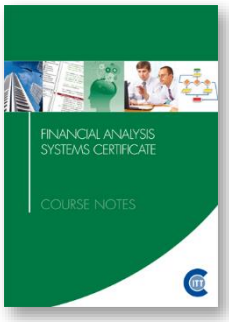
Financial Analysis VBA Certificate



Australia's premier Excel VBA course for finance professionals

A 48-hour Excel programming short course for professionals in the finance & accounting professions looking to better automate models and analytical applications within their enterprise. Effective automation techniques for gathering, modifying and updating models, analytical workbooks and reports is taught.

VBA macro programming techniques are taught from the ground up, assuming no prior VBA experience, but familiarity with Excel for modelling and analysis.

Modes of Study

- Instructor-led online program, consisting of 12 sessions over 3 weeks (day mode) or six weeks (evening mode). Opportunities for extra remediation sessions will arise on presentation days or after presentation subject to time available.

Key Outcomes

- Learn a proven approach for developing automation in Excel-based projects.
- 48 or more hours of continuing professional education for accountants, including the instructor-led sessions and required self-study.
- Gain insights into the development of robust code for routine analytical & reporting tasks.
- Learn appropriate uses of VBA in modelling tasks. How to avoid inappropriate uses and risk factors when code is used in a financial model is also covered.
- Learn effective techniques for importing & validating data.
- Learn how to develop routines from scratch with intent, rather than relying on recorded code or kludging code discovered on the internet.
- Learn how to interact with the user through built-in & custom dialog boxes & VBA functions.
- Understanding of how to add your tools & macros to the Ribbon UI (user interface) and Excel's own context menus.
- Recognise when it is appropriate to use built-in features & when to add custom functionality.
- Student retains a library of useful routines dealing with common tasks in Excel development, for use in their own future projects.

The Course Delivers

- A thorough grounding in the VBA language.
- Familiarity with the VBA integrated development environment.
- Familiarisation with the tools to work with & develop VBA code in Excel.
- An overview of the whole Excel object model and a deep dive into critical objects, properties & methods to develop effective, useful & productive code.
- Standardised layout & naming conventions are explained and demonstrated.
- Proven approaches to development of Excel VBA projects from planning to rollout.
- Effective design approaches for user interface & custom dialog boxes.
- Understanding of the debugging & testing procedures in the VBA development environment.
- Demonstrates how to construct custom functions to extend Excel & reduce complexity in workbooks.
- Demonstrates how to trap and handle errors.
- Demonstrates how to deploy add-ins that extend Excel's native functionality. Coding issues for effective deployment of add-ins and custom functions is covered.

Who Benefits

- Financial modellers looking to automate key tasks in their models, such as Goal-Seeking, handling circularity in interest calculations and sensitivity analysis, or who wish to develop custom functions to manage spreadsheet complexity.
- Business Analysts responsible for routine forecasting & performance analysis who wish to automate and streamline reporting & analytical tasks, such as the importation of data, data cleansing and distribution of results.
- Fund accountants & managers in equity & property funds looking to automate reporting, valuation, analysis and report distribution tasks.
- Procurement, project planning & logistical staff looking to automate planning & reporting tasks.
- Anyone who uses Excel to perform their role, looking to automate routine tasks, such as data import, data cleaning, workbook distribution, data entry and reconciliation processes.

Benefits for the Employer

- This course provides essential knowledge & design principals to aid staff in creating robust applications in Excel and avoid common pitfalls arising from an unstructured learning path.
- It is designed to replace the commonplace ad hoc recording & copying of VBA code with code robustly designed for its purpose. Understanding of important issues such as the appropriate use of code & how to avoid common design flaws & inappropriate bypassing of important Excel controls & features is delivered.
- Many managers rightly have concerns about the unmanaged proliferation of macro code within the workbooks in their departments. If it is developed without clear guidelines & standards and is just kludged-together, then it represents a significant risk. However, well-designed code appropriately deployed can significantly improve productivity & can add to the control of errors in repetitive & time-sensitive tasks. Without training this objective is hard to achieve.
- The Financial Analysis VBA Certificate is specifically designed to take knowledgeable Excel users & not only impart essential technical knowledge to enable them to take the next step, but to canvass issues around design, anticipation & trapping of errors and to demonstrate proven techniques that quickly & robustly deliver solutions.
- Standardisation of approach, coding conventions and the re-use of code is an important outcome.

About VBA & Macros

- VBA stands for Visual Basic for Applications. It is the automation language provided with Microsoft's Office suite of applications & enables the development of automation & the extension of the user interface in those applications.
- Macros are a common name for the procedures developed in this language. The actual derivation of the word goes back to the early days of PC computing. A macro (which should more properly be called a procedure) is a separate executable application or applet which can be attached to a tool bar, a button or other control, graphical objects or linked to an event in the host application to handle a procedural step, automate common actions, or respond to changes in host documents.
- Macros might be responsible for the automation of the importation of data from external sources, the changing of document contents, prompting a user to complete the entry of data to be stored in cells or forms and may be utilised in the quick and effective distribution of reports, or input sheets and the collation of the results.
- The language taught is VBA, not VB.NET. VBA and the integrated development environment provided as a standard extension of Office is the development language used & taught in the course. The .NET languages used in Visual Studio .NET can be used to develop add-ins and extensions for the Office application suite, but that is a more complex development environment than that used for this course.
- New automation languages and tools for use with Office 365 online are becoming available, but they do not provide the complete set of robust automation available through VBA running on a desktop machine, nor do they as yet provide the same rich set of event-driven programming options. They do provide for some interaction between workbooks and other applications, and in the web version of Excel, some button-based commands can be added, however, the interaction with the spreadsheet at a lower level is not yet supported.

Is the content applicable to other Office applications?

- VBA is provided as a development language in most of Microsoft's Office suite. Most users, however, are generally interested in automating Excel, Access & Word. The course content relating directly to the VBA language, intrinsic data types, operators, dealing with the file system, calling user forms and the programming of common Office elements such as tool bars, context menus, the Ribbon and file search is applicable to all Office applications, so to that extent, therefore, the knowledge is transferrable.
- On the other hand, much of the course design rests on Excel specific features, such as worksheets, ranges, cells, formulas and Excel functions & is therefore heavily skewed to working with Excel & the Excel Object Model which exposes the elements of Excel & its features to the end-user programmer.
- The concepts taught about dealing with object models & the programmable elements of an application in a generalised way will apply to working with other object models, although the specifics will be very different. If you have specific questions relating to these matters, feel free to contact us to discuss them.
- A brief introduction to the object models of both Word and Outlook is included in this course in the modules demonstrating cross-application Automation for the sending and handling of email and the generation of Word documents and their export to PDF files. But this does not represent a deep-dve into their object models and guidance on how best to structure applications running in those applications alone.

Testimonials

Most of these testimonials were offered some months after course completion, not just as polite feedback on the final day of the course.

- "I just wanted to say thank you. The skills you taught and knowledge you imparted during the course I attended have made a real difference in my work. I am enjoying my days immensely and doing some interesting and innovative work, no longer limited - at least, not so much - by my (in)ability with Excel." - Todd A
- "The course was fantastic. It was full of useful and practical methods that can be applied to my daily tasks." - Erica H
- "I would like to say thanks for the inspirational course! I've found that my productivity at work has gone through the roof. Without you, none of that would have been possible. So thanks again. " - Simon S
- "Great examples used, great responses to real life situations, great presenter" - Fernando V
- "Your passion and patience was really, really appreciated. I was a bit worried when the course started that it would be too much for me, but following your guidelines, I have surprised myself just how much I can achieve. This has been a great add-on to my professional skill set! Thanks so much!" - Gemma S

Detailed Course Program

Session	Session Focus	Session Description
1	*Technical Session* Introduction to VBA Part 1	An introduction to VBA and the integrated development (programming) environment. The macro recorder is introduced and three essential tips for its effective use are demonstrated. A discussion of what the macro recorder is useful for, and when it should be avoided. Using the Immediate pane to execute ad-hoc commands, derive object state and property values is demonstrated.

Session	Session Focus	Session Description
2	*Technical Session* Introduction to VBA Part 2 *Project # 1* Toolbox macros	Understanding and working with data types. Selecting appropriate types for different tasks. Declaration of subroutines and functions, arguments and function return types. In project # 1 some straight-forward and simple, but useful macros are developed that can be stored in the participant's personal macro workbook to start their toolbox of handy routines.
3	*Project # 2* Workbook Explorer	In project # 2, a tool for reporting the contents and settings of a workbook is developed. The project allows for the exploration of key collections, objects, and properties in Excel's object model as a basis for the report contents. Looping and conditional statements are demonstrated in some depth. Output to the report explores the process of updating and modifying workbook contents, especially the use of styles to minimise the need to make code modifications to meet future formatting requirements. Even experienced Excel users may discover some startling information. This project will prove useful for students after the course in preparing, managing and diagnosing workbooks used in their projects.
4	*Project # 2 (continued)* Workbook Explorer	This session sees the completion of the Workbook Explorer project and introduces approaches for prompting for file and folder locations and saving workbooks. Standardised code to avoid naming conflicts is developed. By the end of this session, the workbook explorer includes reports that document the properties of the workbook, the contents of the workbook and its worksheets, including the names and their definitions and styles used in the workbook. Library code for future use is developed in this project.
5	*Project # 3* Importing CSV Data *Project # 4* Importing Excel Data	In this session two smaller projects are undertaken that involve the importation of data from CSV files and Excel workbooks, one utilising data stored in known web locations, the other prompting the user to select the source file(s). Data is imported and formulas updated to enable reports to reflect the new content. Library code for future use is developed in these projects.
6	*Project # 5* Data Management	Utilising the data imported in projects # 3 and 4, this project explores tasks and approaches for moving, duplicating, and updating data in a workbook. Sorting and filtering routines are developed. In this session the use of copy and paste is demonstrated, and when it should be resorted to - almost never – is discussed. Dealing with common issues in working in worksheets to determine the actual used range is canvassed, and a useful routine is developed. Library code for future use is developed in this project.

Session	Session Focus	Session Description
7	*Project #6* Lease Amortisation Schedule	In this session a custom dialog box is developed (User Form class) to provide for the prompting of inputs for an amortisation schedule. This explores handling user-entered data, validating it and managing dynamic relationships between data entered in a dialog box. The key control events and how and when to use them are discussed and sample code developed.
8	*Project # 6 (continued)* Lease Amortisation Schedule	This session uses the custom dialog box created in the previous session and demonstrates how to call the dialog box and process the results derived from user input to create the schedule. Associated procedures are developed to reset the schedule and to implement conditional formatting in the table to build a complete, re-usable, stand-alone project. Library code for future use is developed in this project.
9	*Project # 7* Data Exporter Project # 8 Data Collator	The process of splitting data from a workbook into several reporting / review workbooks by cost centre, location or some other distinguishing code is demonstrated. This involves building and formatting the workbooks, allocating and updating the data and saving and naming the workbooks appropriately. The second project collates the data from updated copies of the workbooks created in Project # 7, tracking and identifying changes in the data in the master workbook. Library code for future use is developed in these projects.
10	*Project # 9* Word Doc Merge Control Object *Project # 10* Word Doc and PDF Foundry	Sessions 10 and 11 demonstrate how to create and use an object to streamline business logic in a complex project. This allows the user to gain an appreciation of what is happening in the Microsoft-provided objects they work with. Property and method procedures are introduced, and the design of the business object interface is discussed. In Project # 9, an object to control the merging of data from Excel into Word templates is developed. In Project # 10, that new object is used in a project that creates Word documents and PDF files by merging key Excel-based data into the text from a Word template. The documents are also exported to PDF format. This module includes a brief overview of the Word object model. Library code for future use is developed in these projects.
11	*Project # 11* Mail Generation Control Object *Project # 12* Email Foundry	In Project # 11, an object is created that enables the automated creation of emails in Outlook, including merging substitution terms into the email html body text, adding attachments, and adding the user's signature. Project # 12 uses that new object and distribution lists maintained in Excel to manage the creation of Emails in Outlook. Workbooks created in Project # 7 are attached to the emails. The sent emails can be optionally archived to a nominated folder. This module includes a brief overview of the Outlook object model. Library code for future use is developed in these projects.

Session	Session Focus	Session Description
12	*Project # 13* Email Retriever Wrap Up	Project # 13 scans the user's Outlook inbox for emails which need to be retrieved. The retrieved emails are moved to an actioned folder, and the attachments detached to a folder for processing. The detached workbooks could be processed by a collator, such as that developed in project # 9. This module includes a brief overview of the Outlook object model. Library code for future use is developed in this project. The final part of this session is a review of the material covered in the course, a final check on the library code developed during the course for use in future projects.

Detailed Course Contents

Introductory macro techniques using VBA

- Introduction to macros & VBA
- Macro recording & editing recorded macros
- How to record generalised macros
- Recording macros with absolute references
- Recording macros with relative references
- Where to store macros
- Locating & activating macros
- Assigning recorded macros to worksheet controls
- Associating macros with Ribbon and QAT controls
- Introduction to the IDE

introduction to elements of VBA

- VBA keywords & syntax
- Operators & Statements
- Procedure naming & declarations
- Intrinsic data types
- Variable naming conventions & declarations
- Object variables
- Initialising & referencing objects
- Array variables, declaration & re-dimensioning
- Determining array dimensions
- Function Arguments
- Lifetime, visibility & scope of variables & constants
- Use of constants to aid maintenance
- Naming conventions
- The Excel Object Model
- The Office Object Model
- The MS Forms 2.0 Object Model
- Module & procedure naming
- Compiler options: Option Explicit, Option Base, Option Compare, Option Private Module
- Compilation & compile on demand

Program flow control

- If statements
- If blocks
- Select Case blocks
- Do While / Until blocks
- While / Wend blocks
- For loops using counters
- For Each loops using object variables
- With blocks

Layout & Programming Style

- Referencing conventions
- Naming conventions for procedures, variables & constants
- Use of With blocks to improve layout
- Code layout conventions & indentation to create more readable code blocks to aid in maintenance & debugging
- Use of subroutine calls
- Use of function calls
- Common syntax errors & their correction

Procedure & Function declaration

- When to use modules & worksheet modules
- Procedure declaration & scope
- Declaring arguments
- Determining argument & return value types
- Creating optional arguments & default values
- Using ParamArrays
- Passing values to subroutines vs global variables, how to avoid global variable problems
- Passing object references
- ByVal & ByRef keywords
- Calling subroutines & functions
- Handling function return values

Working with the Workbooks collection

- Working with Application object (Excel)
- Working with the Workbooks collection
- Working with the Workbook object
- Key workbook properties
- Creating new workbooks
- Creating new workbooks from templates
- Opening, closing & saving workbooks
- Responding to key workbook events
- Changing key workbook properties

Working with the Sheets collections

- The Sheets, Worksheets & Charts collections
- The Worksheet, Chart & DialogSheet objects
- Key sheet properties
- Selecting & activating sheets
- Creating new sheets
- Creating new sheets from templates
- Adding, deleting, copying & moving sheets
- Responding to key worksheet events
- Changing key sheet properties
- Page setup & printing sheets
- Sheet visibility & sheet protection
- Visibility & protection effects on programming
- Iterating through workbooks with mixed sheet types

Working with Ranges

- The Range object
- Cells collection
- Selecting or referencing a cell or cells
- Selecting or referencing a row or rows
- Selecting or referencing a column or columns
- Selections using SpecialCells & CurrentRegion
- Selection or referencing Unions and Intersections
- Navigating through the cells collection of a range
- Working with the ActiveCell property
- Working with the Selection property
- Reading & changing the values of key properties, such as Formula, FormulaR1C1, Value, Value2 & Text
- Using Find to locate matching cell or cells
- Using MATCH to find matching cell or cells
- Working with named ranges
- Selecting & updating ranges
- Selecting & updating dynamic ranges
- Selecting ranges by address or by Name reference
- Using Styles & formatting properties to change the appearance of ranges
- Updating formulas & values
- Working with Copy, Paste & PasteSpecial
- Replacing Copy & Paste with direct updates

Working with the Names collection

- Reducing code load, allowing for user interaction & maintenance through the use of Names
- Key application settings effectively managed through Names
- Working with the Names collection
- Working with the Name object
- Key Name properties
- Creating, deleting & editing Names
- Selecting named ranges
- Retrieving values from & updating values of named ranges

Working with the Styles collection

- Reducing code load allowing for user interaction & maintenance through the use of Styles
- Working with the Styles collection
- Working with the Style object
- Key style properties
- Creating, deleting, editing & applying Styles
- Removing unwanted Styles from a workbook

Custom functions

- Creating custom functions to manage formula complexity
- Creating custom functions to provide specialised calculations not provided by the Excel function library
- Rules for functions to be called from a worksheet
- Return values & arrays
- Handling argument errors & returning errors from functions
- Detecting missing arguments & argument types
- Default values for arguments
- Sample custom functions
- Deploying custom worksheet functions

Advanced techniques to improve performance

- Locking screen updates
- Selecting appropriate variable types
- More efficient methods of referencing
- Bypassing the clipboard
- Indicating progress to the end user
- Locating a specific workbook in memory
- Prompting for the selection of a workbook from a directory

MS Forms 2.0 & custom dialog boxes

- Understanding control types
- Understanding control properties
- Understanding control events & event execution order
- Understanding containers & the controls collections
- Responding to control events
- Selecting appropriate controls
- Setting default values
- Responding to user inputs & selections
- Validating user inputs & selections
- Interpreting & formatting text box values, especially problematical values
- Dynamically linked controls
- Designing custom dialog boxes
- Instantiating dialog boxes
- Dialog box life cycle
- Referencing controls in dialogs
- Transferring data between the worksheet, named ranges & dialog box controls
- Determining which button closed a dialog box
- Tab order, accelerators & short cuts

Error handling

- Error object & key properties
- On Error statement
- Writing standard error handlers
- Trapping & responding to errors
- Ensuring clean-up code executes
- Closing opened workbooks, without saving changes
- Resetting the user interface, undoing changes

Key Excel techniques

- Working with the Application object to manage core settings & access key references
- Working with the Workbooks collection & referencing workbooks by code
- Working with worksheets & the Worksheets, Charts & Sheets collections
- Working with range objects & understanding the Range, Cells, Rows, Columns, Offset, Resize & Areas properties
- Range validation
- Working with ranges using names
- Range formatting, using Styles, Font, Interior, Borders properties & their associated object structures
- Entering formulas under macro control
- Accessing & updating the values in cells using Value, Value2 & Text properties
- Using counted For loops & For Each loops to process blocks of cells
- Copying & pasting data using the clipboard
- Calling built-in Excel dialog boxes
- Utilising Excel's built-in functions to optimise code

Testing & Debugging VBA Code

- Program testing & debugging
- Debugging with step mode
- Using break points
- Using watch window & locals window to view variables
- Using the Immediate Pane to evaluate expressions
- Interpreting run-time errors
- Trapping & responding to run-time errors
- Planning for recovery

Assessment

- An optional Final assessment may be undertaken either individually or in groups after the conclusion of the course.

Excel Version

- The course materials are fully compatible the following versions of Excel:
  - Excel for Windows 2010 to 2019
  - Excel 365 – desktop installation of Excel 2019
  - Excel for Macintosh 2016 or later
  - Version specific instructions are given where these vary between versions.

Course Materials

- The course materials include a 250+ page course text
- A full set of blank and worked sample workbooks as well as a number of reference documents are available for download online.
- Short video presentations are available to the student to support their studies, covering key techniques, features of the development environment, use of programming and debugging tools.

Course Offerings

Public offerings of the Financial Analysis VBA Certificate are conducted online as instructor-led hands-on sessions, using Zoom. They may also be conducted in-house in a classroom or via Zoom.

Public Courses, Pricing & Discounts

For details, visit the calendar page at our web site (www.clarksonitt.com/calendar) for the schedule of upcoming courses and pricing.

Generous discounts are available for the following categories, and may be combined where more than one category applies:

- past students of Clarkson ITT or UTS presented courses
- group bookings of three or more attendees
- early-bird pricing for bookings eight weeks or more before commencement.

Discount codes are shown on the course booking pages.

A limited number of places are available on each course for past students of this course to attend again at no charge. There is a modest charge for new materials if attending an upgraded course with significantly newer content.

Online Course Presentation

Public courses are conducted using Zoom, with numbers limited to maintain an effective facilitator to participant ratio. The user needs to have the Zoom application installed and register; it is not possible to attend the sessions via browser. A second monitor, or a large format screen is recommended, so that the participant can do the hands-on exercises whilst viewing the facilitator's desktop.

During certain parts of the program, the participant will be required to enable their camera and/or microphone to enable dialog with the facilitator(s).

Private courses

In-house offerings of the Financial Analysis VBA Certificate can be arranged for presentation online via Zoom or in a classroom anywhere.

We can assist with the hire of suitable training facilities where not available in-house. Requirements for facilities are discussed on our web site. For a quotation or to discuss your particular needs, call +61 (0) 417-777-750.

About the Course Author and Lead Presenter

Following an early career in both public and commercial accounting, James Clarkson has thirty years of experience as a presenter of professional programs and more than thirty consulting & model building for corporate clients.

He has gained wide experience in model & analytical report design & development and also has wide experience in application development using Microsoft Office (especially Excel, Word and Outlook), Visual Basic & Visual Basic for Applications (VBA).

The course is full of practical advice which has been gained from real-world problems and solutions that James has worked on over this time as a consultant and developer.

He is an outstanding educator, demonstrating the ability to clearly explain difficult and technical subjects and consistently rates above 90% for satisfaction across a number of measures on student evaluations.