```
import random
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
```

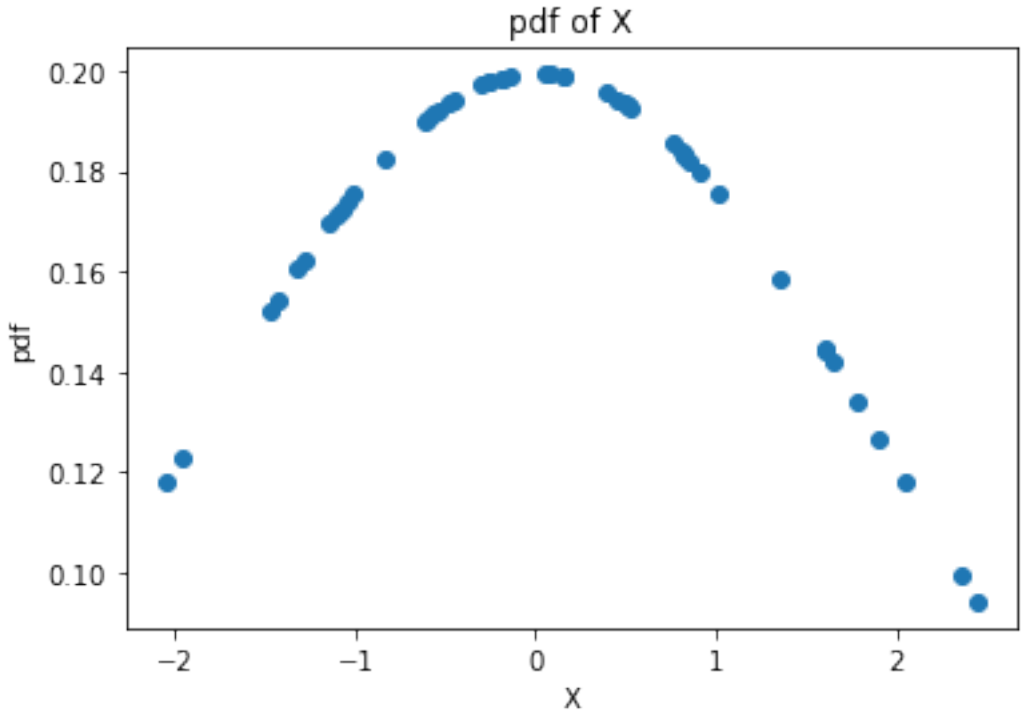
Exercise-A

```
t=[0,1,2,3,4]

x=np.random.normal(0,1,size=(len(t),10))
y=np.zeros((len(t),10))
z=np.zeros((len(t),10))

for i in range(1,len(t)):
    for j in range(x.shape[1]):
        y[i][j]=0.5*(x[i][j]+x[i-1][j])
        z[i][j]=x[i][j]-x[i-1][j]

crossCovarianceXY=np.zeros((5,5))
crossCovarianceYZ=np.zeros((5,5))
crossCovarianceZX=np.zeros((5,5))

for i in range(crossCovarianceXY.shape[0]):
    for j in range(crossCovarianceXY.shape[1]):
        crossCovarianceXY[i][j]=np.cov(x[j],y[i])[0][1]
        crossCovarianceYZ[i][j]=np.cov(y[j],z[i])[0][1]
        crossCovarianceZX[i][j]=np.cov(x[j],z[i])[0][1]


print("The cross covariance of x and y is ","\n",crossCovarianceXY)
print("The cross covariance of y and z is ","\n",crossCovarianceYZ)
print("The cross covariance of z and x is ","\n",crossCovarianceZX)
```

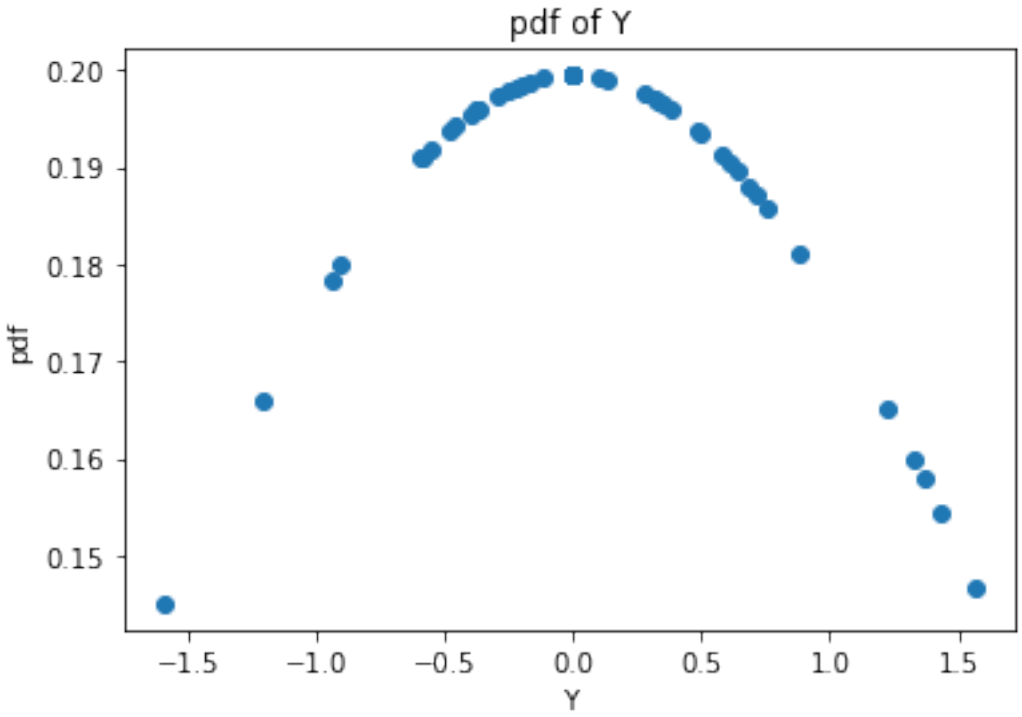
```
The cross covariance of x and y is
 [[ 0.          0.          0.          0.          0.        ]
 [ 0.86186734  0.57542559  0.22125282 -0.47908727 -0.14010911]
 [ 0.07535547  0.72132294  0.37103297 -0.12011707 -0.15950085]
 [-0.391291    0.13345655  0.11745934  0.51527293  0.02270863]
 [-0.57891724 -0.04027914 -0.23933878  0.77732034  0.52590978]]
The cross covariance of y and z is
 [[ 0.          0.          0.          0.          0.        ]
 [ 0.         -0.28644175  0.64596747  0.52474755  0.5386381 ]
 [ 0.         -0.35417277 -0.35028997 -0.01599721 -0.19905964]
 [ 0.         -0.70034009 -0.49115004  0.39781359  1.01665912]
 [ 0.          0.33897816 -0.03938377 -0.49256429 -0.25141056]]
The cross covariance of z and x is
 [[ 0.          0.          0.          0.          0.        ]
 [-1.9274509   1.35456739 -0.06263245  1.11212756 -0.03485136]
 [ 0.35442715 -1.06277269  0.36219274 -0.39418716 -0.00393212]
```

```
 [-1.2877201  -0.11296009 -0.86934     1.66496717  0.36835108]
 [ 0.91246762 -0.23451129  0.15574375 -1.14087234  0.63805122]]
```

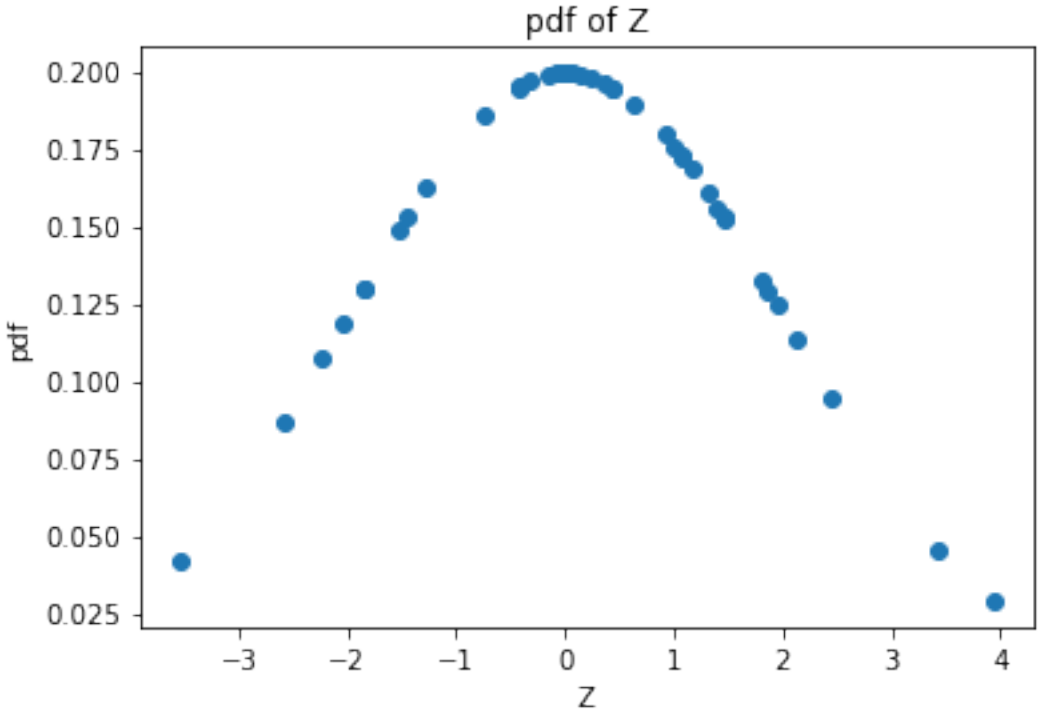
```
y1=norm.pdf(x,loc=0,scale=2)
y2=norm.pdf(y,loc=0,scale=2)
y3=norm.pdf(z,loc=0,scale=2)

plt.scatter(x,y1)
plt.title("pdf of X")
plt.xlabel("X")
plt.ylabel("pdf")
plt.show()
```




```
plt.scatter(y,y2)
plt.title("pdf of Y")
plt.xlabel("Y")
plt.ylabel("pdf")
plt.show()
```

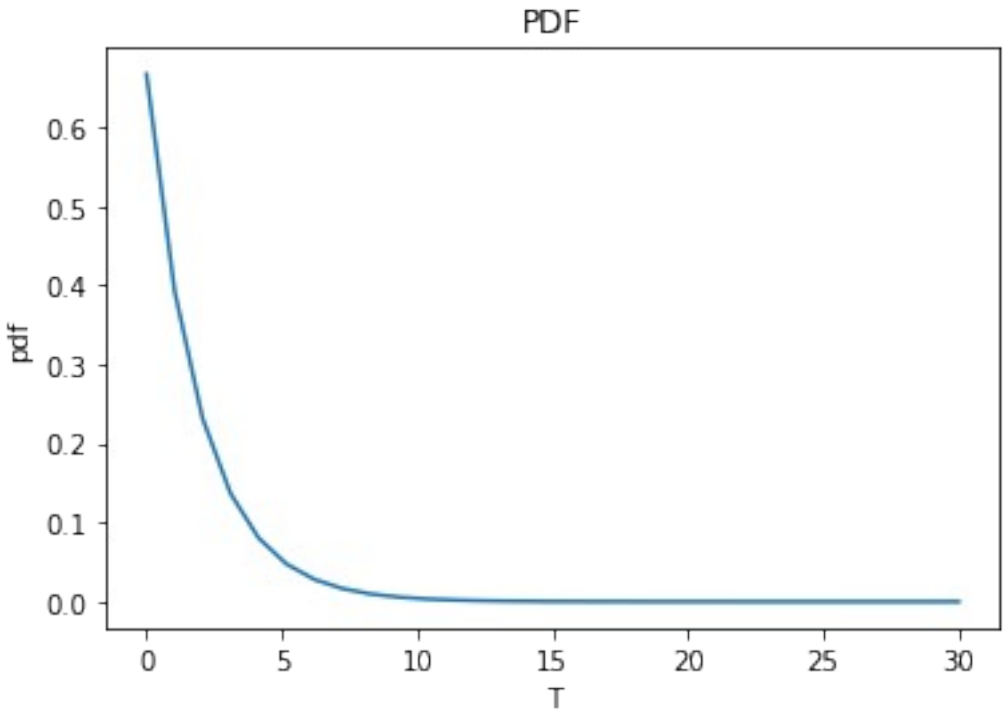



```
plt.scatter(z,y3)
plt.title("pdf of Z")
plt.xlabel("Z")
plt.ylabel("pdf")
plt.show()
```



Exercise B

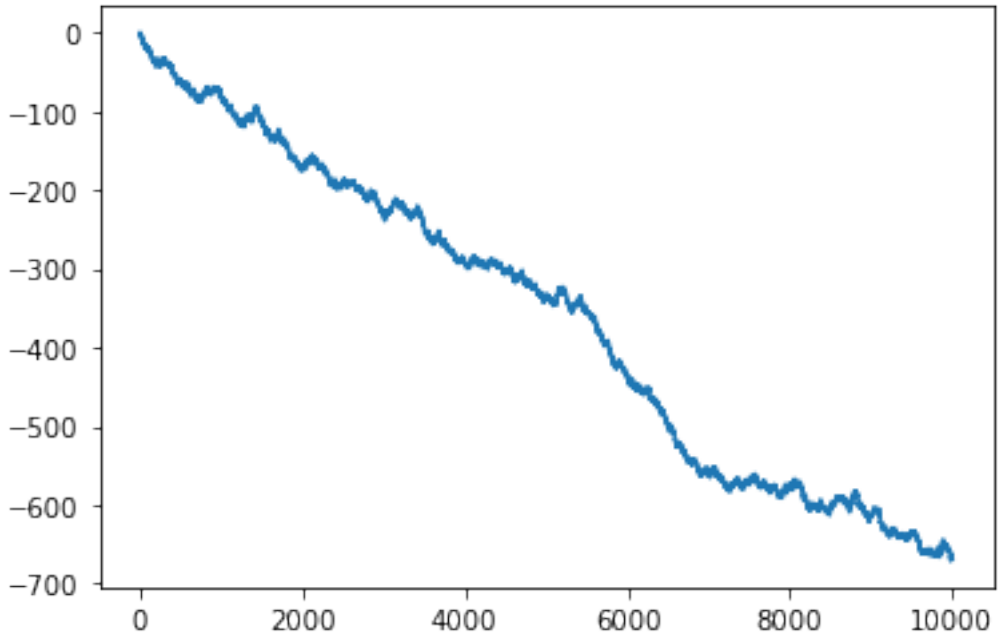
```
tau=np.linspace(0,30,30)

p=0.4
pdf=[]

for i in tau:
    pdf.append(p*(1-p)**(i-1))

plt.plot(tau,pdf)
plt.title("PDF")
plt.xlabel("T")
plt.ylabel("pdf")
plt.show()
```



Exercise-C

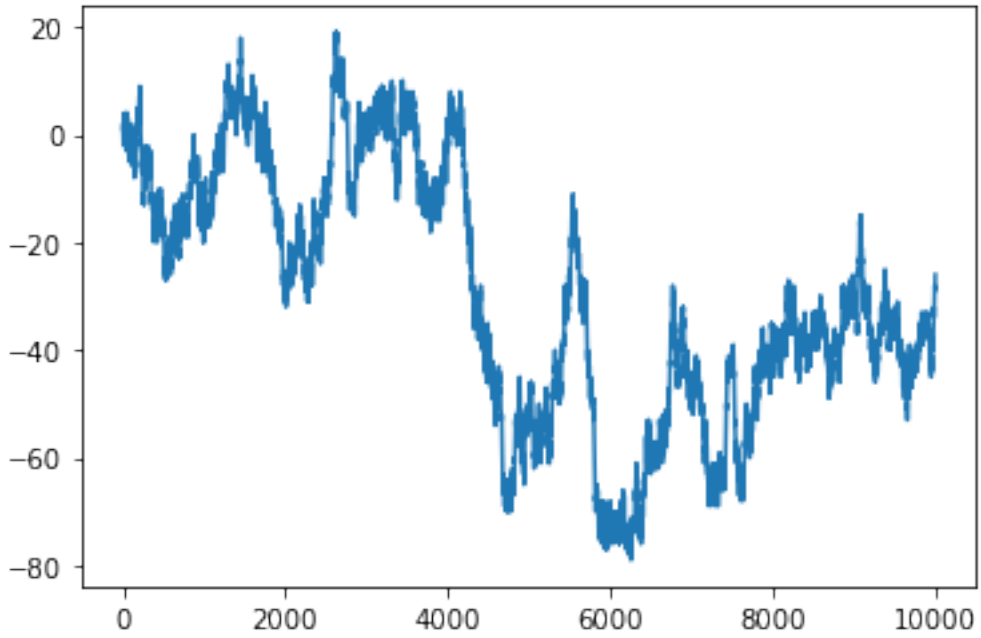
```
result=[]
p=0.46
for i in range(100):
    if i<100*p:
        result.append(1)
```

```
    else:
        result.append(-1)

t=np.linspace(0,10000,10000)

total=0
xt=[]
yt=[]

for i in t:
    x=random.choice(result)
    xt.append(x)
    total+=x
    yt.append(total)

plt.plot(t,yt)
plt.show()
```



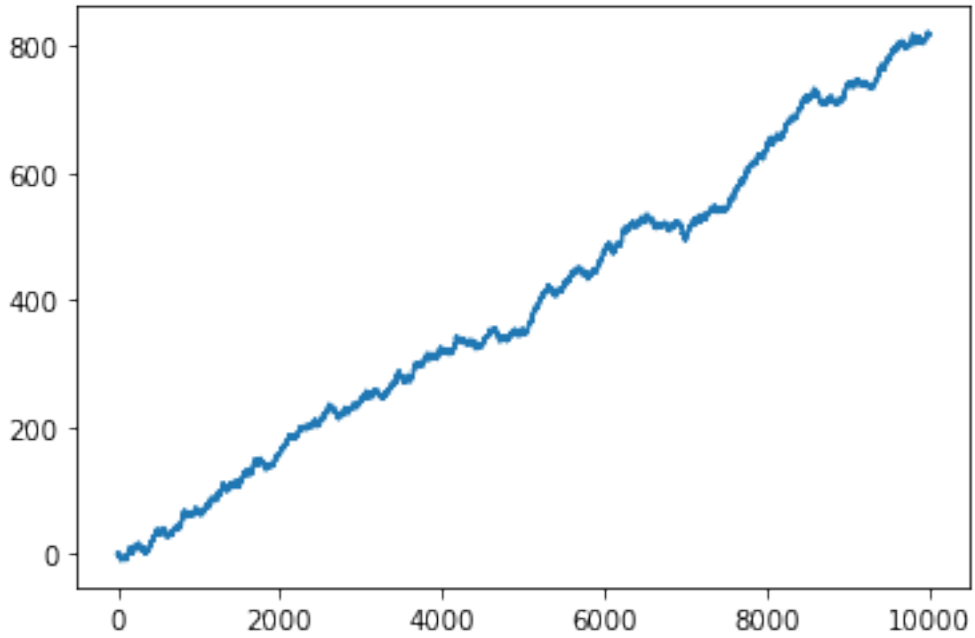

```
result=[]
p=0.5
for i in range(100):
    if i<100*p:
        result.append(1)
    else:
        result.append(-1)

t=np.linspace(0,10000,10000)

total=0
xt=[]
```

```
yt=[]

for i in t:
    x=random.choice(result)
    xt.append(x)
    total+=x
    yt.append(total)

plt.plot(t,yt)
plt.show()
```



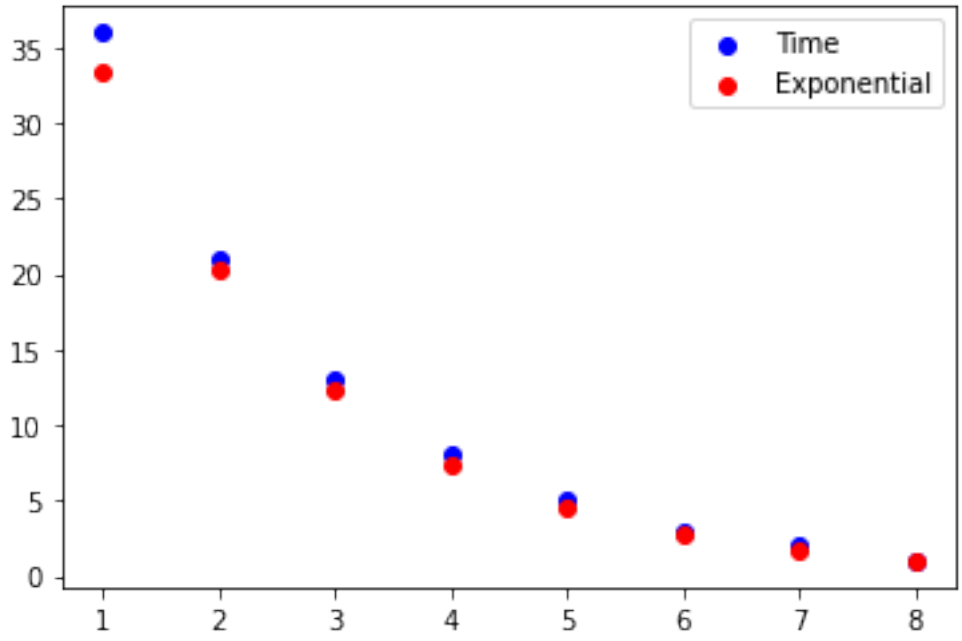

```
result=[]
p=0.54
for i in range(100):
    if i<100*p:
        result.append(1)
    else:
        result.append(-1)

t=np.linspace(0,10000,10000)

total=0
xt=[]
yt=[]

for i in t:
    x=random.choice(result)
    xt.append(x)
    total+=x
    yt.append(total)
```

```
plt.plot(t,yt)
plt.show()
```



Exercise-D

```
def exponential(a,b):
    return np.exp(-1*a*b)

x=np.arange(0,10)
y=[1/exponential(0.5,i) for i in x]
y1=[int(j) for j in y]

time=[y1[i+1]-y1[i] for i in range(len(x)-2,0,-1)]
u=np.arange(1,9)

plt.scatter(u,time,label='Time',color='blue')
plt.scatter(u,55*exponential(0.5,u),label='Exponential',color='red')
plt.legend()
plt.show()
```


Time
Exponential
35
30
25
20
15
10
5
0
1
2
3
4
5
6
7
8