NAME : S.SANJITH

ID NO : 190562G

COURSE CODE : EN2550

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

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

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

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

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

In [ ]:
```
squares=[i**2 for i in range(1,6)]
for i, i2 in enumerate(squares):
    print(i,": ",i2)
```

```
0 :  1
1 :  4
2 :  9
3 :  16
4 :  25
```

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

```
[1, 16]
```

In [ ]:
```
#Matrix Multiplication
X=np.array([[1,2],[3,4],[5,6]])
Y=np.array([[7,8,9,1],[1,2,3,4]])
C=np.matmul(X,Y)                    #np.dot(X,Y) or A@B
print(C)
```

```
[[ 9 12 15  9]
 [25 32 39 19]
 [41 52 63 29]]
```

In [ ]:
```
#element wise multiplication
A=np.array([[1,2],[3,4],[5,6]])
```

```
B=np.array([[3,2],[5,4],[3,1]])
print(A*B)
```

```
[[ 3  4]
 [15 16]
 [15  6]]
```

In [ ]:

```
rand_array = np.random.randint(0,10,(5,7))
print(rand_array[2:5,0:2])
#size of the resultant array : 2*2
```

```
[[7 7]
 [8 9]
 [9 8]]
```

In [ ]:

```
#Broadcasting Example 1
u=np.zeros((10,4))
v=np.ones((1,4))
#Incase we are trying to add u and v
x=u+v
x
```

Out[ ]:

```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [ ]:

```
#Broadcasting Example 2
u=np.array([[1,2,3],[4,5,6],[7,8,9]])
v=np.array([[1],[2],[3]])
#Incase we are trying to multiply u and v
x=u*v
print(x)

#Incase we are trying to add them up
y=u+v
print(y)
```

```
[[ 1  2  3]
 [ 8 10 12]
 [21 24 27]]
[[ 2  3  4]
 [ 6  7  8]
 [10 11 12]]
```

In [ ]:

```
#Estimating Square-root of a number
def square_estimation(number):
    precision=5
    n=0
    a=number
    #hyperbolic estimation
    while not(a<=100):
        a=a/100
```

```
        n=n+1
    h_estimate=(-190/(a+20)+10)*10**n

    #newton-estimation
    r=h_estimate
    root = 0.5 * (h_estimate + (number / h_estimate))
    while (abs(r-root)>10**(-precision)):
        r=root
        root = 0.5 * (r + (number / r))
    return root
```

In [ ]:

```
#Calculating the square-root estimations
square_estimation_vectorized = np.vectorize(square_estimation)
n_array=np.array([64,75,100,1600])
print(square_estimation_vectorized(n_array))
```

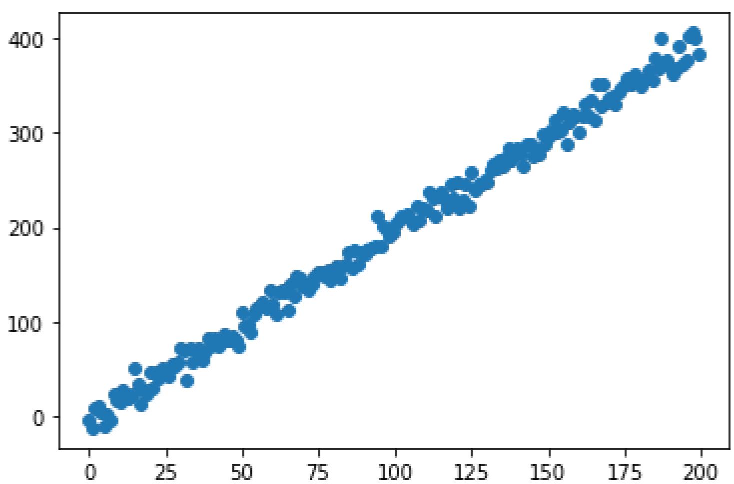
```
[ 8.          8.66025404 10.         40.        ]
```

In [ ]:

```
m, c = 2 , -4
N = 200
x = np . linspace (0 , N-1, N) . reshape (N, 1 )
sigma = 10
y = m*x + c + np . random. normal (0 , sigma , (N, 1 ) )

plt.scatter(x,y)
```

Out[ ]: <matplotlib.collections.PathCollection at 0x1ab233c6070>



In [ ]:

```
X=np.append(x,np.ones((N,1)),axis=1)
ANSWER = np.linalg.inv((X.T@X))@X.T@y
ANSWER
```

Out[ ]:

```
array([[ 2.00027072],
       [-2.94512637]])
```

In [ ]:

```
im=cv.imread(r'./images/gal_gaussian.png')
blur=cv.GaussianBlur(im,(5,5),0)

cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
```

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

In [ ]:

```
im2=cv.imread(r'./images/gal_sandp.png')
fltrd=cv.medianBlur(im2, 3)

cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
cv.imshow('Image',im2)
cv.waitKey(0)
cv.imshow('Image',fltrd)
cv.waitKey(0)
cv.destroyAllWindows()
```

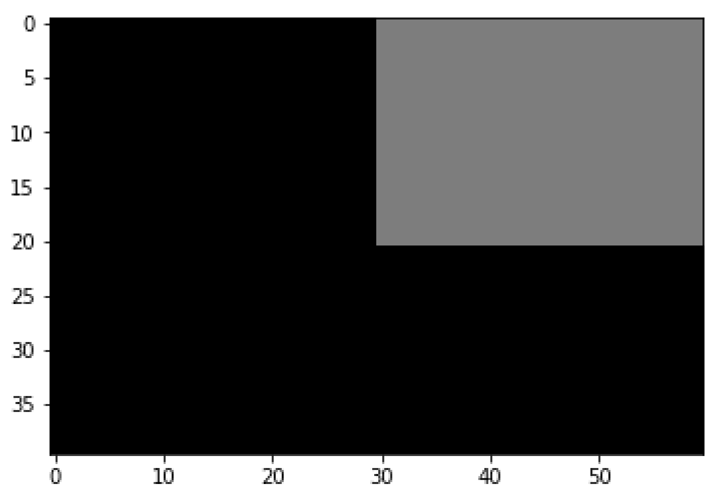
In [ ]:

```
im3=np.zeros((40,60),dtype=np.uint8)
im3[0:21, 30:61]=125

cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
cv.imshow('Image',im3)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:

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



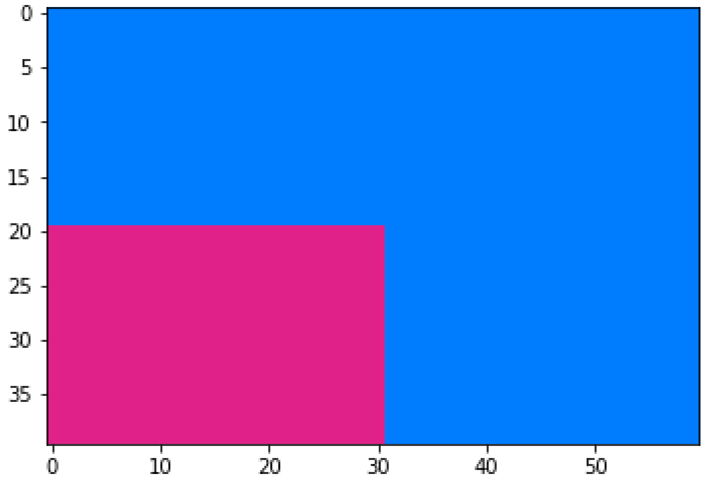
In [ ]:

```
im4=np.zeros((40,60,3),dtype=np.uint8)
im4[:]=(0,124,255)
im4[20:41,0:31]=(224, 33, 138)

cv.namedWindow('Image',cv.WINDOW_AUTOSIZE)
cv.imshow('Image',im4)
cv.waitKey(0)
cv.destroyAllWindows()
```

In [ ]:

```
fig, ax =plt.subplots()
ax.imshow(im4)
plt.show()
```



In [ ]:

```
im5=cv.imread(r'./images/tom_dark.png')
new_image=im5+60


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