EXCEL ASSIGNMENT 19

1. ***What are the data types used in VBA?***

***Ans*** There are quite a few VBA data types, but for the general purposes of financial modeling not all of them are used.

Below is a list of common VBA variables (known as data types) used in macros and their purposes:

Integer: Used to store number values that won’t take on decimal form.

Single: Used to store number values that may take on decimal form. Can also contain integers.

Double: A longer form of the single variable. Takes up more space, but needed for larger numbers.

Date: Stores date values.

String: Stores text. Can contain numbers, but will store them as a text (calculations cannot be performed on numbers stored as a string)

Boolean: Used to store binary results (True/False, 1/0)

***2. What are variables and how do you declare them in VBA? What***

***happens if you don’t declare a variable?***

***Ans.*** In VBA, a Variable is like a storage box that is itself stored in your system but it can store a value in it for you and you can use that value in your code and can change that value if you want (as the name suggests, “VARIABLE” is something whose value is not fixed).

To declare a variable you need to follow a simple procedure:

1. Use the keyword “Dim” in the starting.
2. Specify a name for the variable.
3. Use the keyword “As” after the name.
4. Specify the “Data Type” for the variable according to the value you want to assign to it.

Here we have used the name “startDate” and specified the data type “Date”. After declaring a variable you can assign a value to it.

Dim startDate As Date

startDate = “11/10/2018”

Now, whenever you use the start date in your code Excel will use the date you have assigned to it.

When variables are not declared By default when an undeclared variable is called upon, VBA assigns it as the Variant, which is the largest of the data types stored as 16 bytes. This can accumulate and take up a lot of memory and cause the execution to become sluggish. ***3. What is a range object in VBA? What is a worksheet object?***

***Ans.*** Excel VBA Range Object

Range is a property in VBA that helps specify a particular cell, a range of cells, a row, a column, or a three-dimensional range. In the context of the Excel worksheet, the VBA range object includes a single cell or multiple cells spread across various rows and columns.

For example, the range property in VBA is used to refer to specific rows or columns while writing the code. The code “Range(“A1:A5”).Value=2” returns the number 2 in the range A1:A5.

In VBA, macros are recorded and executed to automate the Excel tasks. This helps perform the repetitive processes in a faster and more accurate way. For running the macros, VBA identifies the cells on which the called tasks are to be performed. It is here that the range object in VBA comes in use.

The VBA range property is similar to the worksheet property and has several applications

**Worksheet object:-**

The **Worksheet** object is a member of the [**Worksheets**](https://learn.microsoft.com/en-us/office/vba/api/excel.worksheets) collection. The **Worksheets** collection contains all the **Worksheet** objects in a workbook.

The **Worksheet** object is also a member of the [**Sheets**](https://learn.microsoft.com/en-us/office/vba/api/excel.sheets) collection. The **Sheets** collection contains all the sheets in the workbook (both chart sheets and worksheets).

**Example**

Use [**Worksheets**](https://learn.microsoft.com/en-us/office/vba/api/excel.workbook.worksheets) (*index*), where *index* is the worksheet index number or name, to return a single **Worksheet** object. The following example hides worksheet one in the active workbook.

VBCopy

Worksheets(1).Visible = False

The worksheet index number denotes the position of the worksheet on the workbook's tab bar. Worksheets(1) is the first (leftmost) worksheet in the workbook, and Worksheets(Worksheets.Count) is the last one. All worksheets are included in the index count, even if they are hidden.

The worksheet name is shown on the tab for the worksheet. Use the [**Name**](https://learn.microsoft.com/en-us/office/vba/api/excel.worksheet.name) property to set or return the worksheet name. The following example protects the scenarios on Sheet1.

VBCopy

Dim strPassword As String

strPassword = InputBox ("Enter the password for the worksheet")

Worksheets("Sheet1").Protect password:=strPassword, scenarios:=True

When a worksheet is the active sheet, you can use the **[ActiveSheet](https://learn.microsoft.com/en-us/office/vba/api/excel.workbook.activesheet)** property to refer to it. The following example uses the [**Activate**](https://learn.microsoft.com/en-us/office/vba/api/excel.worksheet.activate(method)) method to activate Sheet1, sets the page orientation to landscape mode, and then prints the worksheet.

VBCopy

Worksheets("Sheet1").Activate

ActiveSheet.PageSetup.Orientation = xlLandscape

ActiveSheet.PrintOut

1. ***What is the difference between worksheet and sheet in excel?***

***Ans.* The difference between sheets and worksheet are as follows-**

1. **Sheets** can be created using a computer tool. It is mainly created to organize and tabulate the data. **Worksheets** are mainly created to assign students to some questions related to education.

2. The data in **sheets** can be organized using excel which is a type of computer software. **Worksheets** can be printed, typed, and also in some cases handwritten.

3. The data in **sheets**can be changed and manipulated according to the need of the client but worksheets are a group of questions and the person who is assigned with the **worksheet** cannot adjust it.

1. ***What is the difference between A1 reference style and R1C1 Reference style? What are the advantages and disadvantages of using R1C1 reference style?***

# ***Ans.*** Excel – R1C1 Reference Style vs. A1

[**APR 22**](https://excelmate.wordpress.com/2013/04/22/excel-r1c1-reference-style-vs-a1/) If you have ever strayed into the Options screens of Excel you may have noticed something called R1C1 reference style. You may have even tried it and when you saw that all the columns had changed from letters to numbers, panicked and switched it back.

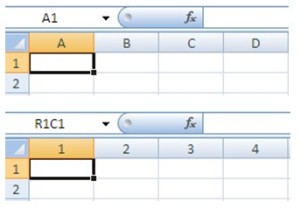
However, the R1C1 style has some advantages over the standard A1 style, but it does take a bit of getting used to!

So is this R1C1 style new? Well in fact not at all. This cell reference style dates back to 1982 and the introduction of Multiplan as a rival to VisiCalc and Lotus 1-2-3. Multiplan was developed by Microsoft for Apple Macintosh and used this cell referencing style instead of what was to become the standard in spreadsheets – A1 style.

Over the next few years, Lotus 1-2-3 became “the” spreadsheet package. In order for Microsoft to steal Lotus users who were used to the A1 style they added the A1 style to Excel so that people who migrated across would find it familiar. Excel eventually overtook Lotus in 1993 as the industry standard for spreadsheets. The rest as they say is history.

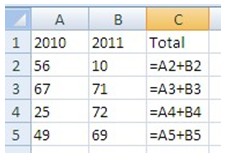
But enough of the history lesson…how does R1C1 differ to A1?

The most obvious difference is the column references. Instead of letters you get numbers.

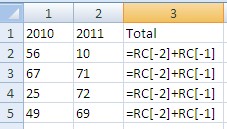
[](https://excelmate.files.wordpress.com/2013/04/screenhunter_67-apr-22-13-42.jpg)

The real difference comes when you want to write formulas. Let’s take a simple example adding two columns together. I am displaying the formulas so the differences are clearer.

First, the familiar A1 style:

[](https://excelmate.files.wordpress.com/2013/04/screenhunter_67-apr-22-13-43.jpg)

And now for R1C1….

[](https://excelmate.files.wordpress.com/2013/04/screenhunter_58-mar-18-15-05.jpg)

At first this looks quite horrific and ten times more complicated that the A1 style. Before I break down how it works, what do you notice between the two styles (other than the obvious one)?

Each formula is different in A1 style: A2+B2, A3+B3…etc.

Whereas using R1C1, they are all identical. So this potentially means that wherever you write a formula in that column it will be same, no need to think about which row or column you are in. This is particularly helpful when you are writing VBA code.

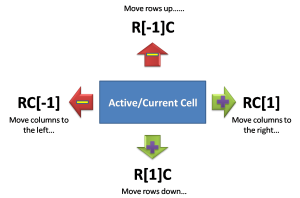
So how does it work?

Any numbers in square brackets refer to relative distance from the current cell. Unlike A1 which refers to columns followed by row number, R1C1 does the opposite: rows followed by columns (which does take some getting used to).

Positive numbers will refer to cells below and/or across to the right.

Negative numbers will refer to cells above and/or to the left.

For example R[2]C[3] is a cell 2 rows down and 3 columns to the right. R[-1]C[-4] is a cell 1 row up and 4 columns to the left. If no number is shown in brackets then you are referring to the same row or column i.e. R[3]C will be a cell 3 rows below the current cell in the **SAME** column.

[](https://excelmate.files.wordpress.com/2013/04/r1c1-ref.png)

So once you have the basics, they are actually not that bad. Note however, that you cannot use an A1 formula when displaying R1C1 style and vice-versa. Whichever system you use, Excel ‘translates’ the styles should you switch between them at any point.

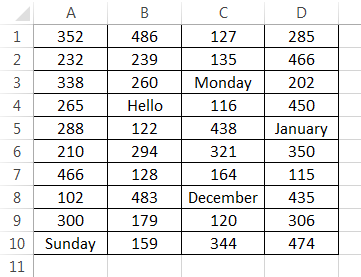
The other difference between the two styles is absolute referencing. In the A1 style I need to add $ symbols all over the place. Granted I can use F4 to put them in place for me but it still needs to be done. In R1C1, there are no $. If I write R3C4 I am referring to $D$3. So if there are no brackets, it’s an absolute reference. This makes partial absolute references easier to enter too.

The R1C1 style definitely does have some neat advantages over A1, but we are so used to the A1 style that moving away from it is an alien concept. But if you ever have a chance, try playing with R1C1, especially if you progress to VBA, and I can guarantee that writing formulas will become a lot easier…the pain is only short-lived but it is worth the effort.

1. ***When is offset statement used for in VBA? Let’s suppose your current highlight cell is A1 in the below table. Using OFFSET statement, write a VBA code to highlight the cell with “Hello” written in it.***

***Ans. VBA Offset****function one may use to move or refer to a reference skipping a particular number of rows and columns. The arguments for this function in VBA are the same as those in the worksheet.*

For example, assume you have a data set like the one below.



Now from cell A1, you want to move down four cells and select that 5th cell, the A5 cell.

Similarly, if you want to move two rows down from the A1 cell and two columns to the right, select that cell, i.e., the C2 cell.

In these cases, the OFFSET function is very helpful. Especially in VBA OFFSET, the function is just phenomenal.