**20 November 2024 at 09:08**

**You said:**

I am a Microsoft 365 developer.

I am testing MAKEARRAY function. But I want to integrate with python's rdrand in a way that I will give range of random numbers (say, between 1001 and 10000) and number of such random numbers to be generated as input to rdrand library (remember that I am using rdrand but using rdseed function in that directory).

The number of such random numbers to will be arranged by wrpper function MAKEARRAY with inputs as number of rows and coulmns.

If however the numbers generated fall short of array dimensions then balance elements of array will be filled with nulls in excel. If numbers are excess compared to dimensions, then excess will be ignored.  
  
Give me python code to implement this.

**Copilot said:**

Sure! Here's a Python code snippet that integrates the rdrand library to generate random numbers within a specified range and arranges them in an array with given dimensions. If the generated numbers fall short of the array dimensions, the remaining elements will be filled with None. If the numbers exceed the dimensions, the excess numbers will be ignored.

import rdrand

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

def make\_array(rows, cols, random\_numbers):

array = []

total\_elements = rows \* cols

for i in range(rows):

row = []

for j in range(cols):

if len(random\_numbers) > 0:

row.append(random\_numbers.pop(0))

else:

row.append(None)

array.append(row)

return array

# Example usage

min\_val = 1001

max\_val = 10000

count = 20 # Number of random numbers to generate

rows = 4

cols = 5

random\_numbers = generate\_random\_numbers(min\_val, max\_val, count)

array = make\_array(rows, cols, random\_numbers)

for row in array:

print(row)

This code will generate random numbers using the rdseed function from the rdrand library and arrange them in a 2D array with the specified number of rows and columns. If the number of generated random numbers is less than the total elements in the array, the remaining elements will be filled with None. If there are more random numbers than needed, the excess numbers will be ignored.

Feel free to adjust the min\_val, max\_val, count, rows, and cols variables as needed for your specific use case. Let me know if you need any further assistance!

**Code**

Copy

1 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:11**

**You said:**

What if I want to use excel 365's new built in function MAKEARRAY()?

**Copilot said:**

If you want to use Excel 365's built-in MAKEARRAY function to create an array with random numbers, you can combine it with a custom Lambda function to generate the random numbers. Here's how you can do it:

1. **Define a Lambda function** to generate random numbers within a specified range.
2. **Use the MAKEARRAY function** to create the array with the desired dimensions.

Here's an example of how you can achieve this:

1. **Define the Lambda function**:
   * Go to the **Formulas** tab.
   * Click on **Name Manager**.
   * Click **New** and define a new name, e.g., RandomNumber.
   * In the **Refers to** box, enter the following formula:

=LAMBDA(min, max, RANDARRAY(1, 1) \* (max - min) + min)

1. **Use the MAKEARRAY function**:
   * In a cell where you want the array to start, enter the following formula:

=MAKEARRAY(4, 5, LAMBDA(r, c, IF(SEQUENCE(4, 5) <= 20, RandomNumber(1001, 10000), NA())))

In this example:

* 4 and 5 are the number of rows and columns, respectively.
* 20 is the number of random numbers you want to generate.
* RandomNumber(1001, 10000) generates a random number between 1001 and 10000.
* IF(SEQUENCE(4, 5) <= 20, RandomNumber(1001, 10000), NA()) ensures that if the number of elements exceeds 20, the remaining cells will be filled with #N/A.

This approach leverages Excel's built-in functions to create a dynamic array with random numbers, filling any excess cells with #N/A.

Feel free to adjust the dimensions and the range of random numbers as needed. Let me know if you need any further assistance!

Edit in Pages

Copy

2 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:13**

**You said:**

But there is no function as RandomNumber(1001, 10000) - I want to you python RDSEED as above in that place.

**Copilot said:**

Got it! To integrate Python's rdseed function with Excel's MAKEARRAY, you can use a combination of Python and Excel. Here's a step-by-step guide:

1. **Generate Random Numbers in Python**:
   * Use Python to generate the random numbers using the rdseed function.
   * Save these numbers to a CSV file.
2. **Import the CSV into Excel**:
   * Use Excel to read the CSV file and populate the array using the MAKEARRAY function.

Here's how you can do it:

**Step 1: Generate Random Numbers in Python**

import rdrand

import csv

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

# Example usage

min\_val = 1001

max\_val = 10000

count = 20 # Number of random numbers to generate

random\_numbers = generate\_random\_numbers(min\_val, max\_val, count)

# Save to CSV

with open('random\_numbers.csv', 'w', newline='') as csvfile:

writer = csv.writer(csvfile)

writer.writerow(random\_numbers)

**Step 2: Import the CSV into Excel**

1. **Open Excel** and go to the **Data** tab.
2. Click on **Get Data** > **From File** > **From Text/CSV**.
3. Select the random\_numbers.csv file and import it.

**Step 3: Use the MAKEARRAY Function**

Assuming the random numbers are imported into column A starting from A1, you can use the MAKEARRAY function to create the array:

=MAKEARRAY(4, 5, LAMBDA(r, c, IF((r-1)\*5+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*5+c), NA())))

In this formula:

* 4 and 5 are the number of rows and columns, respectively.
* COUNTA(A:A) counts the number of random numbers in column A.
* INDEX(A:A, (r-1)\*5+c) retrieves the random number from the list.
* IF((r-1)\*5+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*5+c), NA()) ensures that if the number of elements exceeds the count of random numbers, the remaining cells will be filled with #N/A.

This approach combines Python and Excel to achieve your goal. Let me know if you need any further assistance!

Edit in Pages

Copy

3 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:14**

**You said:**

I want to do it using xlwings not through external csv file so that everything dynamic from within excel.

**Copilot said:**

Got it! You can use xlwings to integrate Python directly with Excel, allowing you to generate random numbers using the rdseed function and populate an Excel sheet dynamically. Here's how you can do it:

**Step 1: Install xlwings**

Make sure you have xlwings installed. You can install it using pip:

pip install xlwings

**Step 2: Write the Python Script**

Here's a Python script that generates random numbers using rdseed and populates an Excel sheet using xlwings:

import rdrand

import xlwings as xw

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

def populate\_excel(rows, cols, random\_numbers):

# Start an instance of Excel

wb = xw.Book()

sheet = wb.sheets[0]

total\_elements = rows \* cols

index = 0

for i in range(rows):

for j in range(cols):

if index < len(random\_numbers):

sheet.cells(i + 1, j + 1).value = random\_numbers[index]

index += 1

else:

sheet.cells(i + 1, j + 1).value = None

# Example usage

min\_val = 1001

max\_val = 10000

count = 20 # Number of random numbers to generate

rows = 4

cols = 5

random\_numbers = generate\_random\_numbers(min\_val, max\_val, count)

populate\_excel(rows, cols, random\_numbers)

**Step 3: Run the Script**

Run the script, and it will open an Excel workbook, generate the random numbers, and populate the cells dynamically.

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using rdseed.
* **populate\_excel**: This function uses xlwings to open an Excel workbook and populate it with the generated random numbers. If the number of random numbers is less than the total elements in the array, the remaining cells will be filled with None.

This approach keeps everything dynamic within Excel using xlwings, without the need for external CSV files. Let me know if you need any further assistance!

Edit in Pages

Copy

4 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:17**

**You said:**

But you have not used MAKEARRAY anywhere in above. Instead you have defined custom function def populate\_excel.  
  
You can generated an array of random numbers using rdseed and xlwings and thenMAKEARRAY will arrange the numbers in rows and columns as defined in input. Excess will be ignored and shortage will be filled with "" (Null) in excel.

**Copilot said:**

I understand your requirement better now. Let's use xlwings to generate random numbers using rdseed and then use Excel's MAKEARRAY function to arrange these numbers in the desired format.

Here's the updated approach:

1. **Generate Random Numbers in Python** using rdseed.
2. **Populate Excel** with these numbers.
3. **Use the MAKEARRAY function** in Excel to arrange the numbers.

**Step 1: Generate Random Numbers in Python**

import rdrand

import xlwings as xw

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

def populate\_excel\_with\_makearray(rows, cols, random\_numbers):

# Start an instance of Excel

wb = xw.Book()

sheet = wb.sheets[0]

# Write the random numbers to a temporary range

temp\_range = sheet.range('A1').expand('table')

temp\_range.value = random\_numbers

# Define the MAKEARRAY formula

makearray\_formula = f"=MAKEARRAY({rows}, {cols}, LAMBDA(r, c, IF((r-1)\*{cols}+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*{cols}+c), \"\")))"

# Write the MAKEARRAY formula to the desired range

sheet.range('B1').formula = makearray\_formula

# Example usage

min\_val = 1001

max\_val = 10000

count = 20 # Number of random numbers to generate

rows = 4

cols = 5

random\_numbers = generate\_random\_numbers(min\_val, max\_val, count)

populate\_excel\_with\_makearray(rows, cols, random\_numbers)

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using rdseed.
* **populate\_excel\_with\_makearray**: This function uses xlwings to open an Excel workbook, write the random numbers to a temporary range, and then use the MAKEARRAY function to arrange these numbers in the desired format.

**Running the Script:**

1. **Run the Python script**. It will open an Excel workbook, generate the random numbers, and populate the cells dynamically.
2. **Check the Excel sheet**. The MAKEARRAY function will arrange the numbers in the specified rows and columns, filling any excess cells with "" (Null).

This approach keeps everything dynamic within Excel using xlwings and the MAKEARRAY function. Let me know if you need any further assistance!

**Code**

Copy

5 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:18**

**You said:**

Why you are doing this:  
  
# Define the MAKEARRAY formula makearray\_formula = f"=MAKEARRAY({rows}, {cols}, LAMBDA(r, c, IF((r-1)\*{cols}+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*{cols}+c), \"\")))"  
  
I want to use excel's nuilt in MAKEARRAY and inputs for that will be output array from xlwings using rdseed.

**Copilot said:**

I understand your requirement now. You want to generate random numbers using Python's rdseed function and then use Excel's built-in MAKEARRAY function to arrange these numbers dynamically. Here's how you can achieve this using xlwings:

**Step 1: Generate Random Numbers in Python**

import rdrand

import xlwings as xw

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

def populate\_excel\_with\_random\_numbers(random\_numbers):

# Start an instance of Excel

wb = xw.Book()

sheet = wb.sheets[0]

# Write the random numbers to a temporary range

sheet.range('A1').value = random\_numbers

# Example usage

min\_val = 1001

max\_val = 10000

count = 20 # Number of random numbers to generate

random\_numbers = generate\_random\_numbers(min\_val, max\_val, count)

populate\_excel\_with\_random\_numbers(random\_numbers)

**Step 2: Use the MAKEARRAY Function in Excel**

1. **Run the Python script**. It will open an Excel workbook and populate column A with the generated random numbers.
2. **Use the MAKEARRAY function** in Excel to arrange these numbers.

In Excel, you can use the following formula to create the array:

=MAKEARRAY(4, 5, LAMBDA(r, c, IF((r-1)\*5+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*5+c), "")))

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using rdseed.
* **populate\_excel\_with\_random\_numbers**: This function uses xlwings to open an Excel workbook and write the random numbers to column A.
* **MAKEARRAY formula**: This formula arranges the numbers in a 4x5 array. If the number of elements exceeds the count of random numbers, the remaining cells will be filled with "" (Null).

This approach keeps everything dynamic within Excel using xlwings and the MAKEARRAY function. Let me know if you need any further assistance!

**Code**

Copy

6 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:22**

**You said:**

This is completely out of place to what I want. Explaining again:  
  
First step will be write function UDF which will be hidden in xlwings. That function will generate array of numbers using rdseed. Number of random numbers and range of random numbers will be given by user. Let's accomplish this part first. Output will be single column array.

**Copilot said:**

Got it! Let's start by creating a User Defined Function (UDF) in xlwings that generates an array of random numbers using rdseed. The number of random numbers and the range will be provided by the user. The output will be a single column array.

**Step 1: Install xlwings**

Make sure you have xlwings installed. You can install it using pip:

pip install xlwings

**Step 2: Write the UDF in Python**

Here's the Python code to create the UDF:

import rdrand

import xlwings as xw

@xw.func

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.rdseed() % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

# To use this UDF in Excel, you need to run the following command in your terminal:

# xlwings runpython install

**Step 3: Use the UDF in Excel**

1. **Run the Python script** to register the UDF with Excel.
2. **Open Excel** and use the UDF as follows:
   * In a cell, enter the formula: =generate\_random\_numbers(1001, 10000, 20)

This will generate 20 random numbers between 1001 and 10000 and display them in a single column.

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using rdseed.
* **@xw.func**: This decorator registers the function as an Excel UDF.

This will allow you to dynamically generate random numbers within Excel using the rdseed function from Python. Let me know if you need any further assistance!

**Code**

Copy

7 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:32**

**You said:**

You missed RdSeedom - rdseed was wrong. I corrected and tried to use first in python. But traceback:  
>>> def generate\_random\_numbers(min\_val, max\_val, count): ... random\_numbers = [] ... for \_ in range(count): ... random\_number = rdrand.RdSeedom() % (max\_val - min\_val + 1) + min\_val ... random\_numbers.append(random\_number) ... return random\_numbers ... >>> generate\_random\_numbers(100000000000001,999999999999999,1000) Traceback (most recent call last): File "", line 1, in File "", line 4, in generate\_random\_numbers TypeError: unsupported operand type(s) for %: 'RdSeedom' and 'int'

**Copilot said:**

It looks like there's an issue with how the RdSeedom function is being used. The RdSeedom function from the rdrand library returns an object, not a direct integer, which is why you're encountering the TypeError.

Let's correct this by ensuring we extract the integer value from the RdSeedom object. Here's the updated Python code:

**Step 1: Correct the Python Function**

import rdrand

import xlwings as xw

@xw.func

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = int(rdrand.RdSeedom()) % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

# To use this UDF in Excel, you need to run the following command in your terminal:

# xlwings runpython install

**Step 2: Use the UDF in Excel**

1. **Run the Python script** to register the UDF with Excel.
2. **Open Excel** and use the UDF as follows:
   * In a cell, enter the formula: =generate\_random\_numbers(1001, 10000, 20)

This will generate 20 random numbers between 1001 and 10000 and display them in a single column.

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using RdSeedom and converts the result to an integer.
* **@xw.func**: This decorator registers the function as an Excel UDF.

This should resolve the issue and allow you to dynamically generate random numbers within Excel using the RdSeedom function from Python. Let me know if you need any further assistance!

Edit in Pages

Copy

8 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:34**

**You said:**

&gt;&gt;&gt; generate\_random\_numbers(100000000000001,999999999999999,1000) Traceback (most recent call last): File "", line 1, in File "", line 4, in generate\_random\_numbers TypeError: int() argument must be a string, a bytes-like object or a real number, not 'RdSeedom'

**Copilot said:**

It looks like the RdSeedom object needs to be properly converted to an integer. Let's correct this by using the appropriate method to extract the integer value from the RdSeedom object.

Here's the updated Python code:

**Step 1: Correct the Python Function**

import rdrand

import xlwings as xw

@xw.func

def generate\_random\_numbers(min\_val, max\_val, count):

random\_numbers = []

for \_ in range(count):

random\_number = rdrand.RdSeedom().get\_bits(64) % (max\_val - min\_val + 1) + min\_val

random\_numbers.append(random\_number)

return random\_numbers

# To use this UDF in Excel, you need to run the following command in your terminal:

# xlwings runpython install

**Step 2: Use the UDF in Excel**

1. **Run the Python script** to register the UDF with Excel.
2. **Open Excel** and use the UDF as follows:
   * In a cell, enter the formula: =generate\_random\_numbers(1001, 10000, 20)

This will generate 20 random numbers between 1001 and 10000 and display them in a single column.

**Explanation:**

* **generate\_random\_numbers**: This function generates the specified number of random numbers within the given range using RdSeedom and converts the result to an integer using get\_bits(64).
* **@xw.func**: This decorator registers the function as an Excel UDF.

This should resolve the issue and allow you to dynamically generate random numbers within Excel using the RdSeedom function from Python. Let me know if you need any further assistance!

**Code**

Copy

9 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:41**

**You said:**

I finalized udf like this:  
  
import rdrand import xlwings as xw @xw.func def RDSEED\_EXCEL(min\_val, max\_val, count): random\_numbers\_rdseed = [] for \_ in range(count): random\_number\_rdseed = rdrand.RdSeedom().get\_bits(64) % (max\_val - min\_val + 1) + min\_val random\_numbers\_rdseed.append(random\_number\_rdseed) return [[random\_number\_rdseed\_1] for random\_number\_rdseed\_1 in random\_numbers\_rdseed]  
  
As you can see the last statement return - I have converted to list of lists because that will generate vertical array. Now the challange is how to nest this function within excel's MAKEARRAY built in function.

**Copilot said:**

Great job on finalizing the UDF! Now, to nest this function within Excel's MAKEARRAY function, you can follow these steps:

1. **Generate the random numbers using the UDF**.
2. **Use the MAKEARRAY function** to arrange these numbers in the desired format.

**Step 1: Generate Random Numbers Using the UDF**

First, ensure your UDF is correctly registered and can be called from Excel. You can use it in a cell like this:

=RDSEED\_EXCEL(1001, 10000, 20)

This will generate 20 random numbers between 1001 and 10000 in a vertical array.

**Step 2: Use the MAKEARRAY Function**

To use the MAKEARRAY function to arrange these numbers, you can follow these steps:

1. **Generate the random numbers** and place them in a temporary range, say column A.
2. **Use the MAKEARRAY function** to arrange these numbers in the desired format.

Here’s how you can do it:

1. **Generate the random numbers**:
   * In cell A1, enter the formula: =RDSEED\_EXCEL(1001, 10000, 20)
2. **Use the MAKEARRAY function**:
   * In another cell, use the following formula to create the array:

=MAKEARRAY(4, 5, LAMBDA(r, c, IF((r-1)\*5+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*5+c), "")))

**Explanation:**

* **RDSEED\_EXCEL(1001, 10000, 20)**: This generates 20 random numbers between 1001 and 10000 and places them in column A.
* **MAKEARRAY(4, 5, LAMBDA(r, c, IF((r-1)\*5+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*5+c), "")))**: This formula arranges the numbers in a 4x5 array. If the number of elements exceeds the count of random numbers, the remaining cells will be filled with "" (Null).

This approach leverages both your UDF and Excel's built-in MAKEARRAY function to dynamically generate and arrange random numbers. Let me know if you need any further assistance!

Edit in Pages

Copy

10 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 09:43**

**You said:**

Integrate RDSEED\_EXCEL within MAKEARRAY using LAMBDA so that everything is in a single place in excel.

**Copilot said:**

To integrate your RDSEED\_EXCEL function within Excel's MAKEARRAY function using LAMBDA, you can follow these steps:

1. **Ensure your UDF is correctly registered and can be called from Excel**.
2. **Use the MAKEARRAY function** with a LAMBDA that calls your UDF.

Here's how you can do it:

**Step 1: Register the UDF**

Make sure your UDF is registered and available in Excel. Your UDF should look like this:

import rdrand

import xlwings as xw

@xw.func

def RDSEED\_EXCEL(min\_val, max\_val, count):

random\_numbers\_rdseed = []

for \_ in range(count):

random\_number\_rdseed = rdrand.RdSeedom().get\_bits(64) % (max\_val - min\_val + 1) + min\_val

random\_numbers\_rdseed.append(random\_number\_rdseed)

return [[random\_number\_rdseed\_1] for random\_number\_rdseed\_1 in random\_numbers\_rdseed]

# To use this UDF in Excel, you need to run the following command in your terminal:

# xlwings runpython install

**Step 2: Use the MAKEARRAY Function**

In Excel, you can use the MAKEARRAY function with a LAMBDA that calls your UDF. Here’s an example formula:

=MAKEARRAY(4, 5, LAMBDA(r, c, IF((r-1)\*5+c <= COUNTA(RDSEED\_EXCEL(1001, 10000, 20)), INDEX(RDSEED\_EXCEL(1001, 10000, 20), (r-1)\*5+c), "")))

**Explanation:**

* **RDSEED\_EXCEL(1001, 10000, 20)**: This generates 20 random numbers between 1001 and 10000.
* **MAKEARRAY(4, 5, LAMBDA(r, c, ...))**: This creates a 4x5 array.
* **IF((r-1)\*5+c <= COUNTA(RDSEED\_EXCEL(1001, 10000, 20)), INDEX(RDSEED\_EXCEL(1001, 10000, 20), (r-1)\*5+c), "")**: This part of the LAMBDA function checks if the current index is within the range of generated numbers. If it is, it uses INDEX to get the corresponding random number. If not, it returns an empty string ("").

This approach ensures that everything is dynamic and contained within a single formula in Excel. Let me know if you need any further assistance!

**Code**

Copy

11 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 10:01**

**You said:**

What is wrong with this:  
  
=MAKEARRAY(100,10,LAMBDA(x,y,RDSEED\_EXCEL(100000000000001,999999999999999,1000)))  
  
Why can't my vertical array from rdseed\_excel function get arranged accroding to MAKEARRARY?

**Copilot said:**

The issue with your formula is that RDSEED\_EXCEL generates a vertical array, but MAKEARRAY expects a single value for each cell in the array it creates. To integrate RDSEED\_EXCEL within MAKEARRAY, you need to ensure that each call to the LAMBDA function retrieves the correct value from the generated array.

Here's how you can achieve this:

1. **Generate the random numbers** using RDSEED\_EXCEL and store them in a named range or a temporary range.
2. **Use the MAKEARRAY function** to arrange these numbers.

Since RDSEED\_EXCEL generates a vertical array, you need to reference this array correctly within the LAMBDA function.

**Step-by-Step Solution**

1. **Generate the random numbers** and store them in a named range:
   * In a cell, enter the formula: =RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)
   * Let's assume this formula is in the range A1:A1000.
2. **Use the MAKEARRAY function** to arrange these numbers:
   * In another cell, use the following formula to create the array:

=MAKEARRAY(100, 10, LAMBDA(r, c, IF((r-1)\*10+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*10+c), "")))

**Explanation:**

* **RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)**: This generates 1000 random numbers between 100000000000001 and 999999999999999 and places them in column A.
* **MAKEARRAY(100, 10, LAMBDA(r, c, IF((r-1)\*10+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*10+c), "")))**: This formula arranges the numbers in a 100x10 array. If the number of elements exceeds the count of random numbers, the remaining cells will be filled with "" (Null).

**Integrating Everything in One Formula**

If you want to integrate everything into a single formula without using a temporary range, you can use a combination of LET and MAKEARRAY functions:

=LET(

randomNumbers, RDSEED\_EXCEL(100000000000001, 999999999999999, 1000),

MAKEARRAY(100, 10, LAMBDA(r, c, IF((r-1)\*10+c <= COUNTA(randomNumbers), INDEX(randomNumbers, (r-1)\*10+c), "")))

)

**Explanation:**

* **LET function**: This function allows you to define a named variable (randomNumbers) that holds the array generated by RDSEED\_EXCEL.
* **MAKEARRAY function**: This function uses the randomNumbers array to populate the 100x10 array.

This approach ensures that everything is dynamic and contained within a single formula in Excel. Let me know if you need any further assistance!

Edit in Pages

Copy

12 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 10:04**

**You said:**

Instead of LET what if I want to use LAMBDA within MAKEARRAY?

**Copilot said:**

To integrate RDSEED\_EXCEL within MAKEARRAY using LAMBDA without using LET, you can directly call the UDF within the LAMBDA function. However, since RDSEED\_EXCEL returns an array, you need to ensure that each call to the LAMBDA function retrieves the correct value from this array.

Here's how you can do it:

1. **Generate the random numbers** using RDSEED\_EXCEL and store them in a named range or directly within the LAMBDA function.
2. **Use the MAKEARRAY function** to arrange these numbers.

**Step-by-Step Solution**

1. **Generate the random numbers** and store them in a named range:
   * In a cell, enter the formula: =RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)
   * Let's assume this formula is in the range A1:A1000.
2. **Use the MAKEARRAY function** to arrange these numbers:
   * In another cell, use the following formula to create the array:

=MAKEARRAY(100, 10, LAMBDA(r, c, IF((r-1)\*10+c <= COUNTA(A:A), INDEX(A:A, (r-1)\*10+c), "")))

**Integrating Everything in One Formula with LAMBDA**

If you want to integrate everything into a single formula using LAMBDA within MAKEARRAY, you can do it like this:

=MAKEARRAY(100, 10, LAMBDA(r, c,

LET(

randomNumbers, RDSEED\_EXCEL(100000000000001, 999999999999999, 1000),

IF((r-1)\*10+c <= COUNTA(randomNumbers), INDEX(randomNumbers, (r-1)\*10+c), "")

)

))

**Explanation:**

* **RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)**: This generates 1000 random numbers between 100000000000001 and 999999999999999.
* **MAKEARRAY(100, 10, LAMBDA(r, c, ...))**: This creates a 100x10 array.
* **LET function within LAMBDA**: This defines randomNumbers to hold the array generated by RDSEED\_EXCEL.
* **IF((r-1)\*10+c <= COUNTA(randomNumbers), INDEX(randomNumbers, (r-1)\*10+c), "")**: This part of the LAMBDA function checks if the current index is within the range of generated numbers. If it is, it uses INDEX to get the corresponding random number. If not, it returns an empty string ("").

This approach ensures that everything is dynamic and contained within a single formula in Excel using LAMBDA and MAKEARRAY. Let me know if you need any further assistance!

Edit in Pages

Copy

13 of 30 responsesAI-generated content may be incorrect

**20 November 2024 at 10:07**

**You said:**

Why don't you integrate RDSEED\_EXCEL within MAKEARRAY and LAMBDA without LET. You can usenesting one LAMBDA within another LAMBDA. Internal lambda will convert vertical array thrown by RDSEED\_EXCEL element wise and external will perform the arranging in cells as required by MAKEARRAY.

**Copilot said:**

To integrate RDSEED\_EXCEL within MAKEARRAY using nested LAMBDA functions, we need to ensure that the internal LAMBDA function converts the vertical array generated by RDSEED\_EXCEL into individual elements, and the external LAMBDA function arranges these elements in the desired format.

Here’s how you can achieve this:

1. **Generate the random numbers** using RDSEED\_EXCEL.
2. **Use the MAKEARRAY function** with nested LAMBDA functions to arrange these numbers.

**Step-by-Step Solution**

1. **Ensure your UDF is correctly registered and can be called from Excel**. Your UDF should look like this:

import rdrand

import xlwings as xw

@xw.func

def RDSEED\_EXCEL(min\_val, max\_val, count):

random\_numbers\_rdseed = []

for \_ in range(count):

random\_number\_rdseed = rdrand.RdSeedom().get\_bits(64) % (max\_val - min\_val + 1) + min\_val

random\_numbers\_rdseed.append(random\_number\_rdseed)

return [[random\_number\_rdseed\_1] for random\_number\_rdseed\_1 in random\_numbers\_rdseed]

# To use this UDF in Excel, you need to run the following command in your terminal:

# xlwings runpython install

1. **Use the MAKEARRAY function** with nested LAMBDA functions:

=MAKEARRAY(100, 10, LAMBDA(r, c,

LAMBDA(i,

IF(i <= COUNTA(RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)),

INDEX(RDSEED\_EXCEL(100000000000001, 999999999999999, 1000), i),

"")

)((r-1)\*10+c)

))

**Explanation:**

* **RDSEED\_EXCEL(100000000000001, 999999999999999, 1000)**: This generates 1000 random numbers between 100000000000001 and 999999999999999.
* **MAKEARRAY(100, 10, LAMBDA(r, c, ...))**: This creates a 100x10 array.
* **Nested LAMBDA functions**: The internal LAMBDA function converts the vertical array into individual elements. The external LAMBDA function arranges these elements in the desired format.
* **IF(i <= COUNTA(RDSEED\_EXCEL(...)), INDEX(RDSEED\_EXCEL(...), i), "")**: This part of the LAMBDA function checks if the current index is within the range of generated numbers. If it is, it uses INDEX to get the corresponding random number. If not, it returns an empty string ("").

This approach ensures that everything is dynamic and contained within a single formula in Excel using nested LAMBDA functions and MAKEARRAY. Let me know if you need any further assistance!

**Code**

Copy

14 of 30 responsesAI-generated content may be incorrect