Exercise: Budget Checker (Partner Up!)

Sunday, September 15, 2019 8:09 PM

Files:

- BudgetChecker
- README

Instructions:

- Create a VBA Script to complete the budget checker.
- There are three parts to this problem.
  - Part 1: Calculate the total after fees and enter the value in the "Total" cell.
  - Part II: Create a Message Box to alert the user as to whether the amount including fees is within or over budget.

Hints:

- Break up the problem into smaller steps.
- Look at old code!
- You got this!

Review: Budget Checker

Sunday, September 15, 2019 8:25 PM

```
' Part I
' -----------------------------------------------------

' 1. Retrieve the Price and Fees from the cells
Dim total As Double

' 2. Use these values to calculate the total
total = Range("F3").Value * (1 + Range("H3").Value)
' MsgBox(total)

' 3. Enter the total into the appropriate cell
Range("L3").Value = total
```

- First, we created a variable called `total` (as a double) to hold our full cost.
- Next, we combined the price and (1 + tax) to calculate the full cost.
- Finally, we entered the final cost into the total cell.

```
' Part II
' -----------------------------------------------------
' 4. Create a variable to store budget
Dim budget As Double
budget = Range("B3").Value
' msgbox(budget)

' 5. Compare using conditionals whether total is greater than or less than the budget
If budget > total Then

    MsgBox ("Under budget")

Else

    MsgBox ("Over budget")
End If
```

- First, we created a variable called `budget` to hold our budget amount.
- We then compared the value of our budget against the total.
- If the budget was greater than the total, we displayed a message that we

were under budget. If greater, the message stated that we were over budget.

For Loops

Sunday, September 15, 2019 9:04 PM

File:

- Activities/02-Ins_ForLoops/basic_for_loop.vbs
- Activities/02-Ins_ForLoops/basic_for_loop.xlsm

- For loops are statements for specifying iteration, which allows code to be executed repeatedly

```
' Loop through from numbers 1 through 20
For i = 1 To 20

    ' Iterate through the rows placing a value of 1 throughout
    Cells(i, 1).Value = 1

    ' Iterate through the columns placing a value of 5 throughout
    Cells(1, i).Value = 5

    ' Places increasing values based upon the variable "i" in B2 to B21
    Cells(i + 1, 2).Value = i + 1

' Call the next iteration
Next i
```

- `For i = 1 to 20` specifies the range to loop through.
- Subsequent uses of `i` change with the loop

```
' Loop through from numbers 1 through 20
For i = 1 To 20

    ' Iterate through the rows placing a value of 1 throughout
    Cells(i, 1).Value = 1

    ' Iterate through the columns placing a value of 5 throughout
    Cells(1, i).Value = 5

    ' Places increasing values based upon the variable "i" in B2 to B21
    Cells(i + 1, 2).Value = i + 1

' Call the next iteration
Next i
```

- `Next i` iterates to the next value of `i`.

```
' Loop through from numbers 1 through 20
For i = 1 To 20

    ' Iterate through the rows placing a value of 1 throughout
    Cells(i, 1).Value = 1

    ' Iterate through the columns placing a value of 5 throughout
    Cells(1, i).Value = 5

    ' Places increasing values based upon the variable "i" in B2 to B21
    Cells(i + 1, 2).Value = i + 1

' Call the next iteration
Next i
```

- Go to example

Exercise: Chicken Nugget Loop

Sunday, September 15, 2019 10:09 PM

Files:

- Activities/03-Stu_ChickenNuggets/Solved/chicken_nuggets.vbs
- Activities/03-Stu_ChickenNuggets/Solved/chicken_nuggets.xlsm

Looping on through

Now it's your chance to see how quickly we can create data utilizing the power of a computer and FOR loops!

Instructions

- Create a `For` loop that will produce the following example. The lines signify new cells.

A	B	C
I will eat	11	Chicken Nuggets
I will eat	12	Chicken Nuggets
I will eat	13	Chicken Nuggets
I will eat	14	Chicken Nuggets
I will eat	15	Chicken Nuggets
I will eat	16	Chicken Nuggets
I will eat	17	Chicken Nuggets
I will eat	18	Chicken Nuggets
I will eat	19	Chicken Nuggets
I will eat	20	Chicken Nuggets

Bonus

- If you finish early, talk to your neighbor about why you may want to use a For loop in preference to the "range" function.

Review: Chicken Nugget Loop

Thursday, September 19, 2019 6:09 PM

```
' Loop through first 10 rows
For i = 1 To 10

    ' Set values in column 1 to "I will eat"
    Cells(i, 1).Value = "I will eat "

    ' Set values in column 2 to the sum of the counter + 10
    Cells(i, 2).Value = i + 10

    ' Set values in column 3 to "Chicken Nuggets"
    Cells(i, 3).Value = "Chicken Nuggets"

' Call the next iteration
Next i
```

- We created a for loop that iterates from 1 through 10

```
' Loop through first 10 rows
For i = 1 To 10

    ' Set values in column 1 to "I will eat"
    Cells(i, 1).Value = "I will eat "

    ' Set values in column 2 to the sum of the counter + 10
    Cells(i, 2).Value = i + 10

    ' Set values in column 3 to "Chicken Nuggets"
    Cells(i, 3).Value = "Chicken Nuggets"

' Call the next iteration
Next i
```

- We set the value of `(i, 1)` and `(i, 3)` to be fixed value of "I will eat " and "Chicken Nuggets"

```
' Loop through first 10 rows
For i = 1 To 10

    ' Set values in column 1 to "I will eat"
    Cells(i, 1).Value = "I will eat "

    ' Set values in column 2 to the sum of the counter + 10
    Cells(i, 2).Value = i + 10

    ' Set values in column 3 to "Chicken Nuggets"
    Cells(i, 3).Value = "Chicken Nuggets"

' Call the next iteration
Next i
```

- We set the value of `(i, 2)` to be `i +10`. This forces the loop to print 11 through 20

```
' Loop through first 10 rows
For i = 1 To 10

    ' Set values in column 1 to "I will eat"
    Cells(i, 1).Value = "I will eat "

    ' Set values in column 2 to the sum of the counter + 10
    Cells(i, 2).Value = i + 10

    ' Set values in column 3 to "Chicken Nuggets"
    Cells(i, 3).Value = "Chicken Nuggets"

' Call the next iteration
Next i
```

- Lastly, we use `Next i` to signal we are done with the loop and onto the next one

```
' Loop through first 10 rows
For i = 1 To 10

    ' Set values in column 1 to "I will eat"
    Cells(i, 1).Value = "I will eat "

    ' Set values in column 2 to the sum of the counter + 10
    Cells(i, 2).Value = i + 10

    ' Set values in column 3 to "Chicken Nuggets"
    Cells(i, 3).Value = "Chicken Nuggets"

' Call the next iteration
Next i
```

Break Time (15 Minutes)

Thursday, September 19, 2019 5:54 PM

`

Loop Conditionals

Sunday, September 15, 2019 10:20 PM

Files:

- Activities/04-Ins_LoopConditionals/Solved/conditional_loops.vbs
- Activities/04-Ins_LoopConditionals/Solved/conditional_loops.xlsm
- Activities/04-Ins_LoopConditionals/Solved/modulus.vbs
- Activities/04-Ins_LoopConditionals/Solved/modulus.xlsm

Modulus

- modulus essentially performs long division on a number and returns the remainder
- Example

```
Sub modulo()

    ' Remainder of 0
    Cells(2, 1).Value = 4 Mod 2

    ' Remainder of 1
    Cells(3, 1).Value = 5 Mod 4

    ' Remainder of 3
    Cells(4, 1).Value = 11 Mod 8

    ' Remainder of 2
    Cells(5, 1).Value = 23 Mod 7

    ' Remainder of 5
    Cells(6, 1).Value = 24 Mod 19

End Sub
```

Loop Conditionals

```
' Create a for loop from 1 to 10
For i = 1 To 10

    ' Use the modulus function to determine if a number is divisible by 2 (even number)
    If Cells(i, 1).Value Mod 2 = 0 Then

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Even Row"

    ' If the number is not divisible by 2 (odd number)
    Else

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Odd Row"

    ' Close the If/Else Statement
    End If

Next i
```

- Structure of the loop

```
' Create a for loop from 1 to 10
For i = 1 To 10

    ' Use the modulus function to determine if a number is divisible by 2 (even number)
    If Cells(i, 1).Value Mod 2 = 0 Then

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Even Row"

    ' If the number is not divisible by 2 (odd number)
    Else

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Odd Row"

    ' Close the If/Else Statement
    End If

Next i
```

- Using Mod function

```
' Create a for loop from 1 to 10
For i = 1 To 10

    ' Use the modulus function to determine if a number is divisible by 2 (even number)
    If Cells(i, 1).Value Mod 2 = 0 Then

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Even Row"

    ' If the number is not divisible by 2 (odd number)
    Else

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Odd Row"

    ' Close the If/Else Statement
    End If

Next i
```

- using if-else statements to route the flow of logic depending on whether `i` is even or odd

```
' Create a for loop from 1 to 10
For i = 1 To 10

    ' Use the modulus function to determine if a number is divisible by 2 (even number)
    If Cells(i, 1).Value Mod 2 = 0 Then

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Even Row"

    ' If the number is not divisible by 2 (odd number)
    Else

        ' Enter "Even Row" the adjacent cell
        Cells(i, 2).Value = "Odd Row"

    ' Close the If/Else Statement
    End If

Next i
```

Exercise: Fizz Buzz

Sunday, September 15, 2019 11:12 PM


fizzbuzz

Files:

- Activities/05-Stu_FizzBuzz/README.md
- Activities/05-Stu_FizzBuzz/Unsolved/fizzbuzz.xlsm

Fizz Buzz

Instructions

- Create a VBA Script that populates the second column with the word "Fizz", "Buzz", or "Fizzbuzz" based on the value in the first column.
  - If the value in column 1 is a multiple of both 3 and 5, print "Fizzbuzz" in column 2.
  - If the value in column 1 is a multiple of just 3, print "Fizz" in column 2.
  - If the value in column 1 is a multiple of just 5, print "Buzz" in column 2.

Hints

- Remember the mod!

Review: Fizz Buzz

Sunday, September 15, 2019 11:18 PM

```
' Loop through the values in Column 1
For i = 2 To 100

    'Set cell value to variable
    num = Cells(i, 1).Value


    ' Check if the number is divisible by 3 and 5....
    If (num Mod 3 = 0 And num Mod 5 = 0) Then

        ' If so, print Fizzbuzz
        Cells(i, 2).Value = "Fizzbuzz"

    ' Check if the number is divisible by just 3...
    ElseIf (num Mod 3 = 0) Then

        ' If so, print "Fizz"
        Cells(i, 2).Value = "Fizz"

    ' Check if the number is divisible by just 5...
    ElseIf (num Mod 5 = 0) Then

        ' If so, print "Buzz"
        Cells(i, 2).Value = "Buzz"

    End If

Next i
```

- Created for loop

```
' Loop through the values in Column 1
For i = 2 To 100

    'Set cell value to variable
    num = Cells(i, 1).Value


    ' Check if the number is divisible by 3 and 5....
    If (num Mod 3 = 0 And num Mod 5 = 0) Then

        ' If so, print Fizzbuzz
        Cells(i, 2).Value = "Fizzbuzz"

    ' Check if the number is divisible by just 3...
    ElseIf (num Mod 3 = 0) Then

        ' If so, print "Fizz"
        Cells(i, 2).Value = "Fizz"

    ' Check if the number is divisible by just 5...
    ElseIf (num Mod 5 = 0) Then

        ' If so, print "Buzz"
        Cells(i, 2).Value = "Buzz"

    End If

Next i
```

- Created a variable to track the value of the number in column 1

```
' Loop through the values in Column 1
For i = 2 To 100

    'Set cell value to variable
    num = Cells(i, 1).Value


    ' Check if the number is divisible by 3 and 5....
    If (num Mod 3 = 0 And num Mod 5 = 0) Then

        ' If so, print Fizzbuzz
        Cells(i, 2).Value = "Fizzbuzz"

    ' Check if the number is divisible by just 3...
    ElseIf (num Mod 3 = 0) Then

        ' If so, print "Fizz"
        Cells(i, 2).Value = "Fizz"

    ' Check if the number is divisible by just 5...
    ElseIf (num Mod 5 = 0) Then

        ' If so, print "Buzz"
        Cells(i, 2).Value = "Buzz"

    End If

Next i
```

- Created a series of if-then statements.
    - We started these by checking for numbers that are both divisible by 3 and 5. It is important to start here, because we have to make sure our code handles the more specific scenario first. (If a number is divisible by 3 and 5, it is also divisible by 3)

```
' Loop through the values in Column 1
For i = 2 To 100

    'Set cell value to variable
    num = Cells(i, 1).Value


    ' Check if the number is divisible by 3 and 5....
    If (num Mod 3 = 0 And num Mod 5 = 0) Then

        ' If so, print Fizzbuzz
        Cells(i, 2).Value = "Fizzbuzz"

    ' Check if the number is divisible by just 3...
    ElseIf (num Mod 3 = 0) Then

        ' If so, print "Fizz"
        Cells(i, 2).Value = "Fizz"

    ' Check if the number is divisible by just 5...
    ElseIf (num Mod 5 = 0) Then

        ' If so, print "Buzz"
        Cells(i, 2).Value = "Buzz"

    End If

Next i
```

Exercise: Lotto Search (Partner Up)

Monday, September 16, 2019 6:10 PM

Files:

- Activities/06-Stu_Lotto/README.md
- Activities/06-Stu_Lotto/Unsolved/lotto_numbers.xlsm

lotto_num... (1)

Instructions

- You are in charge of finding our winners for a local lotto drawing.
  - The results are, in order:
    - First: 3957481
    - Second: 5865187
    - Third: 2817729
  - Create a script that will return those lucky winners and print them on the sheet.
    - For each winner include the following pieces of information:
      - First name
      - Last name
      - The winning number
    - They should be placed in winning order of First, Second, Third.
    - There should also be a message box that congratulates the first place winner.

Bonus

- There may just be one other winner! The below numbers are Wild Lotto Balls. Whichever comes up first in the list will be the fourth place (runner-up) winner.
  - 2275339
  - 5868182
  - 1841402

Hint

- Remember to utilize variables to keep your code clean.
- For the bonus, you may need to use `Exit For`

Review: Lotto Search

Monday, September 16, 2019 6:26 PM

- Create a series of variables to hold our ticket numbers and winner information
  - Note: Because the length of the tickets is so long, we needed to use `double` or `long`

```
' Create variables to hold winners. (Use "Long"
Dim first_place As Long
Dim second_place As Long
Dim third_place As Long
Dim runner1 As Long
Dim runner2 As Long
Dim runner3 As Long

' Establish the winning ticket numbers
first_place = 3957481
second_place = 5865187
third_place = 2817729

' Establish the runner-up numbers
runner1 = 2275339
runner2 = 5868182
runner3 = 1841402
```

- Create a for loop to scan through each of the rows
- Searches for instances when the value in Column 3 matches the value of our first, second, or third place winners
- If there is a match we copy the winner's first name, last name, and ticket information and place them into the winners table

```
' Loop through each of the lotto tickets
For i = 1 To 1001

    ' Check if the lotto number matches the first place winner...
    If Cells(i, 3).Value = first_place Then

        ' If so, create a message box specifying the first place win
        MsgBox " Congratulations " + Cells(i, 1).Value

        ' Retrieve the values associated with the winner and enter them into the winner's box.
        Cells(2, 6).Value = Cells(i, 1).Value
        Cells(2, 7).Value = Cells(i, 2).Value
        Cells(2, 8).Value = first_place

    ' Check if the lotto number matches the second place winner...
    ElseIf Cells(i, 3).Value = second_place Then

        ' Retrieve the values associated with the winner and enter them into the winner's box.
        Cells(3, 6).Value = Cells(i, 1).Value
        Cells(3, 7).Value = Cells(i, 2).Value
        Cells(3, 8).Value = second_place
```

- Create a second *separate* for loop
- The first for loop would replace our Runner Up winner with the final instance and not the first
- To avoid this, we needed to create a for-loop with an **`Exit for`**
- This code would exit the loop the moment the first runner up is found

```
' Loop through the lotto tickets a second time to find the first instance of a "runner-up" winner

For i = 1 to 1001

    ' BONUS: Check for runner ups with an OR operator
    If Cells(i, 3).Value = runner1 Or Cells(i, 3).Value = runner2 Or Cells(i, 3).Value = runner3 Then

        ' Retrieve the values associated with the winner and enter them into the winner's box.
        runner_up = Cells(i, 3).Value
        Cells(5, 6).Value = Cells(i, 1).Value
        Cells(5, 7).Value = Cells(i, 2).Value
        Cells(5, 8).Value = runner_up

        ' If first match is found, exit the for loop
        Exit for

    End If

Next i
```

Nested For Loops

Monday, September 16, 2019 6:50 PM

- For this example, we are looking to loop through both the rows and columns
- Let's simulate

Files:

- Activities/07-Ins_NestedForLoops/Solved/nested_for_loop.vbs
- Activities/07-Ins_NestedForLoops/Solved/nested_for_loop.xlsm


```
Dim TargetStudent As String

' Loop through the rows
For i = 1 To 3

    ' Loop through the columns
    For j = 1 To 5

        ' Print the Student Name
        MsgBox ("Row: " & i & " Column: " & j & " | " & Cells(i, j).Value)

    Next j

Next i
```

Jamie	Jeff	Lizette	Mariah	Ruchi
Sumana	Michael	Ronessa	Andy	Fervis
Deshan	Harshil	Jing	Sharanya	Sean

Exercise: Hornets Nets

Monday, September 16, 2019 7:08 PM

Files:

- Activities/08-Stu_HornetsNest/README.md
- Activities/08-Stu_HornetsNest/Unsolved/hornets_nest.xlsm

Hornets Nest

Instructions

- Create a VBA script to handle the growing Hornet infestation in your spreadsheet.
- There are three parts to this problem:
  - Part I: Count the number of Hornets found and display the number to your user in the form of a message box.
  - Part II: Modify the script such that it changes the word Hornets to "Bugs".
  - Part III: Modify the script a third time, this time keeping in mind that you have a limited number of Bugs and Bees. Use the full set of Bugs and Bees you have available to replace the Hornets. If you run out of Bugs or Bees provide the user with the message: "Oh no! We still have hornets..."

Hints

- You may want to create a backup of your spreadsheet as your macro will write over the contents.
- Take lots of deep breaths!

Review: Hornets Nest

Monday, September 16, 2019 7:27 PM

```
Dim HornetsCount as Integer

' Set the initial value for the HornetsCount to 0
HornetsCount = 0

' Loop through all rows
For i = 1 to 6

  ' Loop through all columns
  For j = 1 to 7

    ' If the value of a cell is equal to Hornets
    If Cells(i, j).Value = "Hornets" Then

      ' Add to the HornetsCounter
      HornetsCount = HornetsCount + 1

      ' Replace the Hornets with Bugs or Bees
      Cells(i, j).Value = "Bugs"

    End If

  Next j

Next i
```

- Part 1, utilized a nested for loop to search for the term "Hornets" in each of the cells.

```
Dim HornetsCount as Integer

' Set the initial value for the HornetsCount to 0
HornetsCount = 0

' Loop through all rows
For i = 1 to 6

  ' Loop through all columns
  For j = 1 to 7

    ' If the value of a cell is equal to Hornets
    If Cells(i, j).Value = "Hornets" Then

      ' Add to the HornetsCounter
      HornetsCount = HornetsCount + 1

      ' Replace the Hornets with Bugs or Bees
      Cells(i, j).Value = "Bugs"

    End If

  Next j

Next i
```

- Part 2, we change the value of these cells to be "Bugs".

```
Dim HornetsCount as Integer

' Set the initial value for the HornetsCount to 0
HornetsCount = 0

' Loop through all rows
For i = 1 to 6

  ' Loop through all columns
  For j = 1 to 7

    ' If the value of a cell is equal to Hornets
    If Cells(i, j).Value = "Hornets" Then

      ' Add to the HornetsCounter
      HornetsCount = HornetsCount + 1

      ' Replace the Hornets with Bugs or Bees
      Cells(i, j).Value = "Bugs"

    End If

  Next j

Next i
```

- Part 3, we had the added challenge of continually tracking our Bug and Bee count.
  - The easiest way to approach this problem was to draw from our Bug stash first and then once depleted to draw from our Bee stash.
  - In essence, this code works by storing our initial bug and bee count, then continually subtracting one from these variables each time we utilized either.
  - Once the Bug or Bee count is equal to 0, we can no longer draw from that stash.

```
' Loop through all rows
For i = 1 To 6

  ' Loop through all columns
  For j = 1 To 7

    ' If the value of a cell is equal to Hornets
    If Cells(i, j).Value = "Hornets" Then

      ' Add to the HornetsCounter
      HornetsCount = HornetsCount + 1

      ' Check if we have bugs available
      If (BugsCount > 0) Then

        ' Replace the Hornets with Bugs
        Cells(i, j).Value = "Bugs"

        ' Subtract from the Bugs Count
        BugsCount = BugsCount - 1

      ' Check if we have bees available
      ElseIf (BeesCount > 0) Then

        ' Replace the Hornets with Bees
        Cells(i, j).Value = "Bees"

        ' Subtract from the Bees Count
        BeesCount = BeesCount - 1

      End If

    End If

  Next j

Next i
```

If we have bugs available, use a bug and subtract one from the count.

If we have bees and no bugs, use a bee and subtract one from the count.

- For the last requirement of Part 3, we concluded if hornets were unaccounted for by comparing the sum of bugs and bees to the initial number of hornets.

  If there were more hornets than bugs and bees to start, then we know there are still hornets after our replacement efforts

```
' Show the number of hornets found
MsgBox (HornetsCount & " Hornets Found")

' Create the final message if we still have hornets
If (Range("L2").Value + Range("R2").Value < HornetsCount) Then

  MsgBox ("Oh no! We still have hornets... ")

End If
```

If the total of bugs and bees is less than the original hornet count, then we know there are still hornets.