In [1]:

```
"""
Created on Tue Aug 23 18:13:12 2022

@author: Brandon Botzer - btb5103


Question 1:
Perform the following actions:

Use the randn function to create an array with a dimension of 5X5, and use a
for loop to calculate the sum of all elements in the diagonal
of the array. (25 points)


Choose any three functions to apply to this array. (25 points)
"""
```

Out[1]:

```
'\nCreated on Tue Aug 23 18:13:12 2022\n\n@author: Brandon Botzer - btb5103\n\n
\nQuestion 1:\nPerform the following actions:\n\nUse the randn function to crea
te an array with a dimension of 5X5, and use a \nfor loop to calculate the sum
of all elements in the diagonal \nof the array. (25 points)\n\n\nChoose any thr
ee functions to apply to this array. (25 points)\n'
```

In [2]:

```
print("Question 1: \n")

import numpy as np
#Set the seed to 862 for comparison
np.random.seed(862)

#declare the total variable
total = 0

#build the 5x5 grid
grid = np.random.randn(5, 5)

#Run through the diagonal
for i in range(len(grid)):
    #sum the diagonal components
    total += grid[i][i]

print("Here is the grid:")
print(grid)
print("\nThe sum total of the diagonal is: " + str(total))
```

```
Question 1:

Here is the grid:
[[-1.01137922 -2.00191133  0.31880728 -3.07196693 -0.18898829]
 [-0.42043878  1.09776275 -0.01172648 -1.31461855 -0.33716974]
 [-1.19100793  0.71009524 -0.56076215 -0.43369985  1.0938274 ]
 [ 0.29792591  1.39142511  1.11695584 -0.95531248 -0.77417981]
 [ 0.68766374 -0.21208213 -0.04516003 -2.11825872 -0.00395109]]

The sum total of the diagonal is: -1.433642179422025
```

In [3]:
```
#Apply three functions to the array
print("\nNow to do apply three functions to the array...")

#determine how sum collapses the array
print("\nWhen summing the grid, 'sum' collapses the columns as seen here:")
print(sum(grid))
print("\n")
```

```

Now to do apply three functions to the array...

When summing the grid, 'sum' collapses the columns as seen here:
[-1.63723628  0.98528964  0.81811446 -7.89385653 -0.21046152]
```

In [4]:
```
#Perform the dreaded and computationally expensive transposition of the grid
print("The origional grid before transposition:")
print(grid)
print("\nNow to transpose the grid...\n")
print(np.transpose(grid))
```

```
The origional grid before transposition:
[[-1.01137922 -2.00191133  0.31880728 -3.07196693 -0.18898829]
 [-0.42043878  1.09776275 -0.01172648 -1.31461855 -0.33716974]
 [-1.19100793  0.71009524 -0.56076215 -0.43369985  1.0938274 ]
 [ 0.29792591  1.39142511  1.11695584 -0.95531248 -0.77417981]
 [ 0.68766374 -0.21208213 -0.04516003 -2.11825872 -0.00395109]]

Now to transpose the grid...

[[-1.01137922 -0.42043878 -1.19100793  0.29792591  0.68766374]
 [-2.00191133  1.09776275  0.71009524  1.39142511 -0.21208213]
 [ 0.31880728 -0.01172648 -0.56076215  1.11695584 -0.04516003]
 [-3.07196693 -1.31461855 -0.43369985 -0.95531248 -2.11825872]
 [-0.18898829 -0.33716974  1.0938274  -0.77417981 -0.00395109]]
```

In [5]:
```
#recall the shape of the array as a tuple
print("\n\nWhat is the shape of this grid?")
print(grid.shape)
```

```


What is the shape of this grid?
(5, 5)
```

In [6]:

```
"""
Question 2:
Perform the following actions:

Use x = np.random.randint(0, 1000, size = (10, 10)) to generate 10x10 array
and use a for loop to find out how many even numbers are in it. (25 points)

Randomly generate a 8x9 array from a normal distribute with
mean = 1, sigma = 0.5. Calculate the mean of elements whose indexes have
a relation of (i+j)%5 == 0  (i is row index and j is column index).


* * Submit your Python file with your results to this assignment with the
extension (.py) if you are using Spyder or your Jupyter Notebook with the
extension (.ipynb). In both cases, make sure to upload the printout of your
code file as PDF file so I can add my comments.

With that being said, you have to upload two files per assignment submission!
"""
```

Out[6]:

```
'\nQuestion 2:\nPerform the following actions:\n\nUse x = np.random.randint(0,
1000, size = (10, 10)) to generate 10x10 array \nand use a for loop to find out
how many even numbers are in it. (25 points)\n\nRandomly generate a 8x9 array f
rom a normal distribute with \nmean = 1, sigma = 0.5. Calculate the mean of ele
ments whose indexes have \na relation of (i+j)%5 == 0  (i is row index and j is
column index).\n\n\n* * Submit your Python file with your results to this assig
nment with the \nextension (.py) if you are using Spyder or your Jupyter Notebo
ok with the \nextension (.ipynb). In both cases, make sure to upload the printo
ut of your \ncode file as PDF file so I can add my comments.\n\nWith that being
said, you have to upload two files per assignment submission!\n'
```

In [7]:

```python
print("\n\nQuestion 2: \n")
#Set the random seed 862 for comparison
np.random.seed(862)

#Generate the grid for the problem
x = np.random.randint(0, 1000, size = (10, 10))

#The running count of even numbers
count = 0

#Loop through both the rows and columns
#This could have also been written in multiple lines by nesting the loops
#and running through the ranges
for i, j in ((xi, xj) for xi in range(len(x)) for xj in range(len(x))):

    #Test for even
    if x[i][j] % 2 == 0:
        #increment the count
        count += 1
print("Here is the grid in question:\n")
print(x)

print("\nThe number of even numbers in the grid is: " + str(count) +"\n\n")
```

```
Question 2:

Here is the grid in question:

[[731 940 218 172 805 677 160 857 872 435]
 [308 506  22 253  49 772 766 896 194 659]
 [550 238 405  51  75 188  16 722 190 566]
 [803 464 217 450 621 332 120 499 622 191]
 [313 623 980 816 826 230  61 545 826 585]
 [825 537 437 353 223 341 693 697 673 634]
 [927  39 355 770 967 340 997  39 333 329]
 [720 781 694 758 837 141 546 615 539 499]
 [380 546 149 100 326 119 505 977 142 568]
 [108 594 102 153 741 580 599 745 725 737]]

The number of even numbers in the grid is: 46
```

In [8]:

```
#Set the random seed 862 for comparison
np.random.seed(862)

sigma = 0.5
mean = 1

#create the 8x9 array
#using the formating from randn()
normal89 = sigma * np.random.randn(8, 9) + mean

#Calculate the mean of elements whose indexes ahve the relation
# (i+j) % 5 == 0

#delcare the total
total = 0
#declare the count for the division later
cnt = 0

print("The values to be used for the mean calculation are: \n")
```

```
The values to be used for the mean calculation are:
```

In [9]:

```
#Loop through all of the elements in the 2D array
for i, j in ((xi, xj) for xi in range(len(normal89)) for xj in range(len(normal89

    if (i+j) % 5 == 0:
        #increase the total value
        total += normal89[i][j]
        #increment the cnt
        cnt += 1
        print(normal89[i][j])
```

```
0.4943103907524756
0.7897806102837853
0.7831500771302188
0.8939589337995089
0.49746238427590805
0.5224564471544746
1.4752417754732063
1.1508036700837416
0.043190075478290235
1.6888518823379939
1.392840773911586
0.67921290365594
```

In [10]:

```
#Divide the total by the cnt to get the mean

avg = total / cnt

print("\nThe total sum of (i+j)%5 == 0 values is: " + str(total))
print("The total number of values is: " + str(cnt))

print("\nThe mean of the used values is: " + str(avg))
```

```

The total sum of (i+j)%5 == 0 values is: 10.41125992433713
The total number of values is: 12

The mean of the used values is: 0.8676049936947609
```

In [ ]: