```
# coding: utf-8

# # 1-D "Molecular dynamics code" for two particles attached by spring.
#

# # Velocity Verlet Algorithm.

# In[83]:


def new_pos1(X_new,X_old,V_old,V_new,F_old,m,t):                                #Algorithm
For new position
    X_new = X_old + (t)*V_old + ((t**2)*F_old)/(2*m)
    return(X_new)

def new_pos2(X_new,X_old,V_old,V_new,F_old,m,t):
    X_new = X_old + (t)*V_old - ((t**2)*F_old)/(2*m)
    return(X_new)

def new_force(X1_new,X2_new,X0,k):                                              #Algorithm
for new force
    new = X2_new - X1_new
    new = new - X0
    new = k*new
    return(new)

def new_vel1(F_old,F_new,V_old,V_new,t,m):                                      #Algorithm
for new velocity
    V_new = (F_old + F_new)
    V_new = (V_new * t)/(2*m)
    V_new = V_old + V_new
    return(V_new)
def new_vel2(F_old,F_new,V_old,V_new,t,m):
    V_new = (F_old + F_new)
    V_new = (V_new * t)/(2*m)
    V_new = V_old - V_new
    return(V_new)


# # Energy Calculation

# In[84]:


def KE(m,v):                                                                    #Kinetic
Energy Calculation
    KE = (m*(v**2))/2
    return(KE)

def PE(k,x):                                                                    #Potential
Energy Calculation
    PE = (k*(x**2))/2
    return(PE)

def TE(U,K):                                                                    #Total
Energy Calculation
    TE = U+K
    return(TE)



# #  Components of the simulation.

# In[87]:


import pandas as pd
import matplotlib.pyplot as plt
                                                                                #Variables
m1 = 50                                                                         #Mass[unit
= kg]
m2 = 50
```

```
66
67  t = 0.01                                                    #Time step
    of Integration[unit = s]
68  k = float(50)                                               #Spring
    Constant[unit = N/m]
69  X0 = float(10)
    #Equilibrium Position[unit = m]
70  T = float()                                                 #Total time
    of experiment[unit = s]
71
72  X1_old = float(-10)
    #Positions[unit = m]
73  X1_new = float()
74
75  X2_old = float(10)
76  X2_new = float()
77
78  F_old = new_force(X1_old,X2_old,X0,k)                       #Force[unit
    = N]
79  F_new = float()
80
81  V1_old = float(0)
    #Velocity[unit = m/s]
82  V1_new = float()
83
84  V2_old = float(0)
85  V2_new = float()
86
87  s=0
88  data = []
89
90  while s<10000:                                              #Total
    steps = 10000 [trial 1 = 100s/trial 2 =100s/trial 3 =10s]
91
92      X1_new = new_pos1(X1_new,X1_old,V1_old,V1_new,F_old,m1,t)        #New position
93      X2_new = new_pos2(X2_new,X2_old,V2_old,V2_new,F_old,m2,t)
94
95      F_new = new_force(X1_new,X2_new,X0,k)                   #New Force
96
97      V1_new = new_vel1(F_old,F_new,V1_old,V1_new,t,m1)       #New velocity
98      V2_new = new_vel2(F_old,F_new,V2_old,V2_new,t,m2)
99
100     T = T + t                                               #New time
101
102     PE1 = PE(k,X1_new)
        #Energy[unit = J]
103     KE1 = KE(m1,V1_new)
104     TE1 = TE(PE1,KE1)
105
106     PE2 = PE(k,X2_new)
107     KE2 = KE(m2,V2_new)
108     TE2 = TE(PE2,KE2)
109
110     list = [X1_new,X2_new,V1_new,V2_new,F_new,T,PE1,KE1,TE1,PE2,KE2,TE2]
111     data.append(list)
112
113     X1_old = X1_new
114     X2_old = X2_new
115     F_old = F_new
116     V1_old = V1_new
117     V2_old = V2_new
118
119     s +=1
120 df = pd.DataFrame(data,
    columns=['X1_new','X2_new','V1_new','V2_new','F_new','T','PE1','KE1','TE1','PE2','KE2'
    ,'TE2'])
121
122
123 # # Data Visualization.
124
125 # In[90]:
126
```

```
x11points = df.loc[:,"X1_new"]
x12points = df.loc[:,"X2_new"]

x21points = df.loc[:,"V1_new"]
x22points = df.loc[:,"V2_new"]

x31points = df.loc[:,"PE1"]
x32points = df.loc[:,"PE2"]

x41points = df.loc[:,"KE1"]
x42points = df.loc[:,"KE2"]

x51points = df.loc[:,"TE1"]
x52points = df.loc[:,"TE2"]

ypoints = df.loc[:,"T"]

plt.rcParams["figure.figsize"] = (15,20)
plt.figure()

plt.subplot(5,1,1)
plt.plot(ypoints, x11points,'r',label="Particle-1")
plt.plot(ypoints, x12points,'m',label="Particle-2")
plt.ylabel('Position (m)')
plt.title('Position, Velocity &Energy as a Function of Time')
plt.legend()

plt.subplot(5,1,2)
plt.plot(ypoints, x21points,'g',label="Particle-1")
plt.plot(ypoints, x22points,'y',label="Particle-2")
plt.ylabel('Velocity (m/s)')
plt.legend()

plt.subplot(5,1,3)
plt.plot(ypoints, x31points,'k',label="Particle-1")
plt.plot(ypoints, x32points,'m',label="Particle-2")
plt.ylabel('Potential Energy (J)')
plt.legend()

plt.subplot(5,1,4)
plt.plot(ypoints, x41points,'b',label="Particle-1")
plt.plot(ypoints, x42points,'c',label="Particle-2")
plt.ylabel('Kinetic Energy (J)')
plt.legend()

plt.subplot(5,1,5)
plt.plot(ypoints, x51points,'k',label="Particle-1")
plt.plot(ypoints, x52points,'y',label="Particle-2")
plt.ylabel('Total Energy (J)')
plt.legend()


# In[ ]:


#Author: Kalith M Ismail.
#Assignment of course: Mathematical Modelling of Biological Process and system.
#Organization: NRNU MEPhI___PhysBIo___Moscow__Russian Federation.
#Date: 19/09/2021.
#Mentor: Prof.Dr.Ivanov Dmitry.


```