Pack 3 of 3

Financial & Advanced Functions

Professional-grade formulas for finance & accounting

10 Cheatsheets **35+** Functions

What's Inside

1. LAMBDA Function Guide
2. LET Function Guide
3. Dynamic Array Functions
4. Financial Functions
5. Statistical Functions
6. Rounding & Precision
7. Variance & Budget Analysis
8. Cash Flow & Investment
9. Depreciation Functions
10. Loan & Amortization

Pack 3 of 3

Contents

Financial & Advanced Functions • 10 Cheatsheets

01 LAMBDA Function Guide

02 LET Function Guide

03 Dynamic Array Functions

04 Financial Functions

05 Statistical Functions

06 Rounding & Precision

07 Variance & Budget Analysis

08 Cash Flow & Investment

09 Depreciation Functions

10 Loan & Amortization

Advanced Formulas
LAMBDA • LET • Dynamic Arrays

Financial Analysis
NPV • IRR • Depreciation • Loans

LAMBDA Function Guide

Create your own custom functions

Syntax

```
=LAMBDA(parameter1, [parameter2, ...], calculation)
```

Arguments

`parameter1` First input parameter name

`[parameter2]` Additional parameters (up to 253)

`calculation` The formula to execute using parameters

Examples

Simple Calculation

→ 10

```
=LAMBDA(x, x*2)(5)
```

Doubles the input value

Tax Calculator

→ 120

```
=LAMBDA(price, rate, price*(1+rate))(100, 0.2)
```

Calculates price with tax

Creating Named Functions

1. Go to Formulas → Name Manager
2. Click "New" and enter a name
3. In "Refers to", enter your LAMBDA
4. Use your function anywhere!

Name: ADDTAX

```
=LAMBDA(price, rate, price*(1+rate))
```

```
Usage: =ADDTAX(100, 0.2)
```

Why Use LAMBDA?

Reusability

Define once, use everywhere

No VBA

Custom functions without macros

Advanced Usage

Recursive Factorial

```
=LAMBDA(n, IF(n<=1, 1, n*FACTORIAL(n-1)))
```

Self-referencing LAMBDA

Array Mapping

```
=MAP(A1:A5, LAMBDA(x, x^2))
```

Apply LAMBDA to each cell

LET Function Guide

Variables inside formulas

Syntax

```
=LET(name1, value1, [name2, value2, ...], calculation)
```

Arguments

`name1`	First variable name
`value1`	Value or calculation for name1
`[name2, value2]`	Additional name-value pairs
`calculation`	Final calculation using the variables

Examples

Simple Variable

```
=LET(x, A1*2, x+10)
```

Store A1*2 as "x", then use it

Multiple Variables

```
=LET(sales, B1, cost, C1, profit, sales-cost, profit/sales)
```

Calculate profit margin with named variables

Avoid Repetition

❌ Without LET (repeats VLOOKUP):

```
=IF(VLOOKUP(A1,B:C,2,0)>100, VLOOKUP(A1,B:C,2,0)*0.9, VLOOKUP(A1,B:C,2,0))
```

✓ With LET (lookup once):

```
=LET(val, VLOOKUP(A1,B:C,2,0), IF(val>100, val*0.9, val))
```

Benefits

Performance

Calculate once, reuse result

Readability

Self-documenting variable names

Pro Tips

- Use descriptive variable names (revenue vs r)
- Combine with LAMBDA for reusable functions

Dynamic Array Functions

FILTER, SORT, UNIQUE & more (Excel 365)

Core Array Functions

FILTER	Return rows matching criteria	`=FILTER(A:C, B:B="Yes")`
SORT	Sort array by column	`=SORT(A:C, 2, -1)`
SORTBY	Sort by external range	`=SORTBY(Names, Scores, -1)`
UNIQUE	Remove duplicates	`=UNIQUE(A2:A100)`
SEQUENCE	Generate number series	`=SEQUENCE(10, 1, 1, 1)`
RANDARRAY	Generate random numbers	`=RANDARRAY(5, 3, 1, 100)`

Power Combinations

Filter + Sort

```
=SORT(FILTER(Data, Region="North"), 3, -1)
```

Filter by region, sort by column 3 descending

Unique + Sort

```
=SORT(UNIQUE(A2:A100))
```

Get unique values and sort them

Unique + Filter

```
=UNIQUE(FILTER(Products, Sales>1000))
```

Unique products with sales over 1000

Sequence for Dates

```
=SEQUENCE(12, 1, DATE(2024,1,1), 30)
```

Generate 12 monthly dates

Understanding Spill

Spill Range

Results automatically expand into adjacent cells

Spill Reference

Use A1# to reference entire spill range

Array Helper Functions

TAKE

First/last n rows or columns

DROP

Remove first/last n rows

VSTACK

Stack arrays vertically

HSTACK

Stack arrays horizontally

Financial Functions

Loans, valuations & time value of money

Loan & Payment Functions

`PMT`	Calculate loan payment	`=PMT(5%/12, 360, -250000)`
`IPMT`	Interest portion of payment	`=IPMT(5%/12, 1, 360, -250000)`
`PPMT`	Principal portion of payment	`=PPMT(5%/12, 1, 360, -250000)`
`RATE`	Calculate interest rate	`=RATE(360, -1342, 250000)*12`
`NPER`	Number of periods	`=NPER(5%/12, -1500, 250000)`

Valuation Functions

`NPV`	Net present value (regular)	*Assumes end-of-period cash flows*
`XNPV`	NPV with specific dates	*Use for irregular cash flows*
`IRR`	Internal rate of return	*Assumes regular periods*
`XIRR`	IRR with specific dates	*Use for irregular cash flows*
`PV`	Present value	*What future cash is worth today*
`FV`	Future value	*Value of investment over time*

Real-World Examples

Monthly Loan Payment

```
=PMT(6%/12, 60, -25000)
```

$483.32/month for 5yr car loan at 6%

Investment NPV

```
=XNPV(10%, B2:B10, A2:A10)
```

NPV of cash flows with actual dates

Savings Future Value

```
=FV(7%/12, 240, -500, -10000)
```

Value after 20yrs of $500/mo + $10k initial

Key Tips

- Use negative for cash outflows (payments you make)
- Divide annual rate by 12 for monthly calculations
- Multiply by 12 periods for annual rate
- XNPV/XIRR are more accurate for real-world scenarios

Statistical Functions

Analysis, trends & data distribution

Central Tendency

- `AVERAGE` Mean of values
- `MEDIAN` Middle value
- `MODE.SNGL` Most frequent value
- `MODE.MULT` All modes (array)

Dispersion & Variability

`STDEV.S`
Sample standard deviation
Most common

`STDEV.P`
Population std dev
Full dataset

`VAR.S`
Sample variance
Squared std dev

`VAR.P`
Population variance
Full dataset

Ranking & Position

- `MIN` Smallest value
- `MAX` Largest value
- `LARGE` Nth largest
- `SMALL` Nth smallest
- `RANK` Rank in dataset
- `PERCENTILE` 90th percentile
- `PERCENTRANK` Percentile rank
- `QUARTILE` Quartile value (0–4)

Counting Functions

- `COUNT` Count numbers
- `COUNTA` Count non-empty
- `COUNTBLANK` Count empty cells
- `COUNTIF` Count with criteria

Advanced Analysis

`CORREL`
Correlation coefficient

`COVARIANCE.S`
Sample covariance

`FORECAST.LINEAR`
Linear prediction

`TREND`
Linear trend values

`GROWTH`
Exponential trend

`LINEST`
Regression statistics

Rounding & Precision

Essential functions for accurate financial calculations

Rounding Functions

`ROUND`	Standard rounding	`=ROUND(3.456, 2) → 3.46`
`ROUNDUP`	Always rounds up	`=ROUNDUP(3.421, 2) → 3.43`
`ROUNDDOWN`	Always rounds down	`=ROUNDDOWN(3.789, 2) → 3.78`
`MROUND`	Round to multiple	`=MROUND(17, 5) → 15`
`CEILING`	Round up to multiple	`=CEILING(23, 10) → 30`
`FLOOR`	Round down to multiple	`=FLOOR(27, 10) → 20`

Truncation & Division

`TRUNC`	Truncate to decimals (no rounding)	`=TRUNC(3.789, 2) → 3.78`
`INT`	Round down to integer	`=INT(3.9) → 3`
`MOD`	Remainder after division	`=MOD(17, 5) → 2`
`QUOTIENT`	Integer portion of division	`=QUOTIENT(17, 5) → 3`

Accounting Use Cases

Round to Nearest Cent

```
=ROUND(A1*B1, 2)
```

Price × Quantity to 2 decimals

Round to Thousands

```
=ROUND(Revenue, -3)
```

Use negative decimals for left-of-decimal rounding

Currency Rounding (0.05)

```
=MROUND(Total, 0.05)
```

Round to nearest 5 cents (Swiss Franc, etc.)

Allocation Without Pennies

```
=FLOOR(Amount/3, 0.01)
```

Split amounts evenly, handle remainder separately

Avoiding Penny Errors

- ✗ Rounding errors compound
- ✗ Allocations don't sum correctly
- ✗ Tax calculations differ
- ✗ Floating point precision
- ✓ Round final results, not intermediate steps
- ✓ Use FLOOR then add remainder to last item
- ✓ Match jurisdiction's rounding rules exactly
- ✓ Use ROUND on comparisons: ROUND(A1−B1,2)=0

Variance & Budget Analysis

Essential formulas for financial reporting

Core Variance Formulas

Variance ($)	`=Actual - Budget`	Absolute difference
Variance (%)	`=(Actual-Budget)/ABS(Budget)`	Percentage change
Favorable/Unfavorable	`=IF(Actual>Budget, "Fav", "Unfav")`	For revenue items
Expense Variance	`=IF(Actual<Budget, "Fav", "Unfav")`	Lower spend = favorable

YTD & Forecasting

YTD Actual

```
=SUMIFS(Actuals, Dates, ">="&DATE(YEAR(TODAY()),1,1), Dates, "<="&TODAY())
```

Sum from Jan 1 to today

YTD Budget

```
=SUMIFS(Budget, Month, "<="&MONTH(TODAY()))
```

Budget through current month

Full Year Forecast

```
=YTD_Actual + Remaining_Budget
```

Actual + future budget periods

Run Rate Projection

```
=(YTD_Actual/MONTH(TODAY()))*12
```

Annualize current performance

Key Metrics

Budget Attainment	`=Actual/Budget`	>100% = over budget (expense) or beat target (revenue)
Spend Rate	`=YTD_Spend/Annual_Budget`	Should roughly equal elapsed time %
Forecast Accuracy	`=1-ABS(Actual-Forecast)/Actual`	Closer to 100% = better forecast
Cost per Unit	`=Total_Cost/Units`	Track efficiency over time

Pro Tips

- Use ABS() in denominators to handle negative budgets correctly
- Use IFERROR to handle divide-by-zero gracefully
- Create named ranges for Budget and Actual columns
- Consider seasonality in YTD comparisons

Cash Flow & Investment Analysis

NPV, IRR, and capital budgeting essentials

Net Present Value Functions

`NPV`	Net present value of periodic cash flows	`=NPV(10%, CF1:CF5)+Initial`
`XNPV`	NPV for irregular cash flow dates	`=XNPV(10%, B2:B10, A2:A10)`

Internal Rate of Return

`IRR`	Internal rate of return (periodic)	`=IRR(B2:B10)` → 12.5%
`XIRR`	IRR for irregular cash flow dates	`=XIRR(B2:B10, A2:A10)`
`MIRR`	Modified IRR with reinvestment rate	`=MIRR(CF, 8%, 10%)`

Practical Applications

Project Comparison

```
=XNPV(WACC, ProjectA_CF, Dates) - XNPV(WACC, ProjectB_CF, Dates)
```

Compare two investment options using NPV difference

Breakeven Analysis

```
=MATCH(TRUE, SCAN(0, CashFlows, LAMBDA(a,b,a+b))>0, 0)
```

Find the period when cumulative cash flow turns positive

Payback Period

```
=-InitialInvest/AVERAGE(AnnualCashFlows)
```

Simple payback (use discounted CF for more accuracy)

Present Value Factor

```
=1/(1+rate)^period
```

Manual PV factor for cash flow discounting

Investment Decision Rules

`NPV > 0`	**Accept**	Project adds value
`IRR > WACC`	**Accept**	Returns exceed cost of capital
$NPV_1 > NPV_2$	**Choose Project 1**	Higher absolute value creation
`Payback < Target`	**Accept**	Meets liquidity requirements

Watch Out For

- ✗ NPV excludes initial investment
- ✗ IRR assumes reinvestment at IRR
- ✗ Multiple IRRs for non-conventional CF
- ✗ Comparing projects of different sizes
- ✓ Add initial (negative) CF separately: =NPV(...)+B1
- ✓ Use MIRR for realistic reinvestment rate
- ✓ Rely on NPV or specify IRR guess
- ✓ Use profitability index: NPV/Initial Investment

Depreciation Functions

Fixed asset depreciation calculations for accounting

Depreciation Functions

SLN	Straight-line depreciation (equal each year)	=SLN(10000, 1000, 5) → 1,800
DB	Declining balance (fixed rate)	=DB(10000, 1000, 5, 1) → 3,690
DDB	Double declining balance	=DDB(10000, 1000, 5, 1) → 4,000
VDB	Variable declining balance	=VDB(10000, 1000, 5, 0, 1)
SYD	Sum-of-years digits	=SYD(10000, 1000, 5, 1) → 3,000

Method Comparison

Straight-Line	Equal amounts	Assets with even usage over time
Declining Balance	Decreasing amounts	Assets losing value quickly
Sum-of-Years	Accelerated decline	Technology, vehicles
MACRS (VDB)	IRS-specified rates	US tax compliance

Practical Applications

Fixed Asset Schedule

```
=SLN(PurchasePrice, SalvageValue, UsefulLife)
```

Calculate annual depreciation for each asset

Accumulated Depreciation

```
=SLN(Cost, Salvage, Life) * MIN(Period, Life)
```

Total depreciation through current period

Book Value

```
=Cost - VDB(Cost, Salvage, Life, 0, Period)
```

Net asset value at any point in time

Mid-Year Convention

```
=DB(Cost, Salvage, Life, Period, 6)
```

First year partial depreciation (6 months)

Tax vs. Book Depreciation

Method	Tax: MACRS (accelerated)	Book: Straight-line (usually)
Useful Life	Tax: IRS class lives	Book: Economic useful life
Salvage Value	Tax: Ignored (depreciate to 0)	Book: Estimated residual
Result	Tax: Deferred tax liability	Book: Matching principle

Pro Tips

- ✓ Use VDB with no_switch=TRUE to prevent automatic switch to straight-line
- ✓ Track book vs. tax depreciation for deferred tax calculations

Loan & Amortization

Payment calculations and debt schedule functions

Payment Functions

`PMT`	Calculate periodic payment amount	`=PMT(5%/12, 60, -25000) → $471.78`
`IPMT`	Interest portion of specific payment	`=IPMT(5%/12, 1, 60, -25000) → $104.17`
`PPMT`	Principal portion of specific payment	`=PPMT(5%/12, 1, 60, -25000) → $367.61`
`RATE`	Calculate interest rate	`=RATE(60, -471.78, 25000)*12 → 5%`
`NPER`	Number of periods	`=NPER(5%/12, -500, 25000) → 55.5`

Cumulative Functions

`CUMIPMT`	Cumulative interest over period range	`=CUMIPMT(5%/12, 60, 25000, 1, 12, 0) → $1,178`
`CUMPRINC`	Cumulative principal over period range	`=CUMPRINC(5%/12, 60, 25000, 1, 12, 0) → $4,483`

Building an Amortization Schedule

1	**Payment Number**	`=ROW()-1 (or sequence)`
2	**Payment Amount**	`=PMT($Rate, $Nper, $PV)`
3	**Interest Portion**	`=IPMT($Rate, Period, $Nper, $PV)`
4	**Principal Portion**	`=PPMT($Rate, Period, $Nper, $PV)`
5	**Remaining Balance**	`=PrevBalance + Principal`

Common Calculations

Mortgage Payment

```
=PMT(4.5%/12, 30*12, -350000)
```

30-year mortgage at 4.5% APR

Total Interest Cost

```
=PMT(Rate, Nper, PV)*Nper + PV
```

Lifetime interest paid on a loan

Payment Frequency Conversions

Monthly	Annual Rate / 12	Years × 12
Bi-Weekly	Annual Rate / 26	Years × 26
Weekly	Annual Rate / 52	Years × 52
Quarterly	Annual Rate / 4	Years × 4

Pro Tips

✓ Use negative PV for loans (cash received), positive for investments

✓ Type=0 for end-of-period, Type=1 for beginning payments

Accounting Workflows. Automated.

You handle the strategy. Let AI handle the repetitive work.

Built for Accountants

- Speed up your daily workflows
- Reduce manual, repetitive tasks
- AI-powered process automation
- More time for high-value work

Try Accounts Draft Free

www.accountsdraft.com

AI software that automates accounting workflows