**ANALYSIS**

**Background**

We will help Steve today. He wants to analyze a higher number of stock.

We will make more research for him and try to expand to dataset to include the entire stock market over the last few years. Our code works well for a dozen stocks, it might not work as well as thousand stocks. And if it does, it may take long time.

We will edit, or refactor, the Module 2 solution code to loop through all the data one time in order to collect the same information that we did in this module.

**Elapsed Time For 2017 and 2018 – Module 2**

Graphical user interface, application, table, Excel

Description automatically generatedGraphical user interface, application, table, Excel

Description automatically generated

**Refactor VBA Code and Measure Performance**

We tried our code more efficient and faster. We looped through the data and collected all information. Our refactored code is now faster than before. See the Original code and refactored code below.

**Refactored Code Performances**

**Graphical user interface, application, table, Excel

Description automatically generated**

**Graphical user interface, application, table, Excel

Description automatically generated**

**Refactored**

Sub AllStocksAnalysisRefactored()

    Dim startTime As Single

    Dim endTime  As Single

    yearValue = InputBox("What year would you like to run the analysis on?")

    startTime = Timer

    'Format the output sheet on All Stocks Analysis worksheet

    Worksheets("All Stocks Analysis").Activate

    Range("A1").Value = "All Stocks (" + yearValue + ")"

    'Create a header row

    Cells(3, 1).Value = "Ticker"

    Cells(3, 2).Value = "Total Daily Volume"

    Cells(3, 3).Value = "Return"

    'Initialize array of all tickers

    Dim tickers(12) As String

    tickers(0) = "AY"

    tickers(1) = "CSIQ"

    tickers(2) = "DQ"

    tickers(3) = "ENPH"

    tickers(4) = "FSLR"

    tickers(5) = "HASI"

    tickers(6) = "JKS"

    tickers(7) = "RUN"

    tickers(8) = "SEDG"

    tickers(9) = "SPWR"

    tickers(10) = "TERP"

    tickers(11) = "VSLR"

    'Activate data worksheet

    Worksheets(yearValue).Activate

    'Get the number of rows to loop over

    RowCount = Cells(Rows.Count, "A").End(xlUp).Row

    '1a) Create a ticker Index

    Dim tickerIndex As Integer

    tickerIndex = 0

    '1b) Create three output arrays

    Dim Volumes(12) As Long

    Dim StartingPrice(12) As Single

    Dim EndingPrice(12) As Single

    ''2a) Create a for loop to initialize the tickerVolumes to zero.

    For idx = 0 To 11

        Volumes(idx) = 0

    Next idx

     ''2b) Loop over all the rows in the spreadsheet.

    For i = 2 To RowCount

        '3a) Increase volume for current ticker

        Volumes(tickerIndex) = Volumes(tickerIndex) + Cells(i, 8).Value

        '3b) Check if the current row is the first row with the selected tickerIndex.

        If Cells(i - 1, 1).Value <> tickers(tickerIndex) Then

            StartingPrice(tickerIndex) = Cells(i, 6).Value

        End If

        '3c) check if the current row is the last row with the selected ticke

        If Cells(i + 1, 1).Value <> tickers(tickerIndex) Then

            EndingPrice(tickerIndex) = Cells(i, 6).Value

            '3d Increase the tickerIndex.

            tickerIndex = tickerIndex + 1

        End If

    Next i

        '4) Loop through your arrays to output the Ticker, Total Daily Volume, and Return.

        '6) Output data for current ticker

        Worksheets("All Stocks Analysis").Activate

    For j = 0 To 11

        Cells(4 + j, 1).Value = tickers(j)

        Cells(4 + j, 2).Value = Volumes(j)

        Cells(4 + j, 3).Value = EndingPrice(j) / StartingPrice(j) - 1

    Next j

    'Formatting

    Range("A3:C3").Font.FontStyle = "Bold"

    Range("A3:C3").Borders(xlEdgeBottom).LineStyle = xlContinuous

    Range("B4:B15").NumberFormat = "#,##0"

    Range("C4:C15").NumberFormat = "0.0%"

    Columns("B").AutoFit

    dataRowStart = 4

    dataRowEnd = 15

    For i = dataRowStart To dataRowEnd

        If Cells(i, 3) > 0 Then

            Cells(i, 3).Interior.Color = vbGreen

        Else

            Cells(i, 3).Interior.Color = vbRed

        End If

    Next i

   endTime = Timer

   MsgBox "This code ran in " & (endTime - startTime) & " seconds for the year " & (yearValue)

End Sub

**Original**

Sub AllStocksAnalysis()

    Dim startTime As Single

    Dim endTime  As Single

    yearValue = InputBox("What year would you like to run the analysis on?")

    startTime = Timer

    'Format the output sheet on All Stocks Analysis worksheet

    Worksheets("All Stocks Analysis").Activate

    Range("A1").Value = "All Stocks (" + yearValue + ")"

    'Create a header row

    Cells(3, 1).Value = "Ticker"

    Cells(3, 2).Value = "Total Daily Volume"

    Cells(3, 3).Value = "Return"

    'Initialize array of all tickers

    Dim tickers(11) As String

    tickers(0) = "AY"

    tickers(1) = "CSIQ"

    tickers(2) = "DQ"

    tickers(3) = "ENPH"

    tickers(4) = "FSLR"

    tickers(5) = "HASI"

    tickers(6) = "JKS"

    tickers(7) = "RUN"

    tickers(8) = "SEDG"

    tickers(9) = "SPWR"

    tickers(10) = "TERP"

    tickers(11) = "VSLR"

    'Activate data worksheet

    Worksheets(yearValue).Activate

    'Get the number of rows to loop over

    RowCount = Cells(Rows.Count, "A").End(xlUp).Row

    '1a) Create a ticker Index

    Dim tickerIndex As Integer

    tickerIndex = 0

    '1b) Create three output arrays

    Dim Volumes(11) As Long

    Dim StartingPrices As Single

    Dim EndingPrices As Single

    ''2a) Create a for loop to initialize the tickerVolumes to zero.

    For idx = 0 To 11

        ticker = tickers(idx)

        totalVolume = 0

    ''2b) Loop over all the rows in the spreadsheet.

    For i = 2 To RowCount

    Worksheets(yearValue).Activate

    '3a) Increase volume for current ticker

        If Cells(i, 1).Value = ticker Then

            totalVolume = totalVolume + Cells(i, 8).Value

            Volumes(idx) = totalVolume

        End If

    '3b) Check if the current row is the first row with the selected tickerIndex.

    'If  Then

        If Cells(i - 1, 1).Value <> ticker And Cells(i, 1).Value = ticker Then

            StartingPrices = Cells(i, 6).Value

        End If

    'End If

    '3c) check if the current row is the last row with the selected ticker

    'If the next rowÕs ticker doesnÕt match, increase the tickerIndex.

    'If  Then

        If Cells(i + 1, 1).Value <> ticker And Cells(i, 1).Value = ticker Then

            EndingPrices = Cells(i, 6).Value

    '3d Increase the tickerIndex.

           tickerIndex = tickerIndex + 1

        End If

    'End If

    Next i

    '4) Loop through your arrays to output the Ticker, Total Daily Volume, and Return.

       '6) Output data for current ticker

       Worksheets("All Stocks Analysis").Activate

       Cells(4 + idx, 1).Value = ticker

       Cells(4 + idx, 2).Value = totalVolume

       Cells(4 + idx, 3).Value = EndingPrices / StartingPrices - 1

    Next idx

    'Formatting

    Range("A3:C3").Font.FontStyle = "Bold"

    Range("A3:C3").Borders(xlEdgeBottom).LineStyle = xlContinuous

    Range("B4:B15").NumberFormat = "#,##0"

    Range("C4:C15").NumberFormat = "0.0%"

    Columns("B").AutoFit

    dataRowStart = 4

    dataRowEnd = 15

    For i = dataRowStart To dataRowEnd

        If Cells(i, 3) > 0 Then

            Cells(i, 3).Interior.Color = vbGreen

        Else

            Cells(i, 3).Interior.Color = vbRed

        End If

    Next i

   endTime = Timer

   MsgBox "This code ran in " & (endTime - startTime) & " seconds for the year " & (yearValue)

End Sub

**Result**

**Advantage of refactoring code**

It is the most obvious advantage of refactoring code it makes it more efficient. It reduced the execution time. It helps to analyzing thousands of row of data

Disadvantage of refactoring code

If you don’t save your original data, it is huge risk the your errors may destroy an already working code. So saving original code is highly recommended.