the corresponding position in a second range.

Syntax **LOOKUP**(*lookup_value*, *lookup_range*, *result_range*)

lookup_value is the value for which to search in the first range.

lookup_range is the first range to search and contains only one row or one column.

- The range can contain numbers, text, or logical values.
- To search *lookup_range* correctly, the expressions in the range must be placed in ascending order (e.g., -2, -1, 0, 1, 2...A through Z, False, True). The search is not case-sensitive.

result_range is a range of one row or one column that is the same size as *lookup_range*.

Remarks If *lookup_value* does not have an exact match in *lookup_range*, the largest value that is less than or equal to *lookup_value* is found and the corresponding position in *result_range* is returned. When *lookup_value* is smaller than the data in *lookup_range*, #N/A is returned.

See Also **HLOOKUP**, **INDEX**, and **VLOOKUP** functions

	A	B
1	**Region**	**Headquarters**
2	Midwest	Kansas City
3	North	Detroit
4	Northeast	Philadelphia
5	Pacific	Portland
6	South	Atlanta
7	Southwest	Phoenix

Examples In the preceding worksheet:

`LOOKUP("North", A2:A7, B2:B7)` returns Detroit
`LOOKUP("Alabama", A2:A7, B2:B7)` returns #N/A

LOWER

Description Changes the characters in the specified string to lowercase characters. Numeric characters in the string are not changed.

Syntax **LOWER**(*text*)

text is any string.

See Also **PROPER** and **UPPER** functions

Examples `LOWER("3rd Quarter")` returns "3rd quarter"
`LOWER("JOHN DOE")` returns "john doe"

MATCH

Description A specified value is compared against values in a range. The position of the matching value in the search range is returned.

Syntax **MATCH**(*lookup_value*, *lookup_range*, *comparison*)

lookup_value is the value against which to compare. It can be a number, text, or logical value or a reference to a cell that contains one of those values.

lookup_range is the range to search and contains only one row or one column. The range can contain numbers, text, or logical values.

comparison is a number that represents the type of comparison to be made between *lookup_value* and the values in *lookup_range*. When you omit this argument, comparison method 1 is assumed.

- When comparison is 1, the largest value that is less than or equal to *lookup_value* is matched. When using this comparison method, the values in *lookup_range* must be in ascending order (e.g., ...-2, -1, 0, 1, 2..., A through Z, False, True).
- When comparison is 0, the first value that is equal to *lookup_value* is matched. When using this comparison method, the values in *lookup_range* can be in any order.
- When comparison is -1, the smallest value that is greater than or equal to *lookup_value is* matched. When using this comparison method, the values in *lookup_range* must be in descending order (e.g., True, False, Z through A, ...2, 1, 0, -1, -2...).

Remarks When using comparison method 0 and *lookup_value* is text, *lookup_value* can contain wildcard characters. The wildcard characters are * (asterisk), which matches any sequence of characters, and ? (question mark), which matches any single character.

When no match is found for *lookup_value*, #N/A is returned.

See Also **HLOOKUP**, **INDEX**, **LOOKUP**, and **VLOOKUP** functions

	A	B
1	Mfr. Code	Stock No.
2	BAJ	0677
3	DOD	0753
4	FMH	0816
5	JMR	0913
6	PLY	7534
7	TJL	7763

Examples In the preceding worksheet:

`MATCH(7600, B2:B7,1)` returns 5
`MATCH("D*", A2:A7,0)` returns 2

MAX

Description Returns the largest value in the specified list of numbers.

Syntax **MAX**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas.

- The list can contain numbers, logical values, text representations of numbers, or a reference to a range containing those values.
- Error values or text that cannot be translated into numbers return errors.
- If a range reference is included in the list, text, logical expressions, and empty cells in the range are ignored.
- If there are no numbers in the list, 0 is returned.

See Also **AVERAGE** and **MIN** functions

Examples `MAX(50, 100, 150, 500, 200)` returns 500
`MAX(A1:F12)` returns the largest value in the range

MID

Description Returns the specified number of characters from a text string, beginning with the specified starting position.

Syntax **MID**(*text*, *start_position*, *num_chars*)

text is the string from which to return characters.

- *start_position* is the position of the first character to return from text.
- If *start_position* is 1, the first character in text is returned.
- If *start_position* is greater than the number of characters in text, an empty string ("") is returned.
- If *start_position* is less than 1, #VALUE! is returned.

num_chars is the number of characters to return. If num_chars is negative, #VALUE! is returned.

Remarks If *start_position* plus the number of characters in num_chars exceeds the length of text, the characters from start_position to the end of text are returned.

See Also **CODE**, **FIND**, **LEFT**, **RIGHT**, and **SEARCH** functions

Examples `MID("Travel Expenses", 8, 8)` returns "Expenses"
`MID("Part #45-7234", 7, 2)` returns 45

MIN

Description Returns the smallest value in the specified list of numbers.

Syntax **MIN**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas.

- The list can contain numbers, logical values, text representations of numbers, or a reference to a range containing those values.
- Error values or text that cannot be translated into numbers return errors.
- If a range reference is included in the list, text, logical expressions, and empty cells in the range are ignored.

- If there are no numbers in the list, 0 is returned.

See Also **AVERAGE** and **MAX** functions

Examples `MIN(50, 100, 150, 500, 200)` returns 50
`MIN(A1:F12)` returns the smallest value in the range

MINUTE

Description Returns the minute that corresponds to the supplied date.

Syntax **MINUTE**(*serial_number*)

serial_number is the time as a serial number. The decimal portion of the number represents time as a fraction of the day.

Remarks The result is an integer ranging from 0 to 59.

See Also **DAY**, **HOUR**, **MONTH**, **NOW**, **SECOND**, **WEEKDAY**, and **YEAR** functions

Examples `MINUTE(34506.4)` returns 36
`MINUTE(34399.825)` returns 48

MIRR

Description Returns the modified internal rate of return for a series of periodic cash flows.

Syntax **MIRR**(*cash_flows*, *finance_rate*, *reinvest_rate*)

cash_flow is a reference to a range that contains values for which to calculate the modified internal rate of return. The values must contain at least one positive and one negative value.

- During calculation, **MIRR** uses the order in which the values appear to determine the order of cash flow.
- Values that represent cash received should be positive; negative values represent cash paid.
- Text, logical values, and empty cells in the range are ignored.

finance_rate is the interest rate paid on money used in the cash flow.

reinvest_rate is the interest rate received on money reinvested from the cash flow.

Remarks The modified internal rate of return considers the cost of the investment and the interest received on the reinvestment of cash.

See Also **IRR**, **NPV**, and **RATE** functions

	A	B
1	Investment	($60,000.00)
2	1989 income	$9,590.00
3	1990 income	$10,580.00
4	1991 income	$12,790.00
5	1992 income	$15,830.00
6	1993 income	$18,930.00

Examples In the preceding worksheet:

`MIRR(B1:B6, 12%, 8%)` returns 5.20%
`MIRR(B1:B3, 12%, 8%)` returns -40.93%

MOD

Description Returns the remainder after dividing a number by a specified divisor.

Syntax **MOD**(*number*, *divisor*)

number is any number.

divisor is any non-zero number. If *divisor* is 0, #DIV/0! is returned.

See Also **INT**, **ROUND**, and **TRUNC** functions

Examples `MOD(-23, 3)` returns 1
`MOD(-23, -3)` returns -2

MONTH

Description Returns the month that corresponds to the supplied date.

Syntax **MONTH**(*serial_number*)

serial_number is the date as a serial number or as text (e.g., “06-21-94” or “21-Jun-94”).

Remarks **MONTH** returns a number ranging from 1 (January) to 12 (December).

See Also **DAY**, **HOUR**, **MINUTE**, **NOW**, **SECOND**, **TODAY**, **WEEKDAY**, and **YEAR** functions

Examples `MONTH(“06-21-94”)` returns 6
`MONTH(34626)` returns 10

N

Description Tests the supplied value and returns the value if it is a number.

Syntax **N**(*value*)

value is a value or a reference to a cell containing a value to test.

Remarks Numbers are returned as numbers, serial numbers formatted as dates are returned as serial numbers, and the logical function TRUE() is returned as 1. All other expressions return 0.

See Also **T** and **VALUE** functions

Examples `N(32467)` returns 32467
`N(A4)` returns 1 if A4 contains the logical function True

NA

Description Returns the error value #N/A, which represents “not available.”

Syntax **NA**()

Remarks Use **NA** to mark cells that lack data without leaving them empty. Empty cells may not be correctly represented in some calculations.

Although **NA** does not use arguments, you must supply the empty parentheses to correctly reference the function.

See Also **ISNA** function

NOT

Description Returns a logical value that is the opposite of its value.

Syntax **NOT**(*logical*)

logical is an expression that returns a logical value (e.g., True or False).

Remarks If logical is false, **NOT** returns True. Conversely, if logical is true, **NOT** returns False.

See Also **AND**, **IF** , and **OR** functions

Examples `NOT(TRUE())` returns False
`NOT(MONTH("12/25/94") = 12)` returns False

NOW

Description Returns the current date and time as a serial number.

Syntax **NOW**()

Remarks In a serial number, numbers to the left of the decimal point represent the date; numbers to the right of the decimal point represent the time. The result of this function changes only when a recalculation of the worksheet occurs.

See Also **DATE**, **DAY**, **HOUR**, **MINUTE**, **MONTH**, **SECOND**, **TODAY**, **WEEKDAY**, and **YEAR** functions

NPER

Description Returns the number of periods of an investment based on regular periodic payments and a fixed interest rate.

Syntax **NPER**(*interest*, *pmt*, *pf* [, *fv*] [, *type*])

interest is the fixed interest rate.

pmt is the fixed payment made each period. Generally, *pmt* includes the principle and interest, not taxes or other fees.

pf is the present value, the lump-sum amount that a series of future payments is currently worth.

fv is the future value, the balance to attain after the final payment. Omitting this argument assumes a future balance of 0.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

See Also **FV** , **IPMT**, **PMT**, **PPMT**, **PV** , and **RATE** functions

Examples `NPER(12%/12, -350, -300, 16000, 1)` returns 36.67
`NPER(1%, -350, -300, 16000)` returns 36.98

NPV

Description Returns the net present value of an investment based on a series of periodic payments and a discount rate.

Syntax **NPV**(*discount_rate*, *value_list*)

discount_rate is the rate of discount for one period.

value_list is a list of as many as 29 arguments or a reference to a range that contains values that represent payments and income.

- During calculation, **NPV** uses the order in which the values appear to determine the order of cash flow.
- Numbers, empty cells, and text representations of numbers are included in the calculation. Errors and text that cannot be translated into numbers are ignored.
- If *value_list* is a range reference, only numeric data in the range is included in the calculation. Other types of data in the range (e.g., empty cells, logical values, text, and error values) are ignored.

Remarks The time span **NPV** uses for calculation begins one period before the first cash flow date and ends when the last cash flow payment is made. This function is based on future cash flows. When your first cash flow occurs at the beginning of the first period, the first value must be added to the **NPV** result, not supplied as a value in *value_list*.

See Also **FV** , **IRR**, and **PV** functions

Example `NPV(8%, -12000, 3000, 3000, 3000, 7000)` returns 811.57

ODD

Description Rounds the specified number up to the nearest odd integer.

Syntax **ODD**(*number*)

number is any number, a formula that evaluates to a number, or a reference to a cell that contains a number.

See Also **CEILING**, **EVEN**, **FLOOR**, **INT**, **ROUND**, and **TRUNC** functions

Examples `ODD(3.5)` returns 5
`ODD(6)` returns 7

OFFSET

Description Returns the contents of a range that is offset from a starting point in the spreadsheet.

Syntax **OFFSET**(*reference*, *rows*, *columns* [, *height*] [, *width*])

reference is a reference to a cell from which the offset reference is based. If you specify a range reference, #VALUE! is returned.

rows is the number of rows from reference that represents the upper-left cell of the offset range. A positive number represents rows below the starting cell; a negative number represents rows above the starting cell. If *rows*

places the upper-left cell of the offset range outside the spreadsheet boundary, #REF! is returned.

columns is the number of columns from reference that represents the upper-left cell of the offset range. A positive number represents columns right of the starting cell; a negative number represents columns left of the starting cell. If *columns* places the upper-left cell of the offset range outside the spreadsheet boundary, #REF! is returned.

height is a positive number representing the number of rows to include in the offset range. Omitting this argument assumes a single row .

width is a positive number representing the number of columns to include in the offset range. Omitting this argument assumes a single column.

Remarks **OFFSET** does not change the current selection in the worksheet. Because it returns a reference, **OFFSET** can be used in any function that requires or uses a cell or range reference as an argument.

See Also **COLUMN**, **INDIRECT**, and **ROW** functions

Examples `OFFSET(B1, 3, 2, 1, 1)` returns the contents of cell D4
`SUM(OFFSET(A1, 2, 4, 3, 2))` equals the sum of the range E3:F5

OR

Description Returns True if at least one of a series of logical arguments is true.

Syntax **OR**(*logical_list*)

logical_list is a list of conditions separated by commas. You can include as many as 30 conditions in the list. The list can contain logical values or a reference to a range containing logical values. Text and empty cells are ignored. If there are no logical values in the list, the error value #VALUE! is returned.

See Also **AND**, **IF** , and **NOT** functions

Example `OR(1 + 1 = 1, 5 + 5 = 10)` returns True because one of the arguments is true.

PI

Description Returns the value of pi (π), which is approximately 3.14159265358979 when calculated to 15 significant digits.

Syntax **PI**()

Remarks Although **PI** does not use arguments, you must supply the empty parentheses to correctly reference the function.

See Also **COS**, **SIN**, and **TAN** functions

PMT

Description Returns the periodic payment of an annuity, based on regular payments and a fixed periodic interest rate.

Syntax **PMT**(*interest*, *nper*, *pv* [, *fv*] [, *type*])

interest is the fixed periodic interest rate.

nper is the number of periods in the annuity.

pv is the present value, or the amount the annuity is currently worth.

fv is the future value, or the amount the annuity will be worth. When you omit this argument, a future value of 0 is assumed.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

Remarks **PMT** returns only the principal and interest payment, it does not include taxes or other fees.

The units used for interest must match those used for *nper*. For example, if the annuity has an 8% annual interest rate over a period of 5 years, specify 8%/12 for interest and 5*12 for *nper*.

Cash paid out, such as a payment, is shown as a negative number. Cash received, such as a dividend check, is shown as a positive number.

See Also **FV** , **IPMT**, **NPER**, **PPMT**, **PV** , and **RATE** functions

Examples `PMT(8%/12, 48, 18000)` returns -439.43
`PMT(8%/12, 48, 18000, 0, 1)` returns -436.52

PPMT

Description Returns the principle paid on an annuity for a given period.

Syntax **PPMT**(*interest*, *per*, *nper*, *pv*, [*fv*], [*type*])

interest is the fixed periodic interest rate.

per is the period for which to return the principle.

nper is the number of periods in the annuity.

pv is the present value, or the amount the annuity is currently worth.

fv is the future value, or the amount the annuity will be worth. When you omit this argument, a future value of 0 is assumed.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

Remarks The units used for interest must match those used for *nper*. For example, if the annuity has an 8% annual interest rate over a period of 5 years, specify 8%/12 for interest and 5*12 for *nper*.

See Also **FV** , **IPMT**, **NPER**, **PMT**, **PV** , and **RATE** functions

Examples `PPMT(8%/12, 2, 48, 18000)` returns -321.56
`PPMT(8%/12, 2, 48, 18000, 0, 1)` returns -319.43

PRODUCT

Description Multiplies a list of numbers and returns the result.

Syntax **PRODUCT**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas.

- The list can contain numbers, logical values, text representations of numbers, or a reference to a range containing those values.
- Error values or text that cannot be translated into numbers return errors.
- If a range reference is included in the list, text, logical expressions, and empty cells in the range are ignored.
- All numeric values, including 0, are used in the calculation.

See Also **FACT** and **SUM** functions

Example `PRODUCT(1, 2, 3, 4)` returns 24

PROPER

Description Returns the specified string in proper-case format.

Syntax **PROPER**(*text*)

text is any string.

Remarks In proper-case format, the first alphabetic character in a word is capitalized. If an alphabetic character follows a number, punctuation mark, or space, it is capitalized. All other alphabetic characters are lowercase. Numbers are not changed by **PROPER**.

See Also **LOWER** and **UPPER** functions

Examples **PROPER("3rd Quarter")** returns "3Rd Quarter"
PROPER("JOHN DOE") returns "John Doe"

PV

Description Returns the present value of an annuity, considering a series of constant payments made over a regular payment period.

Syntax **PV**(*interest*, *nper*, *pmt* [, *fv*] [, *type*])

interest is the fixed periodic interest rate.

nper is the number of payment periods in the investment.

pmt is the fixed payment made each period.

fv is the future value, or the amount the annuity will be worth. When you omit this argument, a future value of 0 is assumed.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

Remarks The units used for interest must match those used for *nper*. For example, if the annuity has an 8% annual interest rate over a period of 5 years, specify 8%/12 for interest and 5*12 for *nper*.

Cash paid out, such as a payment, is shown as a negative number. Cash received, such as a dividend check, is shown as a positive number.

See Also **FV** , **IPMT**, **NPER**, **PMT**, **PPMT**, and **RATE** functions

Examples `PV(8%/12, 48, 439.43)` returns -17999.89

`PV(8%/12, 48, -439.43)` returns 17999.89

RAND

Description Returns a number selected randomly from a uniform distribution greater than or equal to 0 and less than 1.

Syntax **RAND**()

Remarks Although **RAND** does not use arguments, you must supply the empty parentheses to correctly reference the function.

Example `RAND()*10` returns a random number greater than or equal to 0 and less than 10.

RATE

Description Returns the interest rate per period of an annuity, given a series of constant cash payments made over a regular payment period.

Syntax **RATE**(*nper*, *pmt*, *pv* [, *fv*] [, *type*] [, *guess*])

nper is the number of periods in the annuity.

pmt is the fixed payment made each period. Generally, *pmt* includes only principle and interest, not taxes or other fees.

pv is the present value of the annuity.

fv is the future value, or the amount the annuity will be worth. When you omit this argument, a future value of 0 is assumed.

type indicates when payments are due. Use 0 if payments are due at the end of the period or 1 if payments are due at the beginning of the period. When you omit this argument, 0 is assumed.

guess is your estimate of the interest rate. If no argument is supplied, a value of .1 (10%) is assumed.

Remarks **RATE** is calculated iteratively, cycling through the calculation until the result is accurate to .00001 percent. If the result cannot be found after 20 iterations, #NUM! is returned. When this occurs, supply a different value for *guess*.

See Also **FV** , **IPMT**, **NPER**, **PMT**, **PPMT**, and **PV** functions

Example `RATE(48, -439.43, 18000)` returns .0067 (rounded to 4 decimals), which is the monthly interest rate. The annual interest rate (.0067 multiplied by 12) is 8%.

REPLACE

Description Replaces part of a text string with another text string.

Syntax **REPLACE**(*orig_text*, *start_position*, *num_chars*, *repl_text*)

orig_text is the original text string.

start_position is the character position where the replacement begins.

- If *start_position* is greater than the number of characters in *orig_text*, *repl_text* is appended to the end of *orig_text*.

- If *start_position* is less than 1, #VALUE! is returned.

num_chars is the number of characters to replace. If this argument is negative, #VALUE! is returned.

repl_text is the replacement text string.

See Also **MID**, **SEARCH**, and **TRIM** functions

Examples `REPLACE("For the year: 1993", 18, 1, "4")` returns "For the year: 1994"

REPT

Description Repeats a text string the specified number of times.

Syntax **REPT**(*text*, *number*)

text is any text string.

number is the number of times you want text to repeat. If *number* is 0, empty text ("") is returned.

Remarks The result of **REPT** cannot exceed 255 characters.

Example `REPT("error-", 3)` returns "error-error-error-"

RIGHT

Description Returns the rightmost characters from the given text string.

Syntax **RIGHT**(*text* [, *num_chars*])

text is any text string.

num_chars is the number of characters to return. The value must be greater than or equal to zero. If *num_chars* is greater than the number of characters in text, the entire string is returned. Omitting this argument assumes a value of 1.

See Also **LEFT** and **MID** functions

Examples `RIGHT("2nd Quarter")` returns "r"
`RIGHT("2nd Quarter", 7)` returns "Quarter"

ROUND

Description Rounds the given number to the supplied number of decimal places.

Syntax **ROUND**(*number*, *precision*)

number is any value.

precision is the number of decimal places to which *number* is rounded.

- When a negative precision is used, the digits to the right of the decimal point are dropped and the absolute number of significant digits specified by precision are replaced with zeros.
- If *precision* is 0, number is rounded to the nearest integer.

See Also **CEILING**, **FLOOR**, **INT**, **MOD**, and **TRUNC** functions

Examples `ROUND(123.456, 2)` returns 123.46
`ROUND(9899.435, -2)` returns 9900

ROW

Description Returns the row number of the supplied reference.

Syntax **ROW**(*reference*)

reference is a cell or range reference. Omitting this argument returns the row number of the cell in which **ROW** is entered.

See Also **COLUMN** and **ROWS** function

Examples `ROW(B3)` returns 3

ROWS

Description Returns the number of rows in a range reference.

Syntax **ROWS**(*range*)

range is a reference to a range of cells.

See Also **COLUMNS** and **ROW** functions

Examples `ROWS(A1:D5)` returns 5
`ROWS(C30:F35)` returns 6

SEARCH

Description Locates the position of the first character of a specified text string within another text string.

Syntax **SEARCH**(*search_text*, *text* [, *start_position*])

search_text is the text to find.

- The search string can contain wildcard characters. The available wildcard characters are * (asterisk), which matches any sequence of characters, and ? (question mark), which matches any single character.
- To search for an asterisk or question mark, include a tilde (~) before the character.

text is the text to be searched.

start_position is the character position where the search begins. If the number you specify is less than 0 or greater than the number of characters in text, #VALUE! is returned. Omitting this argument assumes a starting position of 1.

Remarks Text is searched from left to right, starting at the position specified. The search is not case-sensitive. If text does not contain the search string, #VALUE! is returned.

See Also **FIND**, **MID**, **REPLACE**, and **SUBSTITUTE** functions

Examples `SEARCH("?5", "Bin b45")` returns 6
`SEARCH("b", "Bin b45", 4)` returns 5

SECOND

Description Returns the second that corresponds to the supplied date.

Syntax **SECOND**(*serial_number*)

serial_number is the time as a serial number. The decimal portion of the number represents time as a fraction of the day.

See Also **DAY**, **HOUR**, **MINUTE**, **MONTH**, **NOW**, **WEEKDAY**, and **YEAR** functions

Examples
`SECOND(.259)` returns 58
`SECOND(34657.904)` returns 46

SIGN

Description Determines the sign of the specified number.

Syntax **SIGN**(*number*)

number is any number.

Remarks **SIGN** returns 1 if the specified number is positive, -1 if it is negative, and 0 if it is 0.

See Also **ABS** function

Examples
`SIGN(-123)` returns -1
`SIGN(123)` returns 1

SIN

Description Returns the sine of the supplied angle.

Syntax **SIN**(*number*)

number is the angle in radians. If the angle is in degrees, convert the angle to radians by multiplying the angle by PI()/180.

See Also **ASIN** and **PI** functions

Examples
`SIN(45)` returns .85
`SIN(90)` returns .89

SINH

Description Returns the hyperbolic sine of the specified number.

Syntax **SINH**(*number*)

number is any number.

See Also **ASINH** and **PI** functions

Examples
`SINH(1)` returns 1.18
`SINH(3)` returns 10.02

SLN

Description Returns the depreciation of an asset for a specific period of time using the straight-line balance method.

Syntax **SLN**(*cost*, *salvage*, *life*)

cost is the initial cost of the asset.

salvage is the salvage value of the asset.

life is the number of periods of the useful life of the asset.

See Also **DDB**, **SYD**, and **VDB** functions

Example `SLN(10000, 1000, 7)` returns 1285.71

SQRT

Description Returns the square root of the specified number.

Syntax **SQRT**(*number*)

number is any positive number. If you specify a negative number, #NUM! is returned.

See Also **SUMSQ** function

Examples `SQRT(9)` returns 3
`SQRT(2.5)` returns 1.58

STDEV

Description Returns the standard deviation of a population based on a sample of supplied values. The standard deviation of a population represents an average of deviations from the population mean within a list of values.

Syntax **STDEV**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas. The list can contain numbers or a reference to a range that contains numbers.

See Also **STDEVP**, **VAR**, and **VARP** functions

Example `STDEV(4.0, 3.0, 3.0, 3.5, 2.5, 4.0, 3.5)` returns .56

STDEVP

Description Returns the standard deviation of a population based on an entire population of values. The standard deviation of a population represents an average of deviations from the population mean within a list of values.

Syntax **STDEVP**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas. The list can contain numbers or a reference to a range that contains numbers.

See Also **STDEV**, **VAR**, and **VARP** functions

Example `STDEVP(4.0, 3.0, 3.0, 3.5, 2.5, 4.0, 3.5)` returns .52

SUBSTITUTE

Description Replaces a specified part of a text string with another text string.

Syntax **SUBSTITUTE**(*text*, *old_text*, *new_text* [, *instance*])

text is a text string that contains the text to replace. You can also specify a reference to a cell that contains text.

old_text is the text string to be replaced.

new_text is the replacement text.

instance specifies the occurrence of *old_text* to replace. If this argument is omitted, every instance of *old_text* is replaced.

See Also **REPLACE** and **TRIM** functions

Examples `SUBSTITUTE("First Quarter Results", "First", "Second")` returns "Second Quarter Results"

`SUBSTITUTE("Shipment 45, Bin 45", "45", "52", 2)` returns "Shipment 45, Bin 52"

SUM

Description Returns the sum of the supplied numbers.

Syntax **SUM**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas.

- The list can contain numbers, logical values, text representations of numbers, or a reference to a range containing those values.
- Error values or text that cannot be translated into numbers return errors.
- If a range reference is included in the list, text, logical expressions, and empty cells in the range are ignored.

See Also **AVERAGE**, **COUNT**, **COUNTA**, **PRODUCT**, and **SUMSQ** functions

Examples `SUM(1000, 2000, 3000)` returns 6000

`SUM(A10:D10)` returns 4000 when each cell in the range contains 1000

SUMSQ

Description Squares each of the supplied numbers and returns the sum of the squares.

Syntax **SUMSQ**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas.

- The list can contain numbers, logical values, text representations of numbers, or a reference to a range containing those values.
- Error values or text that cannot be translated into numbers return errors.
- If a range reference is included in the list, text, logical expressions, and empty cells in the range are ignored.

See Also **SUM** function

Example `SUMSQ(9, 10, 11)` returns 302

SYD

Description Returns the depreciation of an asset for a specified period using the sum-of-years method. This depreciation method uses an accelerated rate, where the greatest depreciation occurs early in the useful life of the asset.

Syntax **SYD**(*cost*, *salvage*, *life*, *per*)

cost is the initial cost of the asset.

salvage is the salvage value of the asset.

life is the number of periods in the useful life of the asset.

period is the period for which to calculate the depreciation. The time units used to determine period and life must match.

See Also **DDB**, **SLN**, and **VDB** functions

Example `SYD(10000, 1000, 7, 3)` returns 1607.14

T

Description Tests the supplied value and returns the value if it is text.

Syntax **T**(*value*)

value is the value to test.

Remarks Empty text ("") is returned for any value that is not text.

See Also **N** , and **VALUE** functions

Examples `T("Report")` returns "Report"

`T(A4)` returns empty text ("") if A4 contains a number

TAN

Description Returns the tangent of the specified angle.

Syntax **TAN**(*number*)

number is the angle in radians. To convert a number expressed as degrees to radians, multiply the degrees by 180/PI().

See Also **ATAN**, **ATAN2**, **PI** , and **TANH** functions

Examples `TAN(45)` returns 1.62
`TAN(90)` returns -2.00

TANH

Description Returns the hyperbolic tangent of a number.

Syntax **TANH**(*number*)

number is any number.

See Also **ATANH**, **COSH**, **SINH**, and **TAN** functions

Examples `TANH(-2)` returns -.96
`TANH(1.2)` returns .83

TEXT

Description Returns the given number as text, using the specified formatting.

Syntax **TEXT**(*number*, *format*)

number is any value, a formula that evaluates to a number, or a reference to a cell that contains a value.

format is a string representing a number format. The string can be any valid format string including "General," "M/DD/YY," or "H:MM AM/PM." The format must be surrounded by a set of double quotation marks. Asterisks cannot be included in format.

See Also **DOLLAR**, **FIXED**, **T** , and **VALUE** functions

Examples `TEXT(123.62, "0.000")` returns 123.620
`TEXT(34626.2, "MM/DD/YY")` returns 10/19/94

TIME

Description Returns a serial number for the supplied time.

Syntax **TIME**(*hour*, *minute*, *second*)

hour is a number from 0 to 23.

minute is a number from 0 to 59.

second is a number from 0 to 59.

See Also **HOUR**, **MINUTE**, **NOW**, **SECOND**, and **TIMEVALUE** functions

Examples `TIME(12, 26, 24)` returns .52
`TIME(1, 43, 34)` returns .07

TIMEVALUE

Description Returns a serial number for the supplied text representation of time.

Syntax **TIMEVALUE**(*text*)

text is a time in text format.

See Also **HOUR**, **MINUTE**, **NOW**, **SECOND**, and **TIME** functions

Examples `TIMEVALUE("1:43:43 am")` returns .07
`TIMEVALUE("14:10:07")` returns .59

TODAY

Description Returns the current date as a serial number.

Syntax **TODAY**()

Remarks This function is updated only when the worksheet is recalculated.

See Also **DATE**, **DAY**, and **NOW** functions

TRIM

Description Removes all spaces from text except single spaces between words.

Syntax **TRIM**(*text*)

text is any text string or a reference to a cell that contains a text string.

Remarks Text that is imported from another environment may require this function.

See Also **CLEAN**, **MID**, **REPLACE**, and **SUBSTITUTE** functions

Example `TRIM(" Level 3, Gate 45 ")` returns "Level 3, Gate 45"

TRUE

Description Returns the logical value True. This function always requires the trailing parentheses.

Syntax **TRUE**()

See Also **FALSE** function

TRUNC

Description Truncates the given number to an integer.

Syntax **TRUNC**(*number* [, *precision*])

number is any value.

precision is the number of decimal places allowed in the truncated number. Omitting this argument assumes a *precision* of 0.

Remarks **TRUNC** removes the fractional part of a number to the specified precision without rounding the number.

See Also **CEILING**, **FLOOR**, **INT**, **MOD**, and **ROUND** functions

Examples `TRUNC(123.456, 2)` returns 123.45
`TRUNC(9899.435, -2)` returns 9800

TYPE

Description Returns the argument type of the given expression.

Syntax **TYPE**(*expression*)

expression is any expression.

Remarks The following table lists the *expression* types and numbers.

Expression type	Number
Number	1
Text string	2
Logical value	4
Error value	16

See Also **ISBLANK**, **ISERR**, **ISERROR**, **ISLOGICAL**, **ISNA**, **ISNONTEXT**, **ISNUMBER**, **ISREF**, and **ISTEXT** functions

Examples `TYPE(A1)` returns 1 if cell A1 contains a number.
`TYPE("Customer")` returns 2

UPPER

Description Changes the characters in the specified string to uppercase characters.

Syntax **UPPER**(*text*)

text is any string.

Remarks Numeric characters in the string are not changed.

See Also **LOWER** and **PROPER** functions

Examples `UPPER("3rd Quarter")` returns "3RD QUARTER"
`UPPER("JOHN DOE")` returns "JOHN DOE"

VALUE

Description Returns the specified text as a number.

Syntax **VALUE**(*text*)

text is any text string, a formula that evaluates to a text string, or a cell reference that contains a text string. You can also specify a date or time in a recognizable format (e.g., M/DD/YY for dates or H:MM AM/PM for time). If the format is not recognized, #VALUE! is returned.

See Also **DOLLAR**, **FIXED**, and **TEXT** functions

Examples `VALUE(9800)` returns 9800
`VALUE("123")` returns 123

VAR

Description Returns the variance of a population based on a sample of values.

Syntax **VAR**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas. The list can contain numbers or a reference to a range that contains numbers.

See Also **STDEV**, **STDEVP**, and **VARP** functions

Example `VAR(4.0, 3.0, 3.0, 3.5, 2.5, 4.0, 3.5)` returns .31

VARP

Description Returns the variance of a population based on an entire population of values.

Syntax **VARP**(*number_list*)

number_list is a list of as many as 30 numbers, separated by commas. The list can contain numbers or a reference to a range that contains numbers.

See Also **STDEV**, **STDEVP**, and **VAR** functions

Example VARP(4.0, 3.0, 3.0, 3.5, 2.5, 4.0, 3.5) returns .27

VDB

Description Returns the depreciation of an asset for a specified period using a variable method of depreciation.

Syntax **VDB**(*cost*, *salvage*, *life*, *start_period*, *end_period* [, *factor*] [, *method*])

cost is the initial cost of the asset.

salvage is the salvage value of the asset.

life is the number of periods in the useful life of the asset.

start_period is the beginning period for which to calculate the depreciation. The time units used to determine *start_period* and *life* must match.

end_period is the ending period for which to calculate the depreciation. The time units used to determine *end_period* and *life* must match.

factor is the rate at which the balance declines. Omitting this argument assumes a default of 2, which is the double-declining balance factor.

method is a logical value that determines if you want to switch to straight-line depreciation when depreciation is greater than the declining balance calculation. Use True to maintain declining balance calculation; use False or omit the argument to switch to straight-line depreciation calculation.

See Also **DDB**, **SLN**, and **SYD** functions

Examples VDB(10000, 1000, 7, 3, 4) returns 1041.23

VLOOKUP

Description Searches the first column of a table for a value and returns the contents of a cell in that table that corresponds to the location of the search value.

Syntax **VLOOKUP**(*search_item*, *search_range*, *column_index*)

search_item is a value, text string, or reference to a cell containing a value that is matched against data in the top row of *search_range*.

search_range is the reference of the range (table) to be searched. The cells in the first column of *search_range* can contain numbers, text, or logical values. The contents of the first column must be in ascending order (e.g., -2, -1, 0, 2...A through Z, False, True). Text searches are not case-sensitive.

column_index is the column in the search range from which the matching value is returned.

- *column_index* can be a number from 1 to the number of rows in the search range.
- If *column_index* is less than 1, #VALUE! is returned.
- When *column_index* is greater than the number of rows in the table, #REF! is returned.

Remarks **VLOOKUP** compares the information in the first column of *search_range* to the supplied *search_item*. When a match is found, information located in the same row and supplied column (*column _index*) is returned.

If *search_item* cannot be found in the first column of *search_range*, the largest value that is less than *search_item* is used. When *search_item* is less than the smallest value in the first column of the *search_range*, #REF! is returned.

See Also **HLOOKUP**, **INDEX**, **LOOKUP**, and **MATCH** functions

	A	B	C	D	E
1	Employee	Start Date	Emp. No.	Salary	Exempt
2	Anderson	10/15/84	2348	$37,800	Y
3	Clark	2/6/90	4891	$28,700	N
4	Davis	6/21/80	2480	$46,950	Y
5	Franklin	4/20/88	3793	$30,275	Y
6	Lee	8/30/89	3961	$25,000	N
7	Olson	11/1/81	2578	$45,780	Y
8	Turner	2/15/93	5129	$26,100	N
9	Wilson	9/1/89	3965	$31,650	Y

Examples In the preceding worksheet:

`VLOOKUP("Clark", A2:E9, 4)` returns $28,700
`VLOOKUP("Lee", A2:E9, 3)` returns 3961

WEEKDAY

Description Returns the day of the week that corresponds to the supplied date.

Syntax **WEEKDAY**(*serial_number*)

serial_number is the date as a serial number or as text (e.g., “06-21-94” or “21-Jun-94”).

Remarks **WEEKDAY** returns a number ranging from 1 (Sunday) to 7 (Saturday).

See Also **DAY**, **NOW**, **TEXT**, and **TODAY** functions

Examples `WEEKDAY(34399.92)` returns 1, indicating Sunday
`WEEKDAY(“06/21/94”)` returns 3, indicating Tuesday

YEAR

Description Returns the year that corresponds to the supplied date.

Syntax **YEAR**(*serial_number*)

serial_number is the date as a serial number or as text (e.g., “06-21-94” or “21-Jun-94”).

See Also **DAY**, **HOUR**, **MINUTE**, **MONTH**, **NOW**, **SECOND**, **TODAY**, and **WEEKDAY** functions

Examples `YEAR(34328)` returns 1993
`YEAR(“06/21/94”)` returns 1994