

# Quiz 1 solutions and explanations

---

**IMPORTANT: Even if you do not intend/need to look through the solutions to Quiz 1, you still need to mark this quiz as completed using the blue "Mark as Completed" button in the lower right of this page. By doing so, you will unlock Assignment 1!**

This document is meant to provide clear explanations for the Quiz 1 questions (not the in-video quizzes since they have explanations already). I do NOT provide feedback during the quiz (like I do for the screencasts) because a learner could just guess, obtain the correct answers, then put them back into the quiz and get 100%!

This document is purely for you to learn more and to correct your misconceptions about the material. If you view this document soon after you take the quiz to see why you missed a certain question, it will serve as a great learning tool!

PLEASE DO NOT SHARE THIS DOCUMENT WITH ANYONE! Using this document to complete Quiz 1 is a violation of Coursera's Honor Code (a.k.a. cheating).

## **Question 1:**

Which of the following statements (select all that apply) are TRUE regarding subroutines, functions of arrays, and array functions?

A. A sub should be used if you want to modify the EXISTING contents of an array (e.g. you want to add 5 to all elements of the range A1:C5) and make these changes permanent.

Correct! An array function can be used to add 5 to the values in a range (like A1:C5) but those would be placed into a different region. For example, the user would select cells E1:G5, enter in the array formula, and the values would represent 5 added to each of the values in range A1:C5. If you want to make a permanent change, use a subroutine.

B. An array function must always return an array that has the same size as the object (range) that it acts on.

Incorrect. You could create an array function that had a 2x1 output area and would display the average and standard deviation of a range of data (any m x n range), for example.

C. If you Dim'ed A(3,4) without Option Base 1 then the size of A would be 3 rows by 4 columns.

Incorrect. Without Option Base 1, arrays have a zeroth row and a zeroth column. So, A would be a 4 x 5 array.

D. If you Dim'ed A(3,4) with Option Base 1 then the size of A would be 3 rows by 4 columns.

Correct! Option Base 1 eliminates the zeroth row and zeroth column, so A would be a 3 x 4 array.

E. Option Base 1 must always be used in subroutines that act on arrays and in array functions.

Incorrect. No, it doesn't have to be used. It's often necessary but not always. In general, though, it's a good idea to put Option Base 1 at the top of your module if you are using arrays and/or vectors.

**Question 2:**

When the Evaluate sub is run, how will the spreadsheet change?

	A	B	
1	0.4		
2	4.2		
3	3.1		
4	2.5		

Option Explicit

Option Base 1

---

```
Sub Evaluate()  
Dim xval(4) As Double, xm As Double  
Dim i As Integer  
Range("A1").Select  
For i = 1 To 4  
    xval(i) = ActiveCell.Offset(i - 1, 0).Value  
Next i  
xm = xval(1)  
For i = 2 To 4  
    If xval(i) > xm Then  
        xm = xval(i)  
    End If  
Next i  
ActiveCell.Offset(0, 2).Value = xm  
xm = xval(1)  
For i = 2 To 4  
    If xval(i) < xm Then  
        xm = xval(i)  
    End If  
Next i  
ActiveCell.Offset(1, 2).Value = xm  
ActiveCell.Offset(2, 1).Select  
End Sub
```

A.

	A	B	C
1	0.4		4.2
2	4.2		0.4
3	3.1		
4	2.5		

Correct! The first For...Next loop populates the xval vector with the items in cells A1:A4 since A1 is first selected as the active cell. Next, xm is set to be the first item of xval, which is 0.4. The second For...Next loop (i = 2 to 4) checks to see if any of the 2nd through 4th items of xval are greater than xm (0.4). If so, xm is reset to that value. The 2nd item of xval is greater than xm, so the new xm is 4.2. The "ActiveCell.Offset(0,2).Value = xm" line places 4.2 into cell C1 (offset 0 rows and 2 columns – importantly, the active cell remains the same since the Select method is not carried out). xm is once again set to the first item of the xval vector (0.4). The third For...Next loop searches through items 2 through 4 of the xval vector and if it finds an item that is less than xm it will redefine xm as that item. However, there are no items that are less than xm so xm remains 0.4. The "ActiveCell.Offset(1,2).Value = xm" line places 0.4 into cell C2 (offset 1 row and 2 columns. Finally, the "ActiveCell.Offset(2,1).Select" line selects cell B3 as the new active cell.

B.

	A	B	C
1	0.4		4.2
2	4.2		0.4
3	3.1		
4	2.5		

Incorrect. See the explanation above for option A.

C.

	A	B	C
1	0.4		0.4
2	4.2		4.2
3	3.1		
4	2.5		
-			

Incorrect. See the explanation above for option A.

D.

	A	B
1	0.4	4.2
2	2.5	3.1
3	3.1	2.5
4	4.2	0.4

Incorrect. See the explanation above for option A.

E.


	A	B
1	0.4	4.2
2	4.2	0.4
3	3.1	
4	2.5	

Incorrect. See the explanation above for option A.

### **Question 3:**

The DevAve array function determines the average of an array and then calculates how far from the average each element in the array is (see below). In this example, the average of the range is 2.5. NOTE: "Option Base 1" has been written at the top of the module.

	A	B	C
1	2	3	3.5
2	1	2.5	4.5
3	2.5	2	1.5
4			
5	=DevAve(A1:C3)		
6			
7			



	A	B	C
1	2	3	3.5
2	1	2.5	4.5
3	2.5	2	1.5
4			
5	-0.5	0.5	1
6	-1.5	0	2
7	0	-0.5	-1

```

1  Function DevAve(DataRange as Range) As Double
2  Dim nRows As Integer, nCols As Integer
3  Dim i As Integer, j As Integer, Ave As Integer
4  Dim A() As Double
5  nRows = Selection.Rows.Count
6  nCols = Selection.Columns.Count
7  Dim A(nRows, nCols) As Double
8  Ave = Application.WorksheetFunction.Average(DataRange)
9  For i = 1 To nRows
10     For j = 1 To nCols
11         A(i, j) = DataRange.Cells(i, j) - Ave
12     Next j
13 Next i
14 A = DevAve
15 End Function

```


Select all lines that HAVE errors. HINT: 6 lines have errors!

Solution: In line 1 the data type for the output should be Variant (whenever there is an array or vector that is created within VBA and you wish to output it, the data type needs to be Variant). Line 2 is fine. Ave in line 3 should be Dim'ed As Double. Line 4 is okay. The object in this function is DataRange, NOT Selection, so lines 5 and 6 should be nRows = DataRange.Rows.Count and nCols = DataRange.Columns.Count, respectively. Line 7 should use ReDim instead of Dim. Lines 8-13 are good. Line 14 should be switched around: DevAve = A.

#### Question 4:

When the rotate array function is run (Ctrl-Shift-Enter), the worksheet layout shown below results. In the VBA code for the function, what goes in the space indicated by <What Goes Here?>?

	A	B	C
1	1	2	3
2	4	5	6
3	7	8	9
4			
5	=rotate(A1:C3)		
6			
7			



	A	B	C
1	1	2	3
2	4	5	6
3	7	8	9
4			
5	7	4	1
6	8	5	2
7	9	6	3

Option Explicit

Option Base 1

---

```
Function rotate(datarange As Range)
Dim i As Integer, j As Integer
Dim nr As Integer, nc As Integer
Dim B() As Variant
nr = datarange.Rows.Count
nc = datarange.Columns.Count
ReDim B(nr, nc) As Variant
For i = 1 To nr
    For j = 1 To nc
        B(i, j) = datarange.Cells(<What Goes Here?>, i)
    Next j
Next i
rotate = B
End Function
```

A.  $nr + 1 - j$

Correct! For this 3 x 3 array, we wish to make the following “mappings” (just a few examples):

Position (3,1) of datarange  $\square$  Position (1,1) of B

Position (1,1) of datarange  $\square$  Position (1,3) of B

Position (1,3) of datarange  $\square$  Position (3,3) of B

Position (3,3) of datarange  $\square$  Position (3,1) of B

We can note that we always place the column number of datarange into the row number of B. So, that’s why the second index is i in “datarange.Cells( <What Goes Here?>, i). However, the first index <What Goes here?> is a bit trickier and more or less requires some trial and error. However, a general formula that maps the column of datarange to the row of B is  $nr + 1 - j$  (this will work for \*any\* square matrix with nr number of rows).

B.  $nc + j - 1$

Incorrect. See the explanation above for Option A.

C.  $i + j - 1$

Incorrect. See the explanation above for Option A.



D.  $nr*(nc - j)$

Incorrect. See the explanation above for Option A.

E.  $3 - nr + j$

Incorrect. See the explanation above for Option A.

### **Question 5:**

When the SomeSubroutine sub is run, which cell of the spreadsheet will have the word "Hello" in it? Only enter a letter followed by a number, no spaces and no other text, when you submit your answer.

#### Option Explicit

---

```
Sub SomeSubroutine()  
Dim A() As Variant, k As Integer  
Dim i As Integer, j As Integer  
k = 1  
ReDim A(k) As Variant  
A(0) = WorksheetFunction.RandBetween(1, 10)  
A(1) = "Hello"  
For i = 1 To 4  
    A = SomeFunction(A, k)  
    k = k + 1  
    ReDim Preserve A(k) As Variant  
    A(k) = WorksheetFunction.RandBetween(1, 10)  
Next i  
Range("A1:A" & k + 1) = WorksheetFunction.Transpose(A)  
End Sub
```

---

```
Function SomeFunction(A As Variant, k As Integer) As Variant  
Dim i As Integer, B() As Variant  
ReDim B(k) As Variant  
For i = 0 To k  
    B(i) = A(k - i)  
Next i  
SomeFunction = B  
End Function
```

Solution: Starting from the beginning, k will equal 1, A(1) will be ReDim'ed as Variant (a vector of size 2 since Option Base 1 not used). The 0th position of A will be a random number between 1 and 10. The 1st position of A will be "Hello". The first For...Next loop will take the vector A, pass it into the SomeFunction function with k = 1. The SomeFunction function will create a vector B of size 2 then the For...Next loop will place A(1) into B(0) and A(0) into B(1) – essentially, this swaps vector A and vector B will now be: B(0) = "Hello" and B(1) = randomnumber. The SomeFunction function returns B to the vector A in the "A = SomeFunction(A,k)" line of

SomeSubroutine, overwriting the old A. So, the new A is: A(0) = "Hello" and A(1) = randomnumber. k is now incremented to 2, A is ReDim'ed while preserving the previous items, so A is now size 3 with empty position 2. A(2) is then set to another random number. That was all for the iteration where i =1 (note that i is not used in any calculations).

After the 1st iteration: A = ["Hello" r1 r2] where r1 and r2 are just random numbers.

The 2nd iteration of the For...Next loop will swap the order of the preexisting A vector and add an item. After this 2nd iteration the vector A will be: A = [r2 r1 "Hello" r3].

The 3rd iteration of the For...Next loop will swap the order of the preexisting A vector and add an item. After this 3rd iteration the vector A will be: A = [r3 "Hello" r1 r2 r4].

The 4th iteration of the For...Next loop will swap the order of the preexisting A vector and add an item. After this 4th iteration the vector A will be: A = [r4 r2 r1 "Hello" r3 r5].

Finally, the vector A is transposed and placed into cells A1:A6. A4 will therefore have "Hello" in it. Whew!