```
import pandas as pd
import numpy as np
from numpy import *
from math import sqrt
from mpl_toolkits.mplot3d import Axes3D

import matplotlib.pyplot as plt

# the calculation of electric charges of quark shells "u" and "d"

# Array input according to the matrix proposed by the author
A1 = array ([[2.0 , 1.0, 1.0, 1.0, 1.0, 0.0], [0.0, 1.0, 0.0, 0.0, 0.0, 1.0],
            [0.0, 0.0, 1.0, 0.0, 0.0, 0.0], [1.0, 1.0, 0.0, 2.0, 1.0, 1.0],
            [0.0, 0.0, 1.0, 0.0, 1.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 1.0]])

# b1 - The values charges on shells in proton, neutron obtained by
#Robert Hofstadter the Nobel laureate
b1 = array ([0.35, 0.5, 0.15, 0.25, -0.5, 0.15])

x1 = linalg . solve (A1, b1)

qvark = list(x1)

data1 = {'index': ['uq1', 'uq2', 'uq3', 'dq1', 'dq2', 'dq3'],
        'Qe': [ qvark[0], qvark[1],  qvark[2],  qvark[3],  qvark[4],  qvark[5]]}

df1 = pd.DataFrame.from_dict(data1)

df1.to_html()

# the calculation of mass of quark shells "u" and "d"

# Array input according to the matrix proposed by the author
A2 = array ([[2.0 , 2.0, 2.0, 1.0, 1.0, 1.0], [1.0, 1.0, 1.0, 2.0, 2.0, 2.0],
            [0.0, 1.0, -0.14265589786296212, 0.0, 0.0, 0.0],
            [0.0, 0.0, 0.0, 0.0, 1.0, -0.14265589786296212],
             [1.0, 0.0, 0.0, 0.0, 0.0, 0.0],
            [0.0, 0.0, 0.0, 1.0, 0.0, 0.0]])

b2 = array ([1.672621898E-27, 1.674927471E-27, 0.0, 0.0,
            0.35794108617E-27, 0.85303716798E-27])
x2 = linalg . solve (A2, b2)

qvarkm = list(x2)

data2 = {'index': ['um1', 'um2', 'um3', 'dm1', 'dm2', 'dm3'],
        'mass': [ qvarkm[0], qvarkm[1], qvarkm[2], qvarkm[3], qvarkm[4], qvarkm[5]]}

uq1 = qvark[0]
uq2 = qvark[1]
uq3 = qvark[2]
dq1 = qvark[3]
dq2 = qvark[4]
dq3 = qvark[5]

# the calculation of volume of quark shells "u" and "d"
# the calculation of the volume by shells is made taking into account
# the assumption that the volume is proportional to the ratio of charges

# V = 4/3πR**3
π = 3.14159265358979
```

```
# quark radius  "− (0.47 · 10E−16 cm)2 < RE2 < (0.43 · 10E−16 cm)2"
# data from scientific article, RE2 - quark radius squared
# square centimeters converted to square meters
# a negative value is regarded as being in a different dimension

# quark volume applied to the quark nucleus
# Vu for 0.2

Vua = 0.47E-20 * sqrt(0.47E-20) * 4/3 * π

# Vd for 0.1
Vub = 0.43E-20 * sqrt(0.43E-20) * 4/3 * π

uv1 = Vua
dv1 = Vub

uv2 = uv1/uq1 * uq2
uv3 = uv1/uq1 * uq3
dv2 = dv1/dq1 * dq2
dv3 = dv1/dq1 * dq3

# Calculation of the amount of charge on the shells
Qe = 1.602176620898e-19
uq11 = Qe * qvark[0]
uq21 = Qe * qvark[1]
uq31 = Qe * qvark[2]
dq11 = Qe * qvark[3]
dq21 = Qe * qvark[4]
dq31 = Qe * qvark[5]

# data entry for quarks u and d

um1 = qvarkm[0]
um2 = qvarkm[1]
um3 = qvarkm[2]
dm1 = qvarkm[3]
dm2 = qvarkm[4]
dm3 = qvarkm[5]

data = {'Index "u"': ['uq11', 'um1', 'uv1', 'uq21', 'um2', 'uv2', 'uq31', 'um3',
                      'uv3'],
         'Value "u"': [ uq11, um1, uv1, uq21, um2, uv2, uq31, um3, uv3],
        'Index "d"': ['dq11', 'dm1', 'dv1', 'dq21', 'dm2', 'dv2','dq31', 'dm3', 'dv3'],
        'Value "d"': [ dq11, dm1, dv1, dq21, dm2, dv2, dq31, dm3, dv3]}

df = pd.DataFrame.from_dict(data)

df.to_html()

# Quark data output in the form of a table

# calculation for proton and neutron
data2 = {'"p" index': ['pq1', 'pm1', 'pv1', 'pq2', 'pm2', 'pv2', 'pq3', 'pm3',
                       'pv3', 'pq4', 'pm4', 'pv4', 'pq5', 'pm5', 'pv5'],
         'Value "p"': [ uq11, um1, uv1, uq21+dq11, um2+dm1, uv2+dv1, uq31+dq21+uq11,
um3+dm2+um1, uv3+dv2+uv1,
                        dq31+uq21, dm3+um2, dv3+uv2, uq31, um3, uv3],
        '"n" index': ['nq1', 'nm1', 'nv1', 'nq2', 'nm2', 'nv2','nq3', 'nm3',
                                'nv3', 'nq4', 'nm4', 'nv4', 'nq5', 'nm5', 'nv5'],
        'Value "n"': [ dq11, dm1, dv1, dq21+uq11, dm2+um1, dv2+uv1, dq31+uq21+dq11,
dm3+um2+dm1, dv3+uv2+dv1,
```

```
                    uq31+dq21, um3+dm2, uv3+dv2, dq31, dm3, dv3]}

df2 = pd.DataFrame.from_dict(data2)

df2.to_html()

# Distribution by shells in proton and neutron

n = 5

Xp = np.array([uq11, uq21+dq11, uq31+dq21+uq11, dq31+uq21, uq31])
Yp = np.array([uv1, uv2+dv1, uv3+dv2+uv1,  dv3+uv2, uv3])
Zp = np.array([um1, um2+dm1, um3+dm2+um1,dm3+um2, um3])

Xn = np.array([dq11,dq21+uq11, dq31+uq21+dq11, uq31+dq21,dq31])
Yn = np.array([dv1, dv2+uv1, dv3+uv2+dv1, uv3+dv2, dv3])
Zn = np.array([dm1, dm2+um1, dm3+um2+dm1, um3+dm2,  dm3])

fig = plt.figure(figsize=(14.,14.))
ax = fig.add_subplot(111, projection='3d')

plt.scatter(Xp,Yp,Zp)
plt.scatter(Xn,Yn,Zn)

xs = (n, -10e-20, 10e-20)
ys = (n, -2e-30,10e-30)
zs = (n, -6e-28,10e-28)

scat = ax.scatter(Xp, Yp, Zp, c=Zp.flatten(), alpha=1, s = 675)
fig.colorbar(scat, shrink=0.5, aspect=5)

scat = ax.scatter(Xn, Yn, Zn, c=Zn.flatten(), alpha=0.7, s = 875, marker='D')

plt.tick_params(axis='both', which='major', labelsize=16)

ax.set_xlabel('\n \n The quantity charge shell \n in Coulomb x E-19', fontsize = 15)
ax.set_ylabel('\n \n Shell volume in\n cbm*E-30', fontsize = 15)
ax.set_zlabel('\n Mass in \n kg. x E-28', fontsize = 15)

ax.text2D(0.2, 0.95, "Distribution by shells in proton and neutron (rhombus)",
          transform=ax.transAxes, fontsize = 20)

import warnings
warnings.filterwarnings("ignore", category=RuntimeWarning)

print("Values of quarks 'u' and 'd' by \n"
      "shells in Qe (electron charges) \n")

print(df1, "\n")

print("   ", 'Values of quarks "u" and "d" by shells \n')
print(df, "\n")


print('Values of proton and neutron by shells, original condition \n')
print(df2, "\n")

plt.show()
```