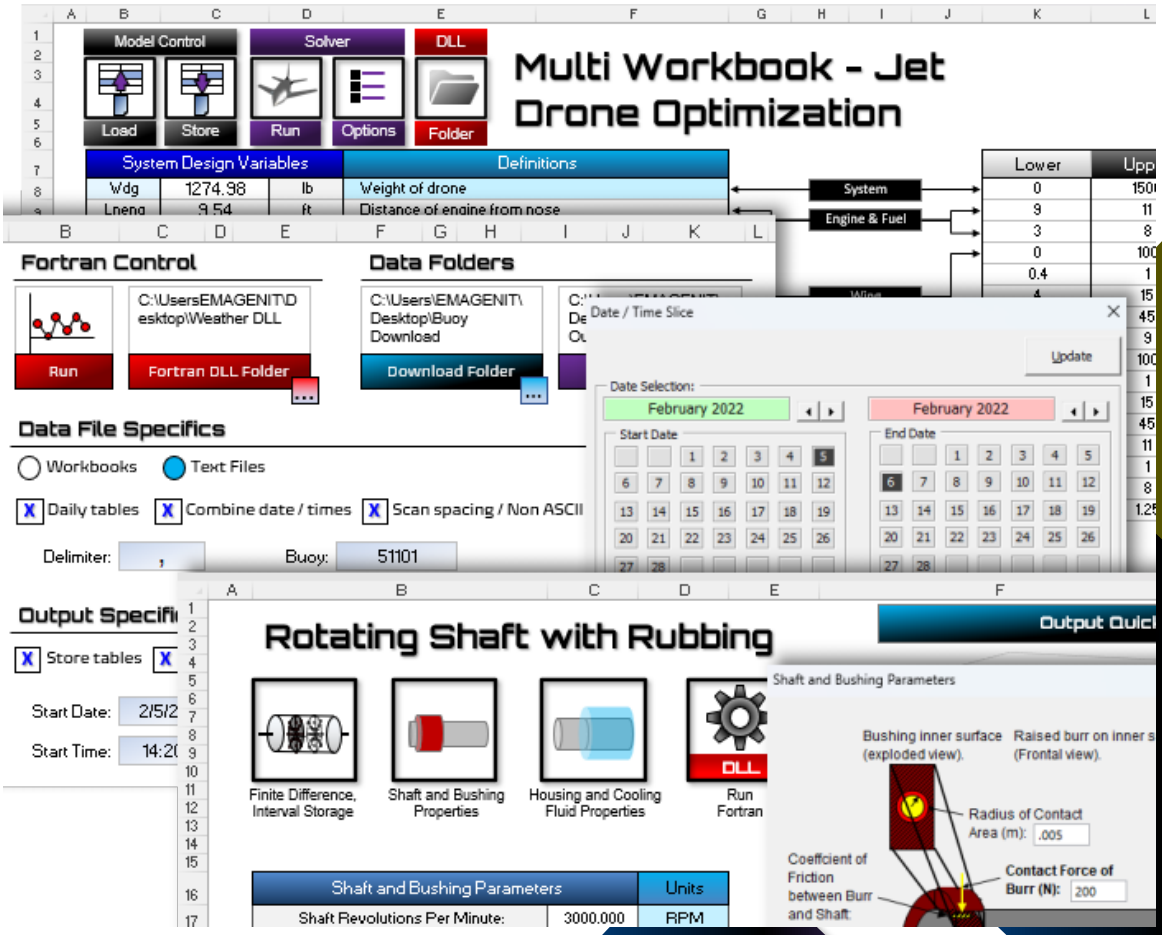
EMAGENIT

State of the art Excel training.

Mixed Language Programming

with Excel VBA and Fortran



Why rewrite verified Fortran code? Use Excel VBA to enhance its UIs and run it.

Learn to combine the GUI, data processing, and charting abilities of Excel VBA with the speed of Fortran DLLs to create powerful modeling, data, and reporting tools.

www.emagenit.com >

How our class can help you.

Our 3-day class shows you hands-on how to create powerful tools that fuse Excel VBA with Fortran. Stressed is how to use Excel VBA as a front and rear end information processor and GUI for your new or legacy FORTRAN code.

What you'll master in our class.

Our training covers all the Excel VBA elements necessary to gather, pass, and return Fortran DLL data.

Also covered is how to upgrade your Fortran code so it can be compiled as a DLL and run from Excel VBA. The class uses Intel Fortran and Visual Studio to create Fortran DLLs (both can be downloaded for free).

Also reviewed is how to assemble models; automate Excel's data tools and charts; command text files; and how to build hi-tech GUIs from userforms, ActiveX controls, shapes, worksheets, pictures, and events.

Join us and our class will show you how to enhance your Excel VBA apps with the computational speed and abilities of Fortran.

Who should attend the class?

Engineers, scientists, and technicians. Class examples will be determined by those in attendance.

Minimum Excel skills needed for the class.

Select this Excel training if you or your group have:

- Operated Excel's data processing tools like Filter, Text to Columns, Remove Duplicates, Sort…
- Used worksheet functions before and built formulas
- Created charts, Excel tables, and used drawing shapes
- Used VBA before in a general capacity
- Been exposed to the concepts of basic programming like loops, logic, functions, and variables
- Used Fortran before in a basic capacity

How we run the class.

We focus our training on what our customers need. When training begins, we analyze those needs and shift our outline appropriately. We will stress or add topics that our customers want.

Class formats and signup.

In-Person, Virtually, and Onsite. Our live hands-on classes can be attended virtually or in-person. Please visit our public signup page for class times and pricing >. Contact EMAGENIT directly at 805.498.7162 for more information about our onsites.

Key Excel VBA topics covered in class.

- Full review of the Excel VBA language
- Fortran DLL calling rules form VBA
- Fortran / VBA argument passing rules including arrays and data types
- How to use VBA to loop through workbooks and worksheets to gather Fortran DLL data
- Controlling the Windows folder and file system with VBA
- How to use Excel VBA and worksheet functions to pre/post process Fortran DLL data
- How to use Excel VBA to create, read, and control Fortran DLL text files
- How to use ActiveX controls, worksheets, and userforms to design UIs that command Fortran
- How to design Excel models that interact with Fortran DLLs
- How to use VBA to upload Fortran DLL information
- How to use VBA to download Fortran DLL output into charts, shapes, and worksheet tables

Detailed class syllabus.

Day-1

Why Run Fortran from Excel VBA?

- The uses of User Defined Worksheet Functions (UDFs) and Sub procedures in Excel VBA
- Running Fortran routines from UDFs; powerful modeling, simulation, design optimization and data analysis capabilities
- Running Fortran routines from Sub procedures; powerful interface, reporting, and event driven application development
- Running Fortran routines from Property procedures; object driven application development
- Using the powerful reporting capabilities and data visualization features of Excel coupled with Fortran

Excel VBA Language Review Emphasizing Fortran DLL Communication

- How to use the Visual Basic Editor, its debugging tools, and module types in Fortran DLL construction
- A review of the Excel VBA language syntax including objects, properties, methods, variables, data types, constants, arrays, operators, expressions, loops, and logic
- Overview of procedure argument design in Excel VBA; using by reference and by value to pass arguments
- Overview of Excel VBA data types that are used to pass argument data to a Fortran DLL routine
- Creating and working with VBA arrays; automatically dimensioning an array with the ReDim statement; clearing arrays; tricks to increase speed

- Creating a user defined type that can be passed to a derived data type (structure) in a Fortran DLL

Must Know VBA Functions and Excel Worksheet Functions Used in Excel VBA / Fortran DLL Apps

- Overview of how to run VBA functions and Excel worksheet functions in VBA
- Using the C... VBA functions to convert data types, using data type coercion in VBA
- Performing complex and matrix operations in VBA using Excel worksheet functions
- Analyzing dates and times using the Excel worksheet and VBA Date / Time functions
- Using Key Excel worksheet functions like MATCH, VLOOKUP, COUNTIF, COUNTA, SUMIF, SUMIFS, COUNTIFS, EVEN, and ODD in your VBA code
- Figuring out the bounds of VBA arrays using the UBOUND and LBOUND VBA functions
- Using key VBA functions like Split, Instr, Len, Left, Right, Replace, Mid, Chr, Format... and the & concatenation operator on VBA strings

Controlling and Looping Through Workbooks, Worksheets, and Folders for Fortran DLL Data

- Storing Fortran DLL data in multiple workbooks; the key to developing an effective data storage system in Excel
- Opening, saving, adding, and closing workbooks with Excel VBA
- Using the Set statement to track workbooks and worksheets in Excel VBA
- Using Excel VBA to add, move, delete, and rename worksheets
- Creating, deleting, moving, and coping folders and files with Excel VBA

- Scanning folders for workbook-based Fortran DLL data using VBA For...Each Next loops, the FileSystemObject, and logic
- Looping through open workbooks and worksheets to find Fortran DLL data and model information using VBA For...Each Next loops and logic

Using Excel VBA to Process Input / Output Fortran Data on the Worksheet

- The Excel worksheet, the perfect place to pre/post process Fortran DLL data
- What is a range and how to control it with VBA using Cells, Range, Rows, Columns, CurrentRegion, ActiveCell, Find...?
- Using Excel VBA to automatically lock onto worksheet table data that changes size, shape, and position
- Copying and pasting worksheet tables and sub sections using Excel VBA
- Finding worksheet table headers with VBA using Find, Offset, and Worksheet functions like MATCH and COUNTIF
- Using Excel VBA to reorganize worksheet table data including inserting/deleting/moving rows, columns, and cells
- Converting Excel ranges and sub ranges into VBA arrays that can be passed to Fortran DLL routines
- Using Excel VBA to read/set cell and range names and loop through them searching for specific cell values
- Controlling Excel's data features like AutoFilter, Advanced Filter, Sort, Remove Duplicates, and Text-to-Columns with VBA
- How to tie Excel's data features into Excel VBA using loops that process massive amounts of worksheet data

Day-2

Calling Fortran DLLs from Excel VBA Procedures

- Creating a Declare statement in Excel VBA to call a Fortran DLL Subroutine or Function routine
- Calling a Fortran DLL Subroutine from Excel VBA
- Calling a Fortran DLL Function routine from Excel VBA
- Using argument lists to pass data between Excel VBA and Fortran
- Function design rules for calling Fortran DLL routines
- Sub procedure design rules for calling Fortran DLL routines
- Property procedure design rules for Calling Fortran DLL routines

Designing a Fortran Routine to be Called from Excel VBA

- Coding compiler directives for your Fortran DLL routines
- Using the STDCALL, ALIAS and DLLEXPORT attributes to define your Fortran DLL entry points
- Using the REFERENCE and VALUE attributes correctly in your compiler directive to specify argument passing convention
- Data typing Fortran dummy arguments to match their VBA counter parts
- Creating Fortran dummy arguments to accept VBA numbers
- Designing Fortran dummy arguments to accept VBA strings with variable lengths
- Laying out Fortran dummy arguments to accept string arrays
- Creating Fortran dummy arguments to accept VBA arrays as explicit arrays, adjustable arrays, and assumed size arrays
- Coding a Fortran dummy argument to accept a VBA user defined type including strings

Passing Different Argument Types from Excel VBA to a Fortran DLL Including Strings

- Passing a number including Booleans from Excel VBA to a Fortran DLL routine
- Passing a numeric array from Excel VBA to a Fortran DLL routine
- Passing a string from Excel VBA to a Fortran DLL routine
- Passing a string array from Excel VBA to a Fortran DLL routine
- Passing a user defined type from Excel VBA to a Fortran DLL routine
- Passing a complex number from Excel VBA to a Fortran DLL routine

Passing Data from Fortran Back to Excel VBA

- Passing a number including Booleans back to Excel VBA from Fortran
- Passing a number array back to Excel VBA from Fortran
- Passing a string and string array back to Excel VBA from Fortran
- Passing a Fortran derived data type back to Excel VBA from Fortran

Passing Data to Fortran from Excel VBA Using Text Files

- Assembling text files on the Excel VBA side using FileSystemObject and TextStream objects
- Quickly assembling strings on the Excel side in VBA using key functions and concatenation strategies
- Designing Fortran routines that quickly read text files into variables and arrays
- Using VBA, ADO, and SQL to query large Fortran output files and return the result to a worksheet

Compiling and Debugging a Fortran DLL

- Compiling a Fortran DLL in Visual Studio using Intel Visual Fortran
- Debugging a Fortran DLL from Fortran
- Debugging a Fortran DLL from Excel VBA
- How Excel VBA reacts when it encounters an error in a Fortran DLL

Day-3

Creating Program Interfaces (UIs) for Your Excel VBA / Fortran DLL Apps

- When to design program interfaces on worksheets vs. VBA userforms
- Userform design and coding strategies in Excel VBA
- When to use ActiveX controls to design program interfaces versus using Excel's built-in features like cells, Data Validation, Conditional Formatting, Excel tables...
- Using Data Validation and Conditional Formatting to quickly create program interfaces that control your Excel VBA / Fortran programs
- Using Excel VBA to control ActiveX controls like list boxes, combo boxes, check boxes, option buttons... on worksheets and on VBA userforms
- Creating folder and file pickers using the Application object in Excel VBA
- Using the Windows Registry to store your program data

Assembling Excel Models That Run Fortran DLLs

- Laying out model input and output ranges on the Excel worksheet
- What is a user defined worksheet function (UDF) and how to build one?
- Designing UDFs that collect worksheet parameters and call Fortran
- Designing UDFs that return scalar and array values back to the worksheet

- How to tie multiple workbook models together that run Fortran
- How to run Solver with Excel VBA/Fortran DLL models

Creating Charts with Excel VBA to Display Your Fortran DLL Output

- Automating chart construction for Fortran DLL output using Excel VBA
- Using VBA arrays, not ranges, to build charts; perfect for emailing reports
- Refreshing chart data with Excel VBA, no rebuild
- Using VBA to position multiple charts on a worksheet for reporting purposes
- Formatting charts (axes, data points, labels, positioning labels...) with Excel VBA
- Creating limit lines for charts using Excel VBA
- Using VBA to color chart data points that are below or above specified limits

Using VBA to Create Worksheet Tables and PivotTables to Display Your Fortran DLL Output

- Methods for filling in worksheet tables automatically using Excel VBA
- Using Excel VBA to create multiple worksheets and separate report data out to them
- How to use VBA to automatically build formulas in worksheet tables
- Quickly formatting worksheet tables using Excel VBA
- Automating PivotTables, Excel Tables, and their features using Excel VBA

Visualizing Your Fortran DLL Output Using Excel's Drawing Shapes

- Using Excel VBA to command primary shapes like rectangles, circles, lines and shapeforms to form technical drawings in Excel

- Formatting drawing shapes and altering their text using Excel VBA
- Using worksheet cells and trigonometry to help position drawing shapes
- Basics strategies for using Excel VBA to align dimension lines and values around drawing shapes