Author: John Weiss

Date: 22/02/2023

Hazell Bros JobPac Excel Macro

Logo

Description automatically generated

1. Introduction

Hazell Bros have requested two Excel Visual Basic Macros. The macro’s purpose is to automate the payroll process. Hazell Bros’ current structure involves individuals manually adding data to an excel file. After the data is manually updated to fit the upload structure JobPac structure it gets uploaded to the JobPac website. The first macro is for the wages pivot table and the second macro is for the salary pivot table.

The requested excel macros should format the payroll file of employees and display them in a suitable format so that the data can be uploaded to the JobPac website. The purpose of the macros is to eliminate the repetitive task of manually formatting payroll files, so they are acceptable to be uploaded. The salary pivot macro will be similar to the wages pivot macro except different cells will be referenced and not as much data is filtered.

Hazell Bros Excel macro will be developed in Excel through the programming language Visual Basic Analysis (VBA). VBA is developed by Microsoft and is used as a programming language for Office applications. The team developing the macros includes Sean Thomas of Hazell Bros and John Weiss.

1. Project Description

See (Figure 1) for a diagram of the current structure of the payroll process.

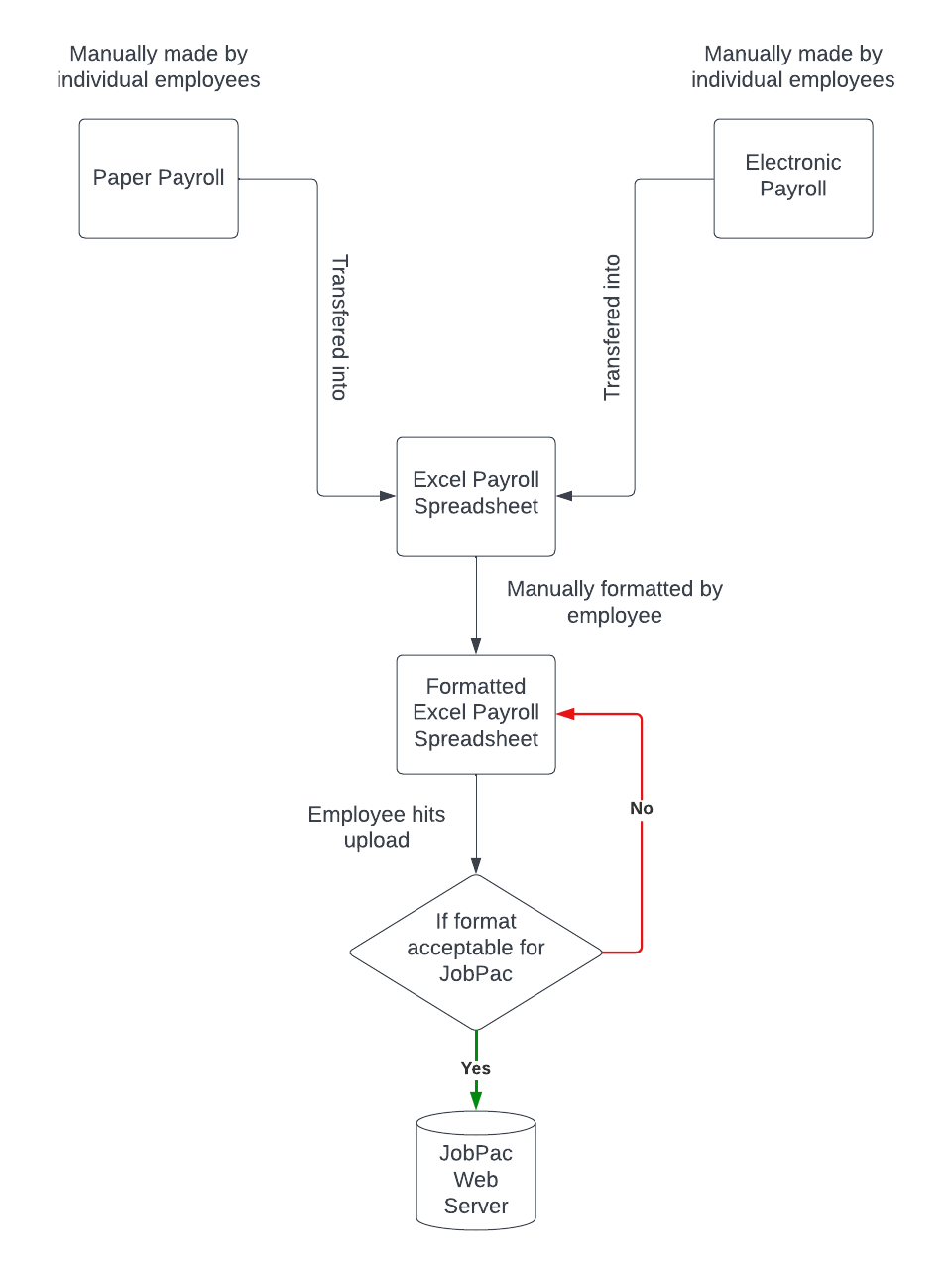


Figure 1. Shows the current structure of the payroll process without the new macro in place.

Current Process: Employees at Hazell Bros currently fill out their timesheets with the choice of doing it digitally or on paper. This data is then collected and stored in one big Excel Spreadsheet file that holds the records of the employees’ timesheets (Excel payroll sheet).

Another Hazel Bros employee formats the data in the Excel payroll sheet. Formatting of the Excel payroll sheet is a requirement so that the Excel payroll sheet can get uploaded to the JobPac web server without being rejected for formatting reasons. If the information in excel Spreadsheet is not formatted correctly to match the upload process it will fail and an employer will have to manually fix the formatting problem.

The two macros to be developed will be displayed as a button one will be displayed on the wages pivot updated worksheet and the other will be displayed on the salary pivot worksheet. Once the user clicks the macro button the macro will run a list of VBA commands that will automatically format the Excel data into a new worksheet. The newly formatted data will be displayed on a worksheet called salary pivot output and wages pivot output which is displayed in the same Excel project. The salary pivot output and wages pivot output data can then be uploaded to the JobPac website.

Diagram

Description automatically generated

Figure 2. Shows the payroll process with the new macro in place.

1. Requirements

Macro Requirements Table Key:

Requirement Code: NTH = Nice To Have

MH = Must Have

Priority: is a number 1-10, where 10 has the highest priority and 1 the lowest.

Requirement connection shows if a requirement is connected to another requirement. If two requirements are connected, they will hold the same first numerical value. The number after the decimal place will show the order that the requirements are connected in, e.g. (2.1 where 1 shows the first requirement of that connection and 2.2 would be the third in the chain as we are starting from 0.)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MACRO REQUIREMENTS TABLE |  |  |  |  |
| REQUIREMENT NAME | **REQUIREMENT CODE** | **REQUIREMENT DESCRIPTION** | **REQUIREMENT CONNECTION** | **PRIORITY** |
| Macro coded in Excel. | MH | The macro must be written in Excel in the VBA programming language to make sure it's compatible with Excel | 1.0 | 10 |
| Macro displayed as a button. | NTH | Macro is displayed as a button. | 2.0 | 6 |
| Macro copies data to a new worksheet. | MH | Macro will take the information from the Excel payroll sheet and copy it to a new worksheet named output | 3.0 | 10 |
| Only relevant data is copied. | MH | Macro will take only the necessary rows and columns from the Excel payroll sheet needed for uploading. | 3.1 | 10 |
| Macro output heading. | NTH | Macro will highlight the first row and make the headers bold. | 4.0 | 4 |
| Macro deletes any old data. | NTH | Macro will delete any data that may still be in the output worksheet before pasting new data. | 5.0 | 7 |
| Dynamic macro copy. | MH | Macro must copy the number of rows and columns dynamically to the new worksheet. | 3.2 | 10 |
| Scanning output data. | NTH | Macro will loop through all the data in the output section. | 6.0 | 8 |
| Looking for (blank). | NTH | While looping through the data macro will look for (blank). | 6.1 | 7 |
| Compare and delete rows. | NTH | Once (blank) has been found we check that the hours in that row are also 0 for safety reasons before we delete that row. | 6.2 | 7 |
| Error highlighting. | NTH | If the row has (blank) and has hours greater than 0 we have found an issue and will highlight that row in red. | 6.3 | 5 |
| Identical comments. | MH | Comments written at the top header section should cascade down until the last row. The comment is the same for every row. | 7.0 | 10 |
| No absolute referencing. | MH | Macro will not need absolute referencing. | 8.0 | 9 |
| Eliminate screen flickering. | NTH | Macro will not have the display flickering when running | 9.0 | 3 |

4. Problem & Solution

Through this inconvenient process, a Hazell Bros employee must manually format the Excel payroll sheet, this is a timely process costing the company time and money. The manual process is also prone to human errors meaning rows may be falsely edited, deleted, or added to the Excel payroll sheet causing incorrect or missing data to not be uploaded potentially.  
  
The solution involves automating the process of formatting the Excel payroll sheet, so it is acceptable to upload without any errors or missing data. Taking away the process of manually formatting the Excel payroll sheet will eliminate all those previous problems mentioned and reduce the amount of time required to upload the Excel pay sheet to the JobPac web server. The macros will be available through the Excel payroll sheet.

5. Requirements 2.0

These requirements were initially overseen and found during development. The same principal and logic apply to this requirements table as the Macro Requirements Table.

Macro Requirements Table Key:

Requirement Code: NTH = Nice To Have

MH = Must Have

Priority: is a number 1-10, where 10 has the highest priority and 1 the lowest.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MACRO REQUIREMENTS TABLE 2.0 |  |  |  |  |
| REQUIREMENT NAME | **REQUIREMENT CODE** | **REQUIREMENT DESCRIPTION** | **REQUIREMENT CONNECTION** | **PRIORITY** |
| KM Cost Code column needs to be added. | MH | Kilometre Cost Code was not in original Excel file and needed to be added manually next to the KM Units Column. | 1.0 | 10 |
| KM Cost Code Data. | MH | KM Cost Code data is copied over from the first Cost Code Column (Column AC). | 1.1 | 10 |
| Deleting (blank) in KM Cost Code. | MH | KM Cost Code must delete all (blank)’s same as the original cost code gets filtered. | 1.2 | 8 |
| Finding the zeros. | MH | Macro will filter out each hour row and search for the number 0. | 2.0 | 10 |
| Deleting the zeros. | MH | Macro will delete the row if the hours column is 0. | 2.1 | 10 |
| Macro colours data based on categories. | NTH | Data that belongs together will be the same colour. | 3.0 | 4 |
| Cost Code for Site & Asbestos Allowance, and Crib Time | MH | Cost Code for all three categories gets copied over from the previous cost code data given in column (AT) OA2 Cost Code. | 4.0 | 8 |
| Dealing with missing cost codes. | NTH | If a cost code is not given and missing, we need to fill it. To fill the missing cost code, we look for the next closest cost code with a number and then take that number and copy it into our missing cost code. | 5.0 | 6 |

7.0 Technical Coding Overview of Macro

This section of the report will cover the source code for the VBA Macro created for Hazell Brothers Wages Pivot Table. The Salary Pivot Table VBA Macro will not be covered as it is the exact same principle as the Wages Pivot Table Macro except with less data. Please download/open the wages\_macro\_code.txt file to follow along with the code and information provided below. All code snippets from the wages\_macro\_code.txt file will be in italic. Duplicated code will not be explained twice.



1. While the macro is running, we want to prevent screen flickering which is happening due to the macro changes worksheets back and forth rapidly. To prevent these two lines of code are given. One at the very start of the macro the first line of code and one at the very end of the macro the last line before end sub.

First line of code: *Application.ScreenUpdating = False*

Last line of code: *Application.ScreenUpdating = True*

1. Macro must delete old information in the output sheet before pasting new data.

Three lines of code are needed for this to clear the data, formats, and contents.

*Sheets("Wages Pivot Output").Cells.Clear*

*Sheets("Wages Pivot Output").Cells.ClearFormats*

*Sheets("Wages Pivot Output").Cells.ClearContents*

1. Wages Pivot updated file showing header from column X to AE.

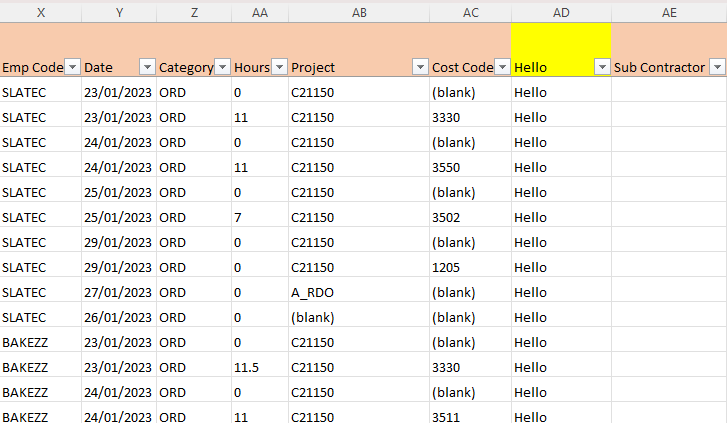


Figure Wages Pivot updated excel file

Macro pastes all data dynamically in column X1 to AE. Paste Special is used to prevent references from copying over and this causes the date column to need formatting back to dates. Finally, the header gets highlighted.

*'Pasting all the values dynamically*

*Range("X1:AE" & Rows.Count).Select*

*Selection.Copy*

*Sheets("Wages Pivot Output").Select*

*Range("A1").Select*

*'Paste special so we just paste the values to overcome #REF!*

*Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks \_*

*:=False, Transpose:=False*

*'Formatting date column*

*Range("B:B").NumberFormat = "d/mm/yyyy;@"*

*' Header\_Highlight Macro*

*' Highlight the header of the Wages Pivot Output file*

*Range("A1:H1").Select*

*Selection.Font.Bold = True*

*With Selection.Interior*

*.Pattern = xlSolid*

*.PatternColorIndex = xlAutomatic*

*.ThemeColor = xlThemeColorAccent4*

*.TintAndShade = 0*

*.PatternTintAndShade = 0*

*End With*

1. Since this data seen in (Figure 3) has (blank) values of data in it we can delete the (blank) values only if the hours inside column AA are also 0 though, helping from deleting employees’ hours.

Filtered output of Wages Pivot updated.

A screenshot of a spreadsheet

Description automatically generated with medium confidence

Figure Wages Pivot Output file showcasing the filtered output after deleting (blanks) & 0’s.

The Macro for matching blanks and deleting rows.

*' Matching Macro*

*'Looking for (blank) and highlighting row red if found*

*Range("F2").Select ' Select cell F2, \*first line of data\*.*

*Do Until IsEmpty(ActiveCell) ' Set Do loop to stop when an empty cell is reached.*

*If ActiveCell.Value = "(blank)" Then 'If the cell value matches (blank) we found a match and highlight that row red up until column H*

*With Range("A" & ActiveCell.row & ":H" & ActiveCell.row).Interior*

*.Pattern = xlSolid*

*.PatternColorIndex = xlAutomatic*

*.Color = 255*

*.TintAndShade = 0*

*.PatternTintAndShade = 0*

*End With*

*End If*

*' Step down 1 row from present location.*

*ActiveCell.Offset(1, 0).Select*

*Loop*

*' Delete all red highlighted rows where column D is 0*

*Dim i As Long*

*For i = Sheets("Wages Pivot Output").UsedRange.Rows.Count To 2 Step -1 'loop through all rows in reverse order*

*If Sheets("Wages Pivot Output").Range("H" & i).Interior.Color = 255 And Sheets("Wages Pivot Output").Range("D" & i).Value = 0 Then 'check if row is highlighted in red and value in column D is 0*

*Sheets("Wages Pivot Output").Rows(i).Delete 'delete the row*

*End If*

*Next i*

1 The macro starts by selecting the cost code column located in F2.

2 A do until loop is started that will check each cell in column F2 for the value (blank).

3 If the value (blank) is found the macro will highlight that row red from column A to H.

4 Next the macro goes down to next empty cell in column F.

5 Next we create the variable i as a counter and assign it the type long so that it can be a decimal number.

6 Next we go through all the rows in reverse order.

7 While looking through the rows if we find a row highlighted red, we check that the hours column has the value 0.

8 If the hours column is red and has the value 0, we can safely delete that row.

9 If the hours column does not have the value 0, we leave it highlighted red indicating an error has been found.

1. This part of the macro focuses on copying the Travel Time data. The code is identical for copying the kilometres, travel allowances, OA1 & OA2 only different columns in the Range() field will differ.

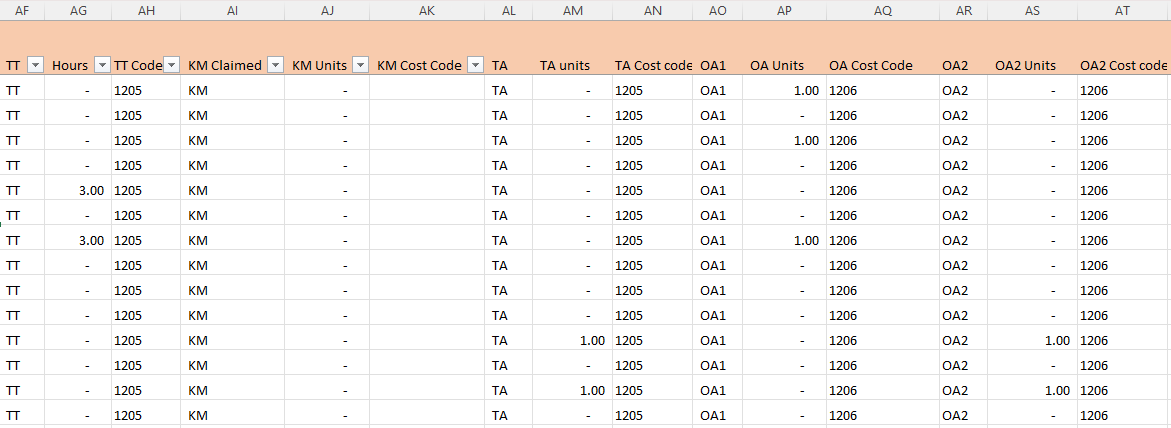


Figure Wages Pivot updated excel file showcasing extended data including the TT columns for reference.

1 The code starts off by going back to the Wages Pivot updated worksheet.

2 The travel time columns of data are selected by using the excel shortcut ctrl+shift+down to select all the data in the selected range.

3 The data is copied, and the macro goes back to the output worksheet.

4 The next available cell in column C1 is selected by going to the bottom of the data in that column and going down one more cell to the next empty cell.

5 Here the data is pasted special again, so the values only get pasted and not the references.

6 We then highlight all the pasted data to a certain colour to show it is in a group.

*'Part two Travel Time Macro*

*'Comments can be found on Salary Pivot Macro same code as that just different columns*

*Sheets("Wages Pivot updated").Select*

*Range("AF2:AG2").Select*

*Range(Selection, Selection.End(xlDown)).Select*

*Selection.Copy*

*Sheets("Wages Pivot Output").Select*

*Range("C1").Select*

*Selection.End(xlDown).Offset(1).Select*

*Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks \_*

*:=False, Transpose:=False*

*Application.CutCopyMode = False*

*With Selection.Interior*

*.Pattern = xlSolid*

*.PatternColorIndex = xlAutomatic*

*.Color = 65535*

*.TintAndShade = 0*

*.PatternTintAndShade = 0*

*End With*

1. The next section in the macro is copying the cost code over from the Wages Pivot updated to the Wages Pivot Output worksheet. This code is identical for all the rest of the sections. This section is referring to the Travel Time code.

A screenshot of a computer

Description automatically generated with low confidence

Figure Wages Pivot updated excel file TA columns.

1 Macro selects the Wages Pivot updated worksheet.

2 Macro selects the cost code column for current category in this case Travel Time cost code.

3 Macro copies the data and goes back to the Wages Pivot Output worksheet.

4 Macro selects cost code header and jumps to the last cell with data.

5 Macro offsets and jumps down 1 cell to the next empty cell.

6 Macro pastes the data via paste special so only values are pasted.

7 Macro highlights column same colour as its predecessor.

*Sheets("Wages Pivot updated").Select*

*Range("AH2").Select*

*Range(Selection, Selection.End(xlDown)).Select*

*Selection.Copy*

*Sheets("Wages Pivot Output").Select*

*ActiveWindow.SmallScroll Down:=-123*

*Range("F1").Select*

*'Going to bottom via excel shortcut ctrl+downArrow*

*'Offsetting by one to go to next empty cell*

*Selection.End(xlDown).Offset(1).Select*

*'Paste special so we just paste the values to overcome #REF!*

*Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks \_*

*:=False, Transpose:=False*

*Application.CutCopyMode = False*

*With Selection.Interior*

*.Pattern = xlSolid*

*.PatternColorIndex = xlAutomatic*

*.Color = 65535*

*.TintAndShade = 0*

*.PatternTintAndShade = 0*

*End With*

1. The next section in the macro is copying the Emp Code & Date over from the Wages Pivot updated to the Wages Pivot Output worksheet. This code is identical for all the rest of the sections.

A picture containing text, screenshot, font, number

Description automatically generated

Figure Wages Pivot updated excel file Emp Code & Date column.

1 Macro selects the Wages Pivot updated worksheet.

2 Macro selects the employee code & date column.

3 Macro copies the data and goes back to the Wages Pivot Output worksheet.

4 Macro selects the first column the employee code column and jumps to the last cell with data.

5 Macro offsets and jumps down 1 cell to the next empty cell.

6 Macro pastes the data via paste special so only values are pasted.

7 Macro highlights column same colour as its predecessor.

*Sheets("Wages Pivot updated").Select*

*Range("X2:Y2").Select*

*Range(Selection, Selection.End(xlDown)).Select*

*Selection.Copy*

*Sheets("Wages Pivot Output").Select*

*Range("A1").Select*

*Selection.End(xlDown).Offset(1).Select*

*Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks \_*

*:=False, Transpose:=False*

*Application.CutCopyMode = False*

*With Selection.Interior*

*.Pattern = xlSolid*

*.PatternColorIndex = xlAutomatic*

*.Color = 65535*

*.TintAndShade = 0*

*.PatternTintAndShade = 0*

*End With*

1. At the end of the macro a few more checks are made to make (blank) cost code values into real cost code values.

1 lastRow3 refers to the last row of data in column F the cost code column.

2 rng refers to the range which is set to be all the data in column F up to and including last row of data.

3 cell refers to each cell in column F in the specified range which is the whole column F.

4 diff refers to the difference between a (blank) cell and a cell filled with a cost code value.

5 closestDiff is the closest number found away from the (blank) cell.

6 First we set the lastRow3 to store the row number of the last cell with data in it.

7 Next we set rng to be the whole range of column F.

8 Now we set a for each loop that looks at each cell in our range the whole of column. 9 In the for each loop we check if each cell in our range has the value (blank).

10 Next we set diff, closestDiff and closestNumber to the value 0.

11 If the value (blank) is found in our cell range we have our first match.

12 Once (blank) is found a new for each loop is created in our if loop.

13 This loop is for finding the next closest number to the (blank) value in our range.

14 *For Each c In rng.Resize(cell.row - 1, 1)* This code is the for each loop and works by going over each cell given the variable c in the range specified early given the variable rng. The rng has been resized to only go over the cells in the current row. -1 indicates one row up to the first row and the 1 indicates only in the first column, meaning this loop is looking for the closest number above our current cell not below.

15 *If IsNumeric(c.Value) Then* Checks to see if the value a numeric value so that calculations can be made later with numeric values.

16 *diff = cell.row - c.row* subtraction to show the difference between the current row “*cell.row*” and the row of the current cell “*c.row*”. Giving us the distance between the number of rows between the current cell and the original cell.

17 Next we start a new if loop that checks to see if the *closestNumber* is equal to 0 or if the difference is smaller than the previous closest difference. If *closestNumber* is 0 that means no closer number has been found. If the *diff* is bigger than the previous closest number *closestDiff* means, we also should not consider it because we want the number closest.

18 If both checks at number 17 fail that means we have found our next closest number, so we can assign the *closestNumber* variable to the current cells value *c.Value.*

19 We update the *closestDiff* variable to the new difference *diff*.

20 *Next c* shows the end of the loop and continues to the next cell in our range.

21 We repeat the same code changing our for each loop to look below our current cell for the closest non blank number and not above. The range needs to be modified for this as such *For Each c In rng.Resize(lastRow3 - cell.row + 1, 1).Offset(cell.row - 1)*.

22 Once both loops have determined what non blank number is closest we can assing the current blank cell number to be the closest number by this line *cell.Value = closestNumber*.

*'Filling in the cost code.*

*Dim lastRow3 As Long*

*Dim rng As Range*

*Dim cell As Range*

*Dim closestNumber As Double*

*Dim diff As Double*

*Dim closestDiff As Double*

*' Set the range to Column F*

*lastRow3 = Sheets("Wages Pivot Output").Cells(Rows.Count, "F").End(xlUp).row*

*Set rng = Sheets("Wages Pivot Output").Range("F1:F" & lastRow3)*

*' Loop through each cell in the range*

*For Each cell In rng*

*' Check if the cell is blank or "(blank)"*

*If IsNumeric(cell.Value) Or cell.Value = "(blank)" Then*

*closestNumber = 0*

*closestDiff = 0*

*diff = 0*

*' Find the closest non-blank number above*

*For Each c In rng.Resize(cell.row - 1, 1)*

*If IsNumeric(c.Value) Then*

*diff = cell.row - c.row*

*If closestNumber = 0 Or diff < closestDiff Then*

*closestNumber = c.Value*

*closestDiff = diff*

*End If*

*End If*

*Next c*

*' Find the closest non-blank number below*

*For Each c In rng.Resize(lastRow3 - cell.row + 1, 1).Offset(cell.row - 1)*

*If IsNumeric(c.Value) Then*

*diff = c.row - cell.row*

*If closestNumber = 0 Or diff < closestDiff Then*

*closestNumber = c.Value*

*closestDiff = diff*

*End If*

*End If*

*Next c*

*' Assign the closest number to the blank cell*

*cell.Value = closestNumber*

*End If*

*Next cell*

1. The last macro check is to determine if the hours column holds any zeros. If the hours have the value 0 the data is classified as irrelevant and should be deleted.

1 First we set our *ws* worksheet variable to be the Wages Pivot Output worksheet.

2 then we our *lastRow2* variable to last nonempty cell in the hour’s column, column D. This is done by going to the last cell in column D and going up to last nonempty cell. Making a range essentially.

3 Next we make a for loop that iterates over our hour’s column. *ws.Rows(i2).Delete*

4 Next our variable i2 which is looping down each cell in the hour’s column finds or is equal to the value 0 we have found a match.

5 Once a match has been found we can delete that row since 0 hours is irrelevant data.

6. *Next i2* Shows the end of the loop and goes on to the next row in the worksheets since we are decreasing by one row every time in our step-down statement at the very top of our loop *For i2 = lastRow2 To 2 Step -1*.

*' Specify the worksheet where you want to delete rows*

*Set ws = Sheets("Wages Pivot Output")*

*' Define the last row in the worksheet*

*lastRow2 = ws.Cells(ws.Rows.Count, "D").End(xlUp).row*

*' Loop through each row from the last row to the second row*

*For i2 = lastRow2 To 2 Step -1*

*' Check if the value in column D is 0*

*If ws.Cells(i2, "D").Value = 0 Then*

*' Delete the entire row if the condition is met*

*ws.Rows(i2).Delete*

*End If*

*Next i2*