In [1]:
```
import numpy as np
```

In [3]:
```
#1) Which of the following function is used to transpose the below matrix?
import numpy as np
a=np.array([[1,2,3,8],[3,2,6,11],[1,3,5,14]])
a=a.transpose()
a
```

Out[3]:
```
array([[ 1,  3,  1],
       [ 2,  2,  3],
       [ 3,  6,  5],
       [ 8, 11, 14]])
```

In [11]:
```
#2) Calculate the determinant of a matrix?
n_array = np.array([[15,14,18],[10,20,34],[121,3,11]])
det=np.linalg.det(n_array)
det=round(det,2)
det
```

Out[11]:
```
14806.0
```

In [19]:
```
#3) What will be the output for the inverse of the below matrix?

A = np.array([[1, 2, 3, 4],[5, -2, 1, 1],[3, 6, 10, 9],[3, 1, 90, 11]])
inv=np.linalg.inv(A)
inv
```

Out[19]:
```
array([[-0.58367093,  0.17108133,  0.25334995, -0.0105952 ],
       [-1.01776254, -0.0754129 ,  0.48395139, -0.01900904],
       [-0.08787784, -0.00467435,  0.02586476,  0.01121845],
       [ 0.97070739, -0.00155812, -0.32471175,  0.00373948]])
```

In [21]:
```
#4) Calculate the trace of a matrix?
a = np.matrix('[13, 18, 19; 11, 4, 14; 13, 15, 16]')
a=a.trace()
a
```

Out[21]:
```
matrix([[33]])
```

In [23]:
```
#5) What will be the dot product of a and b?
import numpy as np
a = [[11, 17], [10, 33]]
b = [[3, 4], [2, 1]]
c=np.dot(a,b)
c
```

Out[23]:
```
array([[67, 61],
       [96, 73]])
```

In [25]:
```
#6) Create a covariance matrix for the following data.
x = [35, 27, 32, 25, 29]
y = [28, 21, 16, 18, 23]
z = [1, 5, 7, 11, 2]
# convert it into array
data=np.array([x,y,z])
data=np.cov(data,bias=True)
data
```

Out[25]:
```
array([[ 12.64,   7.68,  -8.52],
       [  7.68,  17.36, -12.44],
       [ -8.52, -12.44,  12.96]])
```

In [27]:
```
#7) Compute the eigen values and eigen vectors for matrix A?
import numpy as np
a = np.array([[3, 3, 5], [2, 4, 6],[1, 4, 5]])
eig_value,eig_vec=np.linalg.eig(a)
print(eig_value)
print(eig_vec)
```

```
[10.86124656  1.40152556 -0.26277213]
[[-0.57406254 -0.93550764 -0.25779845]
 [-0.62741487 -0.07643661 -0.74500451]
 [-0.52612051  0.34493898  0.61523023]]
```

In [29]:
```
#8) Given a 4x4 matrix below. Find the difference between the maximum element across the columns and the minimum element
import numpy as np
arr1= np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]])

max_accorss_col=np.max(arr1,axis=0)
print("max_accross_col=",max_accorss_col)
min_accross_row=np.min(arr1,axis=1)
print("min_accross_row=",min_accross_row)
difference=max_accorss_col-min_accross_row
print("difference=",difference)
```

```
max_accross_col= [13 14 15 16]
min_accross_row= [ 1  5  9 13]
difference= [12  9  6  3]
```

In [31]:
```
#9) Given two vectors A and B, find the cross product between the two vectors.
A = np.array([5,3])
B = np.array([1,9])

c=np.cross(A,B)
c
```

Out[31]:
```
array(42)
```

In [33]:
```
#10) Given two numpy arrays a and b. Stack together these arrays along the horizontal axis. Which of the following optio
import numpy as np
a=np.array([[44,33,51],[31,45,55]])
b=np.array([[59,61,41], [71,80,99]])

result=np.hstack((a,b))
result
```

Out[33]:
```
array([[44, 33, 51, 59, 61, 41],
       [31, 45, 55, 71, 80, 99]])
```

In [37]:
```
#11) Which of the following code is correct to access the 2nd row and 3rd column from a matrix?
import numpy as np
A = np.array([[1, 4, 5, 12],[-5, 8, 9, 0],[-6, 7, 11, 19]])
result=A[1,2]
print(result)
print("A[1] =", A[1])
print("A[:,2] =",A[:,2])
```

```
9
A[1] = [-5  8  9  0]
A[:,2] = [ 5  9 11]
```

In [39]:
```
#12) Answer the following questions based on the below sample data -
data=[199.23,278.34,135.22,23,233,199.23,275,80,23,233,199.23]
#What will be the mean, median and mode ?

print("Mean=", np.mean(data))

print("median=", np.median(data))
```

```
Mean= 170.75
median= 199.23
```

In [41]:
```
from scipy import stats
data=np.array(data)
mode_value=stats.mode(data)
mode_value
```

Out[41]:
```
ModeResult(mode=199.23, count=3)
```

In [43]:
```
#13) Answer the following questions based on the below sample data -
data=[199.23,278.34,135.22,23,233,199.23,275,80,23,233,199.23]
#Calculate the lower and upper quartile for the data?

data_arr=np.array(data)
Q1=np.percentile(data_arr,25)
Q3=np.percentile(data_arr,75)
print("Lower quartile=",Q1)
print("Upper quartile=",Q3)
```

```
Lower quartile= 107.61
Upper quartile= 233.0
```

In [53]:
```
#14) Calculate the range and IQ Range?
data=[199.23,278.34,135.22,23,233,199.23,275,80,23,233,199.23]
arr=np.array(data)
Range=np.max(data)-np.min(data)
print("Range=",round((Range),2))
q1=np.percentile(arr,25)
q3=np.percentile(arr,75)
IQR=q3-q1
print("IQR=",IQR)
```

```
Range= 255.34
IQR= 125.39
```

In [55]:
```
#15) Answer the following questions based on the below sample data -
data=[199.23,278.34,135.22,23,233,199.23,275,80,23,233,199.23]
data=np.array(data)
#Which of the following options is an outlier for the sample data?

#(A) -25.47, 321.08
#(B) -10.47, 221.08
#(C) -9.47, 110.08
#(D) -85.47, 455.08

q1=np.percentile(arr,25)
q3=np.percentile(arr,75)
IQR=q3-q1

upper_bound=q3+1.5*IQR

outlier_upper=data[data>upper_bound]

print("upper_bound=",upper_bound)
print("upper_outlier=",outlier_upper)

lower_bound=q1-1.5*IQR
lower_outlier=data[data<lower_bound]
print("lower_bound=",lower_bound)
print("lower_outlier=",lower_outlier)

outliers = data[(data < lower_bound) | (data > upper_bound)]
print("outliers=",outliers)

# Option D is below lower limit and about upper bound so option D is correct ans
```

```
upper_bound= 421.08500000000004
upper_outlier= []
lower_bound= -80.47500000000001
lower_outlier= []
outliers= []
```

In [57]:
```
#16) Answer the following questions based on the below sample data -
data=[199.23,278.34,135.22,23,233,199.23,275,80,23,233,199.23]
#Calculate variance and standard deviation for the sample data.
data=np.array(data)
var=np.var(data,ddof=1)
print("var=",round((var),2))
std=np.sqrt(var)
print("std=",round((std),2))
```

```
var= 8578.52
std= 92.62
```

In [ ]: