

One-Day Advanced Excel Workshop: Formulas, Macros, and Workbook Security

Duration: 8 hours (9:00 to 5:00, including breaks) **Audience:** Advanced users who want to master complex formulas, automate work with VBA, and secure workbooks they share
Format: Instructor-led, about 20% concepts and 80% hands-on labs, with one running case study (a sales and commissions workbook) built up across the day **Prerequisites:** Excel 365 or 2021 on Windows (macro and security labs use the Developer tab). Comfort with nested IF, lookups, and PivotTables.

Learning Objectives

By the end of the day, participants will be able to:

- Write, debug, and optimize complex formulas using array logic, LET, LAMBDA, and text and date engineering
- Record, read, write, and debug VBA macros that automate real reporting tasks
- Build user forms and custom functions (UDFs) for repeatable tools
- Password protect cells, sheets, workbook structure, files, and VBA code, and understand what each layer does and does not protect
- Combine formulas, macros, and protection into a secure, shareable workbook

Schedule

Time	Module	Duration
9:00 – 9:15	Welcome and setup check	15 min
9:15 – 10:30	Module 1: Advanced formula engineering	75 min
10:30 – 10:45	Break	15 min
10:45 – 12:00	Module 2: Array, lookup, and dynamic formulas	75 min
12:00 – 1:00	Lunch	60 min
1:00 – 2:30	Module 3: Macros and VBA fundamentals	90 min

Time	Module	Duration
2:30 – 3:15	Module 4: Advanced VBA: forms, UDFs, and error handling	45 min
3:15 – 3:30	Break	15 min
3:30 – 4:30	Module 5: Password protection: cells, sheets, workbooks, and code	60 min
4:30 – 5:00	Capstone and wrap-up	30 min

Module Details

Welcome and Setup Check (15 min)

- Confirm Excel version, Developer tab, and Trust Center macro settings on every machine
- Overview of the running case study workbook
- Ground rules: save often, work on copies, and write passwords down in a safe place (lost passwords are very hard to recover)

Module 1: Advanced Formula Engineering (75 min)

- Reading and debugging formulas: Evaluate Formula, F9 on selected pieces, Trace Precedents and Dependents, Watch Window
- Reference mastery: mixed references, INDIRECT and OFFSET (and why to avoid them), INDEX as a reference, structured references
- LET to name values and remove repeated calculations
- LAMBDA and the Name Manager for reusable custom functions without VBA
- Logical engineering: IFS, SWITCH, XOR, nested logic vs lookup tables, boolean arithmetic (multiplying conditions)
- Text engineering: TEXTSPLIT, TEXTBEFORE, TEXTAFTER, TEXTJOIN, SUBSTITUTE chains, and extracting numbers from mixed strings
- Date and time engineering: WORKDAY.INTL, NETWORKDAYS.INTL, EDATE, EOMONTH, fiscal calendars, and ages and tenures with DATEDIF
- Error trapping patterns: IFERROR vs IFNA, validating inputs before calculating

- **Lab:** Build a commission engine with tiered rates, fiscal-quarter logic, and clean error messages, using LET to keep each formula readable

Module 2: Array, Lookup, and Dynamic Formulas (75 min)

- Dynamic array behavior: spill ranges, the # operator, and avoiding #SPILL! errors
- FILTER, SORTBY, UNIQUE, SEQUENCE, and combining them
- SUMPRODUCT as a multi-condition engine; AGGREGATE for ignoring errors and hidden rows
- XLOOKUP and XMATCH advanced use: wildcards, last match, approximate match, two-way lookups, and returning whole rows
- Stacking and reshaping: VSTACK, HSTACK, TAKE, DROP, CHOOSECOLS, TOCOL
- Iterating without VBA: MAP, BYROW, BYCOL, REDUCE, and SCAN
- Performance: volatile functions, full-column references, calculation modes, and when to move heavy logic to VBA
- **Lab:** Create a self-updating leaderboard and exceptions report that spills from raw data, then convert one repeated calculation into a named LAMBDA

Module 3: Macros and VBA Fundamentals (90 min)

- Enabling the Developer tab, Trust Center settings, and saving as .xlsm (and .xlsb for large files)
- Recording a macro: absolute vs relative recording, then reading the code in the VBA Editor (VBE)
- VBE tour: Project Explorer, Properties window, Immediate window, Locals window
- Core language: variables and data types, Option Explicit, constants, Sub vs Function, and comments
- The Excel object model: Application, Workbook, Worksheet, and Range; referencing cells with Cells, Range, and named ranges
- Control flow: If/Elseif, Select Case, For/Next, For Each, Do While, and finding the last row
- Writing fast, clean code: avoid Select and Activate, use With blocks, turn off ScreenUpdating and set calculation to manual during runs, then restore
- Working with data: reading ranges into arrays, writing results back in bulk, and copying and pasting values
- Debugging: breakpoints, stepping through code (F8), Debug.Print, and watches

- Running macros: keyboard shortcuts, buttons and shapes, the Quick Access Toolbar, and the Personal Macro Workbook
- **Lab:** Write a macro that imports a raw export, cleans it, applies formulas, formats the report, and saves a dated copy

Module 4: Advanced VBA: Forms, UDFs, and Error Handling (45 min)

- User-defined functions (UDFs): writing a Function, using it in the grid, and limits (no cell side effects)
- Error handling: On Error GoTo, Err object, cleanup blocks, and friendly messages
- Events: Workbook_Open, Worksheet_Change, and Worksheet_SelectionChange
- Input and output: InputBox, MsgBox, and Application.GetOpenFilename
- UserForms: text boxes, combo boxes, and buttons to build a data-entry form
- Looping through sheets and workbooks to consolidate files
- Exporting: saving sheets as PDF and emailing through Outlook (overview)
- **Lab:** Build a data-entry UserForm with validation that appends records to a table and logs each entry with user and timestamp

Module 5: Password Protection: Cells, Sheets, Workbooks, and Code (60 min)

Understand the layers first. Excel offers several separate protections. They solve different problems, and only one of them is real encryption.

Layer	What it does	Strength
Cell locking and sheet protection	Stops accidental edits to formulas and layout	Convenience only; passwords can be bypassed with freely available tools
Workbook structure protection	Stops adding, deleting, renaming, hiding, or moving sheets	Convenience only
File encryption (open password)	Requires a password to open the file	Strong encryption with a strong password; unrecoverable if lost
VBA project password	Hides and locks macro code from viewing	Deters casual users; not secure against determined attackers

Part A: Protect specific cells (15 min)

1. By default every cell is locked, but locking has no effect until the sheet is protected.
2. Select the cells users *should* be able to edit (inputs), press Ctrl+1, open the Protection tab, and clear **Locked**.
3. To hide formulas from the formula bar, select those cells and tick **Hidden**.
4. Protect the sheet (Part B) so the settings take effect.

Part B: Protect a sheet with a password (15 min)

1. Go to **Review > Protect Sheet**.
2. Enter a password (optional but recommended) and choose what users may still do: select locked cells, select unlocked cells, format cells, insert rows, sort, use AutoFilter, use PivotTables, and so on.
3. Confirm the password and click OK.
4. To remove it, use **Review > Unprotect Sheet** and enter the password.
5. To let different people edit different areas, use **Review > Allow Edit Ranges** to define ranges with their own passwords (or Windows user permissions), then protect the sheet.
6. Sheets can also be hidden or set to Very Hidden through the VBE Properties window. Treat this as obscurity, not security.

Part C: Protect the workbook and the file (15 min)

1. **Workbook structure:** Review > Protect Workbook, tick Structure, and add a password. Users can no longer add, delete, rename, move, or unhide sheets.
2. **Open password (encryption):** File > Info > Protect Workbook > Encrypt with Password. Anyone opening the file needs the password.
3. **Modify password (read-only recommended):** File > Save As > Tools > General Options, then set a password to modify, and optionally tick Read-only recommended.
4. **Mark as Final** and **Always Open Read-Only** are polite warnings, not security.
5. For sensitive data, combine encryption with proper storage and sharing controls (access-controlled locations, sensitivity labels, or IRM where your organization provides them).

Part D: Protect macros and automate protection with VBA (15 min)

1. **Lock the VBA project:** in the VBE, Tools > VBAProject Properties > Protection tab, tick Lock project for viewing, and set a password. Save and reopen to apply.

2. **Protect and unprotect through code:** use `ws.Protect Password:="..."` and `ws.Unprotect Password:="..."`. Never store real passwords in plain text inside code you share; prompt for them or keep them out of the workbook.
3. **Let macros edit protected sheets:** either unprotect, edit, and re-protect inside the macro, or set `UserInterfaceOnly:=True` when protecting (this setting is not saved with the file, so set it again in `Workbook_Open`).
4. **Good practice:** always re-protect in an error handler's cleanup block so a failed run never leaves a sheet unprotected.

Best practices and cautions

- Use long, unique passwords and store them in a password manager. Microsoft cannot recover lost Excel passwords.
- Do not rely on sheet or VBA passwords to protect confidential data; encrypt the file or keep sensitive data out of the workbook.
- Separate the workbook into Input, Calculation, and Output sheets so protection rules are simple to apply.
- Keep an unprotected master copy in a secure location.
- **Lab:** Take the case-study workbook: unlock only the input cells, hide formulas on the calculation sheet, protect each sheet with a different set of permissions, protect the workbook structure, add an open password, lock the VBA project, and update the macro so it still works on the protected sheets.

Capstone and Wrap-Up (30 min)

- **Capstone (20 min):** Deliver a secure commissions tool containing (1) a formula-driven calculation sheet using LET and a named LAMBDA, (2) a macro button that refreshes data and generates a PDF report, (3) a data-entry UserForm, and (4) protection on cells, sheets, workbook structure, and the VBA project.
- **Peer review (5 min):** Swap workbooks and try to break each other's inputs and edit protected areas.
- **Recap and next steps (5 min):** Key takeaways, post-training quiz, feedback form, and resources for Office Scripts, Power Query, and Power Automate.

Materials Needed

- Start and solution workbooks (.xlsx and .xlsm) for every lab
- Case-study dataset (about 5,000 sales rows with reps, regions, products, and dates)

- Cheat sheets: advanced functions, VBA syntax and object model, and the protection layers table above
- VBA code snippet library (last row, loop sheets, protect/unprotect, error handler template)
- Pre- and post-training quizzes

Assessment

- Pre/post quiz covering formulas, VBA, and security concepts
- Lab completion checked by the trainer during rounds
- Capstone checklist: formulas return correct results and handle errors, macros run without debugging and re-protect sheets, only input cells are editable, the file opens only with a password, and code is commented

Trainer Tips

- Set expectations early: sheet and VBA passwords prevent accidents and casual tampering, but they are not strong security. Encryption of the file is the only strong layer.
- Have everyone choose a simple practice password for labs, written on the handout, so nobody locks themselves out of their own exercise file.
- Make a backup folder of unprotected solution files in case someone forgets a password.
- Run VBA labs on Windows desktop Excel; Excel for the web does not run VBA, and Mac support differs for some objects and forms.
- Stress the habit of Option Explicit, commenting, and testing macros on copies of data before running on live files.
- Keep a parking lot for topics outside scope, such as advanced class modules, API calls, and Power BI.