Name : Pankajan Index No : 190428D

In [ ]:
```
for i in range(1,6):
  print(i,':',i**2)
```

```
1 : 1
2 : 4
3 : 9
4 : 16
5 : 25
```

In [ ]:
```
import sympy
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv
```

In [ ]:
```
for i in range(1,6):
  if not sympy.isprime(i):
    print(i,':',i**2)
```

```
1 : 1
4 : 16
```

In [ ]:
```
square = [i**2 for i in range(1,6) ]
square
```

Out[ ]:
```
[1, 4, 9, 16, 25]
```

In [ ]:
```
squares = [i for i in range(1,6) if not sympy.isprime(i)]
squares
```

Out[ ]:
```
[1, 4]
```

In [ ]:
```
A=np.array([[1,2],[3,4],[5,6]])
C=np.array([[7,8,9,1],[1,2,3,4]])
np.matmul(A,C)
```

Out[ ]:
```
array([[ 9, 12, 15,  9],
       [25, 32, 39, 19],
       [41, 52, 63, 29]])
```

In [ ]:
```
B=np.array([[3,2],[5,4],[3,1]])
np.multiply(A,B)
```

Out[ ]:
```
array([[ 3,  4],
       [15, 16],
       [15,  6]])
```

In [ ]:
```
ori_array = np.random.randint(10, size=(5, 7))
print(ori_array)
print(ori_array[2:5,:3])
np.size(ori_array[2:5,:3])
```

```
[[6 3 0 3 2 8 0]
 [1 0 0 9 8 6 5]
 [1 1 2 8 6 5 6]
 [5 3 0 6 2 1 4]
 [4 3 4 7 7 9 2]]
[[1 1 2]
 [5 3 0]
 [4 3 4]]
```

Out[ ]:
```
9
```

In [ ]:
```
#broadcasting allows the arithmetic calculations on different size matrixes
#basic example
a = np.array((2,3))
a*2
```

Out[ ]:
```
array([4, 6])
```

In [ ]:
```
#2nd example
a = np.array([[2,3],[3,4]])
b = np.array([4,5])
a+b
```

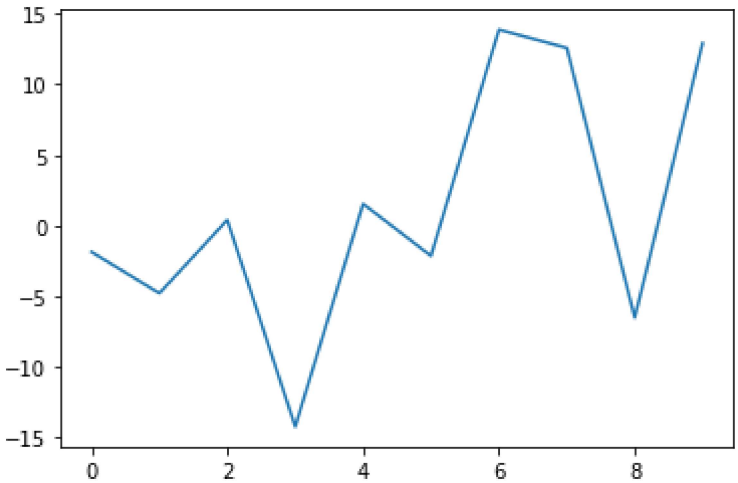
Out[ ]:
```
array([[6, 8],
       [7, 9]])
```

In [ ]:
```
#3rd example
a[:, np.newaxis] + b
```

Out[ ]:
```
array([[[6, 8]],

       [[7, 9]]])
```

In [ ]:
```
m, c = 2 , -4
N = 10
x = np . linspace (0 , N-1, N) . reshape (N, 1 )
sigma = 10
y = m*x + c + np . random . normal (0 , sigma , (N, 1 ) )
plt.plot(x,y)
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x1915c464d30>]
```



In [ ]:
```
X =np.append(np.ones((N,1)),x,axis = 1)
X
```

Out[ ]:
```
array([[1., 0.],
       [1., 1.],
       [1., 2.],
       [1., 3.],
       [1., 4.],
       [1., 5.],
       [1., 6.],
       [1., 7.],
       [1., 8.],
       [1., 9.]])
```

In [ ]:
```
Ans=np.linalg.inv(X.T@X)@X.T@y
Ans
```

Out[ ]:
```
array([[-5.98392549],
       [ 1.58981854]])
```

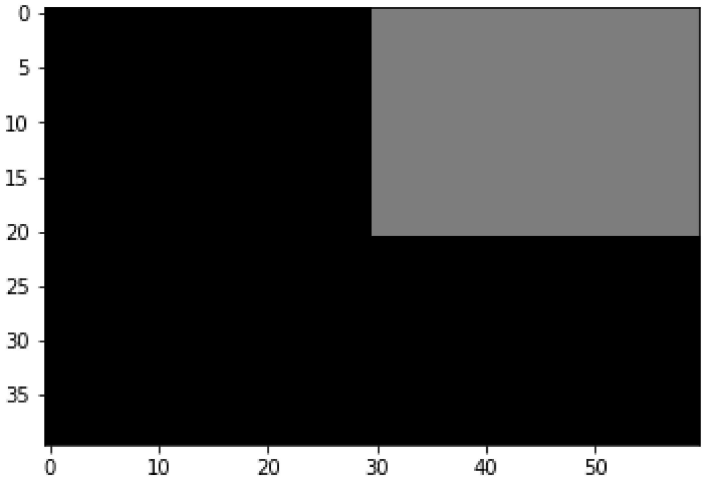
In [ ]:
```
im = cv.imread(r'./gal_gaussian.png')

cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
cv.imshow('Image',im)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:
```
blur = cv.GaussianBlur(im,(5,5),0)

cv.imshow('Image',blur)
cv.waitKey(0)
cv.destroyAllWindows()
```

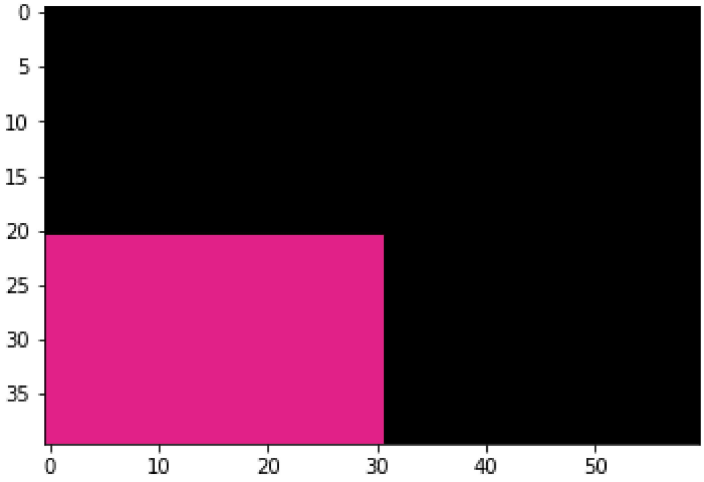
In [ ]:
```
im2 = cv.imread(r'./gal_sandp.png')
median = cv.medianBlur(im2,5)
cv.imshow('Image',median)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:
```
im3 = np.zeros((40,60),dtype=np.uint8)

im3[0:21,30:61]=125
cv.imshow('Image',im3)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:
```
fig , ax =plt.subplots()
ax.imshow(im3,cmap='gray',vmax=255,vmin=0)
plt.show()
```



In [ ]:
```
im4 = np.zeros((40,60,3),dtype=np.uint8)

im4[21:41,0:31]=[224,33,138]
#cv.imshow('Image',im4)
#cv.waitKey(0)
#cv.destroyAllWindows()
fig , ax =plt.subplots()
ax.imshow(im4,vmax=255,vmin=0)
plt.show()
```



In [ ]:
```
im5 = cv.imread(r'./tom_dark.jpg')

cv.imshow('Image',im5)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:
```
import sympy
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv


im5 = cv.imread(r'./tom_dark.jpg')

alpha=2
beta=50
```

```
result = cv.addWeighted(im5, alpha, np.zeros(im5.shape, im5.dtype), 0 , beta)
cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
cv.imshow('Image',result)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]: