**VBA Cheat Sheet for Excel**

**Basics**

**Variables**

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Subroutine Declaration | Sub NameOfSubroutine() |
| End Subroutine | End Sub |
| Message Box | MsgBox "Message Text" |
| InputBox | VariableName = InputBox("Prompt Text") |
| Comment | ' Comment Text |
| Open Workbook | Workbooks.Open "Path\Filename.xlsx" |
| Close Workbook | ActiveWorkbook.Close SaveChanges:=True |
| Find | Range("A1:B10").Find("SearchText").Select |
| Range | Range("A1") |
| Select a Cell | Range("A1").Select |
| Select a Range | Range("A1:B10").Select |
| Cell Value | Range("A1").Value |
| Clear Cell | Range("A1").ClearContents |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Declare Variable | Dim VariableName As DataType |
| Assign Value to  Variable | VariableName = Value |
| Dynamic Arrays | Dim VariableName() As DataType |
| Static Arrays | Dim VariableName(Start To End) As DataType |
| Multiple Variables | Dim Variable1 As DataType, Variable2 As DataType, ... |
| Global Variables | Public VariableName As DataType |
| Constants | Const ConstantName = Value |
| Boolean | True or False |
| Integer | -32,768 to 32,767 |
| Long | -2,147,483,648 to 2,147,483,647 |
| Double | -1.79769313486232E+308 to 1.79769313486232E+308 |
| String | "Text" |
| Date | #yyyy/mm/dd# |

**Operators**

**Procedures**

|  |  |
| --- | --- |
| **Category** | **Operators** |
| Arithmetic  Operators | +, -, \*, /, ^ (exponentiation) |
| Comparison  Operators | =, <, >, <=, >=, <> (not equal to) |
| Logical Operators | And, Or, Not |
| Concatenation  operator | & |
| Assignment  operator | = |
| Increment/decreme  nt operators | i = i + 1, i += 1, i -= 1, i = i - 1 |
| Concatenation  assignment operator | s = s & "new text" |

|  |
| --- |
| **Syntax** |
| Sub procedureName() |
| ' code to execute |
| End Sub |
| **Example** |
| Sub PrintMessage() |
| MsgBox "Hello, World!" |
| End Sub |

**Loops**

**Conditional Statements**

**Working with Data**

**Formatting Cells**

|  |
| --- |
| **If Statement** |
| If Condition Then |
| 'Code to be executed |
| End If |
| **If-Else Statement** |
| If Condition Then |
| 'Code to be executed if condition is true |
| Else |
| 'Code to be executed if condition is false |
| End If |
| **Select Case Statement** |
| Select Case Variable |
| Case Value1 |
| 'Code to be executed if variable equals Value1 |
| Case Value2 |
| 'Code to be executed if variable equals Value2 |
| Case Else |
| 'Code to be executed if variable does not match any value |
| End Select |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Offset | Range("A1").Offset(1, 2).Select (selects cell B2) |
| Hide Rows/Columns | Rows("1:10").Hidden = True or  Columns("A:C").Hidden = True |
| Unhide  Rows/Columns | Rows("1:10").Hidden = False or  Columns("A:C").Hidden = False |
| Read Data from Cell | VariableName = Range("A1").Value |
| Write Data to Cell | Range("A1").Value = VariableName |
| Create Named  Range | Range("A1:C10").Name = "NamedRange" |
| Delete Named Range | Names("NamedRange").Delete |
| Read Data from  Named Range | VariableName = Range("NamedRange").Value |
| Write Data to  Named Range | Range("NamedRange").Value = VariableName |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Font | Range("A1").Font.Bold = True or  Range("A1").Font.Italic = True |
| Background Color | Range("A1").Interior.Color = RGB(255, 255,  0) (yellow) |
| Border | Range("A1").Borders(xlEdgeLeft).LineStyle =  xlContinuous |
| Alignment | Range("A1").HorizontalAlignment = xlCenter |

|  |
| --- |
| **For Loop** |
| **Syntax** |
| For counter\_variable = start\_value to end\_value |
| 'Statements to be executed inside the loop |
| Next counter\_variable |
| **Example** |
| For i = 1 To 10 |
| Cells(i, 1).Value = i |
| Next i |

|  |
| --- |
| **Syntax** |
| Function functionName(argument As DataType) As ReturnType |
| ' code to execute |
| End Function |
| **Example** |
| Function AddNumbers(num1 As Integer, num2 As Integer) As |
| AddNumbers = num1 + num2 |
| End Function |

|  |
| --- |
| **Do-While Loop** |
| **Syntax** |
| Do While condition |
| 'Statements to be executed inside the loop |
| Loop |
| **Example** |
| i = 1 |
| Do While i <= 10 |
| Cells(i, 1).Value = i |
| i = i + 1 |
| Loop |

|  |
| --- |
| **While Loop** |
| **Syntax** |
| While condition |
| 'Statements to be executed inside the loop |
| Wend |
| **Example** |
| i = 1 |
| While i <= 10 |
| Cells(i, 1).Value = i |
| i = i + 1 |
| Wend |

**Manage Files**

**Settings**

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Create Worksheet | Worksheets.Add or  Worksheets.Add(After:=Worksheets("Sheet Name")) |
| Delete Worksheet | Worksheets("SheetName").Delete |
| Select Worksheet | Worksheets("SheetName").Select |
| Set Worksheet | Set WorksheetVariable = Worksheets("SheetName") |
| Activate Worksheet | Worksheets("SheetName").Activate |
| Hide Worksheet | Worksheets("SheetName").Visible = xlSheetHidden |
| Unhide Worksheet | Worksheets("SheetName").Visible =  xlSheetVisible |
| Copy Worksheet | Worksheets("SheetName").Copy Before:=Worksheets("SheetName2") |
| Move Worksheet | Worksheets("SheetName").Move Before:=Worksheets("SheetName2") |
| Rename Worksheet | Worksheets("SheetName").Name = "NewName" |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Activate Workbook 1 | Workbooks("Workbook1.xlsx").Activate |
| Get Active  Workbook Name | MsgBox "The active workbook is: " &  ActiveWorkbook.Name |
| Add Workbook and Set to Variable | Dim newWb As Workbook: Set newWb = Workbooks.Add |
| Open Workbook and Set to Variable | Dim wb As Workbook: Set wb = Workbooks.Open("C:\Folder\Workbook2.xlsx") |
| Close Workbook  without Saving Changes | Workbooks("Workbook1.xlsx").Close SaveChanges:=False |
| Close Workbook with Saving Changes | Workbooks("Workbook1.xlsx").Close SaveChanges:=True |
| Save Workbook | Workbooks("Workbook1.xlsx").Save |
| Save Workbook As | Dim strFileName as String: strFileName = "C:\Folder\NewWorkbookName.xlsx"<br>Work books("Workbook1.xlsx").SaveAs strFileName |
| Protect Workbook with Password | Workbooks("Workbook1.xlsx").Protect Password:="mypassword" |
| Unprotect Workbook with  Password | Workbooks("Workbook1.xlsx").Unprotect Password:="mypassword" |
| Check if Workbook Exists | If Dir("C:\Folder\Workbook1.xlsx") = "" Then<br>MsgBox "The workbook does not  exist."<br>End If |
| Copy File with Custom Message | FileCopy "C:\Folder\OldWorkbook.xlsx", "C:\Folder\NewWorkbook.xlsx"<br>MsgBox "The workbook has been copied successfully." |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Copy a file from one location to another | FileCopy "C:\Data\OldFolder\Report\_2022\_Q1.xlsx", "C:\Data\NewFolder\Report\_2022\_Q1\_Copy  .xlsx" |
| Delete a file | Kill "C:\Data\OldFolder\ObsoleteReport.xlsx" |
| Create a new folder | MkDir "C:\Data\NewFolder" |
| Delete all files from  a folder | Kill "C:\Data\OldFolder\*" |
| Delete a folder | RmDir "C:\Data\OldFolder" |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Turn off Screen  Updating | Application.ScreenUpdating = False |
| Turn on Screen  Updating | Application.ScreenUpdating = True |
| Disable Display  Alerts | Application.DisplayAlerts = False |

**Worksheets**

**Workbooks**

**Error Handling**

**Miscellaneous**

|  |  |
| --- | --- |
| Get the current directory | strPath = CurDir() |
| Get the path of the current workbook | strPath = ThisWorkbook.Path |
| Loop through all files and folders in a folder | strFile = Dir("C:\Data\Reports\2022" & ".") |

|  |  |
| --- | --- |
| Enable Display Alerts | Application.DisplayAlerts = True |
| Disable Events | Application.EnableEvents = False |
| Enable Events | Application.EnableEvents = True |
| Ignore Text Case | Option Compare Text |
| Require Variable  Declaration | Option Explicit |
| Disable Automatic  Calculations | Application.Calculation = xlManual |
| Enable Automatic  Calculations | Application.Calculation = xlAutomatic |
| Disable Background Error Checking | Application.ErrorCheckingOptions.BackgroundC hecking = False |
| Enable Background Error Checking | Application.ErrorCheckingOptions.BackgroundC hecking = True |
| Hide Formula Bar | Application.DisplayFormulaBar = False |
| Show Formula Bar | Application.DisplayFormulaBar = True |
| Enter Full Screen  View | Application.DisplayFullScreen = True |
| Switch to PageBreak Preview | ActiveWindow.View = xlPageBreakPreview |
| Switch to Normal  View | ActiveWindow.View = xlNormalView |
| Hide Scroll Bars | With ActiveWindow .DisplayHorizontalScrollBar  = False .DisplayVerticalScrollBar = False End With |
| Show Scroll Bars | With ActiveWindow .DisplayHorizontalScrollBar  = True .DisplayVerticalScrollBar = True End With |
| Hide Status Bar | Application.DisplayStatusBar = False |
| Show Status Bar | Application.DisplayStatusBar = True |
| Display Custom  Message in Status Bar | Application.StatusBar = “Your Message” |
| Hide Custom  Message in Status Bar | Application.StatusBar = False |
| Hide Workbook Tabs | ActiveWindow.DisplayWorkbookTabs = False |
| Show Workbook  Tabs | ActiveWindow.DisplayWorkbookTabs = True |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Stop execution and show error message | On Error GoTo 0 |
| Ignore error and continue execution | On Error Resume Next |
| Jump to a specific code section | On Error GoTo [Label] |
| Display a custom  error | Err.Raise |
| Display error  description | MsgBox Err.Description |
| Display error number | MsgBox Err.Number |
| Reset error handling | On Error GoTo -1 |

|  |  |
| --- | --- |
| **Task** | **VBA Code** |
| Create Object | Set ObjectName = CreateObject("ProgID") |
| Delete Object | Set ObjectName = Nothing |
| Object Properties | object.property = value |
| Methods | object.method(argument1, argument2) |

**Visual Basic Editor Shortcut Keys**

**VBA Functions for Excel**

|  |  |
| --- | --- |
| **Shortcut Keys** | **Task** |
| Alt + F11 | Switches between the Excel window and the Visual Basic Editor (VBE) window. |
| Alt + F8 | Shows the list of all macros. |
| F5 | Executes the current block of code or resumes execution from the last pause point. |
| Alt + F4 | Exits the Visual Basic Editor and return to the Excel window. |
| Ctrl + Space | Autocompletes a line. |
| Ctrl + Home | Jumps to the beginning of a module. |
| Ctrl + End | Jumps to the end of a module. |
| Ctrl + Right Arrow | Moves the cursor one word to the right. |
| Ctrl + Left Arrow | Moves the cursor one word to the left. |
| End | Moves the cursor to the end of the line. |
| Home | Moves the cursor to the beginning of the  line. |
| Ctrl + A | Selects all texts in the current module. |
| Ctrl + Break | Stops the execution of the code that's  currently running. |
| F8 | Enters the debugging mode and executes one line of code at a time. |
| Tab | Indents the current line. |
| Shift + Tab | Removes the indent from the current line. |

|  |  |
| --- | --- |
| **Category** | **Functions** |
| Date and Time Functions | Date, Time, Now, DateAdd, DateDiff, DatePart, TimeSerial, Weekday, Year, Month, Day, Hour, Minute, Second, DateValue, TimeValue, DateSerial, TimeSerial, Now, TimeSerial |
| Financial Functions | FV, PV, NPV, PMT, RATE, IRR |
| Math and Numeric Functions | Abs, Sqr, Exp, Log, Round, Int, Fix, Mod, Power, Max, Min, Sum, Product, Average, Count, CountA, CountBlank, CountIf, CountIfs, Rand, RandBetween |
| Logical Functions | If, IfError, And, Or, Not, TRUE, FALSE |
| String Functions | Left, Right, Mid, Len, InStr, Replace, Trim, LCase, UCase, Str, StrReverse, Format, Chr, Asc |
| Array Functions | Array, Split, Join, Filter, Sort |
| Lookup and Reference Functions | VLookup, HLookup, Index, Match, Choose, Offset |
| Miscellaneous  Functions | MsgBox, InputBox, Dir, Environ, Shell |

|  |
| --- |
| **VBA Commands for Excel** |

|  |  |
| --- | --- |
| **Category** | **Commands** |
| Range and Cell Selection | Range(), Cells(), ActiveCell, Offset() |
| Clipboard | Copy(), Paste() |
| Worksheets and Sheets | Worksheets(), ActiveSheet, Sheets(), Protect, Unprotect |

|  |  |
| --- | --- |
| Sleep | Application.Wait(Now + TimeValue("0:00:10")) (wait for 10 seconds) |
| SendKeys | Application.SendKeys "TextToType" (types the specified text) |
| Debugging Mode | Debug.Print "Message Text" |

|  |  |
| --- | --- |
| Set User Name | Application.UserName = “MyName” |
| Set Application Caption | Application.Caption = “MyApp” |
| Set Zoom Level | ActiveWindow.Zoom = 90 or any desired value |

|  |  |
| --- | --- |
| Shift + F10 | Displays the active window's context menu, similar to right-clicking. |
| Ctrl + F | Finds text in the current module. |
| Ctrl + H | Replaces text in the current module. |
| Ctrl + Y | Redoes the last action that was undone. |
| Ctrl + Z | Undoes the last action. |

|  |  |
| --- | --- |
| Workbooks | Workbooks(), ActiveWorkbook, SaveAs(), Open(), Save, Close, ThisWorkbook |
| PivotTables | PivotTables(), PivotCharts(), PivotFields(), RefreshTable(), PivotSelect(), PivotLayout(), PivotTableWizard() |
| Charts | ActiveChart, ChartObjects(), SeriesCollection(), ChartTitle, ChartType, SetElement |
| Data Validation | DataValidation, ValidationType, InputTitle, InputMessage, ErrorTitle, ErrorMessage, ShowInput, ShowError, InCellDropdown, IgnoreBlank, Formula1, Formula2, Add, Modify, Delete |
| Miscellaneous | MsgBox, InputBox, CreateObject, Shell, Timer, Environ, ClearContents() |

|  |
| --- |
| **Useful VBA Resources** |
| 40 Advanced Useful VBA Codes for Excel [Free PDF Download] |
| [https://excel](https://excelgraduate.com/advanced-useful-vba-codes-for-excel/)graduate.com/advanced-useful-vba-codes[-f](https://excelgraduate.com/advanced-useful-vba-codes-for-excel/)or-[ex](https://excelgraduate.com/advanced-useful-vba-codes-for-excel/)cel/ |
| A Complete Guideline to Insert and Run VBA Code in Excel |
| [https://excel](https://excelgraduate.com/insert-and-run-vba-code-in-excel/)graduate.com/insert-and-run-vba-code-in[-e](https://excelgraduate.com/insert-and-run-vba-code-in-excel/)x[ce](https://excelgraduate.com/insert-and-run-vba-code-in-excel/)l/ |
| A List of Excel VBA Functions [Free PDF Download] |
| [https://excel](https://excelgraduate.com/list-of-excel-vba-functions/)graduate.com/list-of-excel-vba-functions/ |
| A List of Excel VBA Commands [Free PDF Download] |
| [https://excel](https://excelgraduate.com/list-of-excel-vba-commands/)graduate.com/list-of-excel-vba-command[s/](https://excelgraduate.com/list-of-excel-vba-commands/) |
| A List of 50 VBA Shortcut Keys for Excel [Free PDF Download] |

[https://excel](https://excelgraduate.com/vba-shortcut-keys-for-excel/)graduate.com/vba-shortcut-keys-for-excel[/](https://excelgraduate.com/vba-shortcut-keys-for-excel/)

***Help:*** [***https://excelgraduate.com/vba-cheat-shee***](https://excelgraduate.com/vba-cheat-sheet-for-excel/)***t-for-***[***ex***](https://excelgraduate.com/vba-cheat-sheet-for-excel/)***cel/***



**Excelgraduate**

*© 2023 excelgraduate.com | All Rights Reserved.*