Random Walk in 1-D

Plot the displacements from origin against time

```
import numpy as np
import matplotlib.pyplot as plt
L = np.random.choice( (-1, +1), 10, p = (0.6,0.4) )
X = np.cumsum(L)
T = np.linspace(1, 10, 10)
plt.grid()
plt.plot(T, X)
plt.axhline()
plt.show()
```

Find mean x, mean x^2 and σ, after 10 steps

```
import numpy as np
sum = 0
sum2 = 0
for i in range(100) :
    L = np.random.choice( (-1, +1), 10, p = (0.2,0.8) )
    x = np.sum(L)
    sum = sum + x
    sum2 = sum2 + x*x
mean = sum/100
mean2 = sum2/100
sigma = np.sqrt(mean2 - (mean)**2)
print ("Mean = ", mean)
print("Mean2 = ", mean2)
print('sigma = ', sigma)
```

Plot r.m.s. displacement against number of steps

```
import numpy as np
import matplotlib.pyplot as plt
N = np.linspace(1, 25, 25)
Xrms = []
for n in N :
    n = int (n)
    sn = 0.0
    for i in range(500) :
        Ln = np.random.choice( (-1, +1), n)
        xn = np.sum(Ln)
        sn = sn + xn*xn
    xrms = np.sqrt(sn/500)
    Xrms.append(xrms)
plt.plot (N, Xrms)
plt.show()
```

Random Walk on a 2-D Square Lattice

Plot the positions after different steps

```
import numpy as np
import matplotlib.pyplot as plt
Th = np.random.choice( (0, np.pi/2, np.pi, 3*np.pi/2), 10)
X = np.cos(Th)
Y = np.sin(Th)
X1 = np.cumsum(X)
Y1 = np.cumsum(Y)
plt.grid()
plt.plot(X1, Y1)
plt.axhline()
plt.axvline()
plt.show()
```

Find r.m.s distance from the origin after 10 steps

```
import numpy as np
sum = 0.0
for i in range(100) :
    Th = np.random.choice( (0, np.pi/2, np.pi, 3*np.pi/2), 10)
    X = np.cos(Th)
    Y = np.sin(Th)
    x1 = np.sum(X)
    y1 = np.sum(Y)
    sum = sum + x1*x1 + y1*y1
rms = np.sqrt(sum/100)
print ("rms  = ", rms)
```

Plot r.m.s. displacement against number of steps

```
import numpy as np
import matplotlib.pyplot as plt
N = np.linspace(1, 25, 25)
RMS = []
for n in N :
   n = int (n)
   sn = 0.0
   for i in range(500) :
       Th = np.random.choice( (0, np.pi/2, np.pi, 3*np.pi/2), n)
       X = np.cos(Th)
       Y = np.sin(Th)
       xn = np.sum(X)
       yn = np.sum(Y)
       sn = sn + xn*xn +yn*yn
   rms = np.sqrt(sn/500)
```

```
    RMS.append(rms)
plt.plot (N, RMS)
plt.show()
```

Random Walk on a 2-D Plane (**Not in Syllabus**)

Plot the positions after different steps

```
import numpy as np
import matplotlib.pyplot as plt
Th = 2 * np.pi * np.random.rand(10)
X = np.cos(Th)
Y = np.sin(Th)
X1 = np.cumsum(X)
Y1 = np.cumsum(Y)
plt.grid()
plt.plot(X1, Y1)
plt.axhline()
plt.axvline()
plt.show()
```

Monte Carlo Integration

Integrate sin (x) from 0 to π

```
import numpy as np
def f(x) :
    return np.sin(x)
X1 = np.linspace(0, np.pi, 100)
h = np.max(f(X1))
a = 0.0
b = np.pi
box = (b - a) * h
N = int(input(" No. of points"))
M = np.random.random((N, 2))
X, Y = M[: , 0], M[: , 1]
X = a + (b - a) * X
Y = h*Y
Y1 = Y[Y<= f(X)]
integ = Y1.size/N * box
print("Integral = ", integ)
```

Plotting Fermi Distribution Function f(E) at Different Temperatures

Information : Fermi energy for Na = 3.24 eV, for K = 2.12 eV

$K = 8.6 \times 10^{-5}$ eV/K

KT at 300 K = 0.259 eV

Program :

```
import matplotlib.pyplot as plt
import numpy as np
K = 8.6 *10**(-5)
T1 = float(input("T1 = "))
T2 = float (input ("T2 = "))
EF = 4.0

def f(E, T) :
    x  = (E - EF)/(K*T)
    return 1/(np.exp(x) + 1)

E = np.linspace(3.0, 6.0, 100)
plt.plot(E, f(E, T1), "r")
plt.plot (E, f(E, T2), "b")
plt.show()
```

Plotting Einstein's Sp. Heat against x = hn/KT

Information :

Molar Einstein Specific heat $C_v = 3NK \times (h\nu/KT)^2 \times e^{h\nu/KT} / (e^{h\nu/KT} - 1)^2$

$h\nu/K$ is called the Einstein Temperature $T_E \Rightarrow h\nu/KT = T_E/T$

Molar Dulong-Petit Specific heat $C_{v1} = 3NK$

Program :

```
import matplotlib.pyplot as plt
import numpy as np

# 3NK = 3R = 6.0
# In units of Einstein Temp :
TE = 1
T = np.linspace(0.01, 2.0, 100)

def Cv(T) :
    X  = TE/T
    Y = np.exp(X)
    return 6*X**2*Y/(Y - 1)**2
Cv1 = np.zeros(100)      # generates hundred zeros : (0.0, 0,0, . . . )
Cv1 = Cv1 + 6.0          # generates the array : (6.0, 6.0, . . . )
plt.plot(T, Cv(T), "r")
plt.plot(T, Cv1, "b")
plt.show()
```