```
 1: import numpy as np
 2: from matplotlib import animation
 3: import pylab
 4:
 5:
 6: def RungeKutta(f1, f2, f3, f4, a_init, b_init, c_init, d_init, t0, tf, h=0.1):
 7:     A = []
 8:     B = []
 9:     C = []
10:     D = []
11:     T = np.arange(t0, tf, h)
12:     a = a_init
13:     b = b_init
14:     c = c_init
15:     d = d_init
16:     for t in T:
17:         A.append(a)
18:         B.append(b)
19:         C.append(c)
20:         D.append(d)
21:         k1 = h * f1(a, b, c, d, t)
22:         l1 = h * f2(a, b, c, d, t)
23:         m1 = h * f3(a, b, c, d, t)
24:         n1 = h * f4(a, b, c, d, t)
25:         k2 = h * f1(a + k1 / 2, b + l1 / 2, c + m1 / 2, d + n1 / 2, t + h / 2)
26:         l2 = h * f2(a + k1 / 2, b + l1 / 2, c + m1 / 2, d + n1 / 2, t + h / 2)
27:         m2 = h * f3(a + k1 / 2, b + l1 / 2, c + m1 / 2, d + n1 / 2, t + h / 2)
28:         n2 = h * f4(a + k1 / 2, b + l1 / 2, c + m1 / 2, d + n1 / 2, t + h / 2)
29:         k3 = h * f1(a + k2 / 2, b + l2 / 2, c + m2 / 2, d + n2 / 2, t + h / 2)
30:         l3 = h * f2(a + k2 / 2, b + l2 / 2, c + m2 / 2, d + n2 / 2, t + h / 2)
31:         m3 = h * f3(a + k2 / 2, b + l2 / 2, c + m2 / 2, d + n2 / 2, t + h / 2)
32:         n3 = h * f4(a + k2 / 2, b + l2 / 2, c + m2 / 2, d + n2 / 2, t + h / 2)
33:         k4 = h * f1(a + k3, b + l3, c + m3, d + n3, t + h)
34:         l4 = h * f2(a + k3, b + l3, c + m3, d + n3, t + h)
35:         m4 = h * f3(a + k3, b + l3, c + m3, d + n3, t + h)
36:         n4 = h * f4(a + k3, b + l3, c + m3, d + n3, t + h)
37:         a += (k1 + 2 * k2 + 2 * k3 + k4) / 6
38:         b += (l1 + 2 * l2 + 2 * l3 + l4) / 6
39:         c += (m1 + 2 * m2 + 2 * m3 + m4) / 6
40:         d += (n1 + 2 * n2 + 2 * n3 + n4) / 6
41:     return T, A, B, C, D
42:
43:
44: def main():
45:     MsG = 4 * np.pi**2
46:     f1 = lambda x, y, vx, vy, t: vx
47:     f2 = lambda x, y, vx, vy, t: vy
48:     f3 = lambda x, y, vx, vy, t: -MsG * x / pow(x**2 + y**2, 3 / 2)
49:     f4 = lambda x, y, vx, vy, t: -MsG * y / pow(x**2 + y**2, 3 / 2)
50:     T, X, Y, Vx, Vy = RungeKutta(f1, f2, f3, f4, 1, 0, 0, 2 * np.pi, 0.0, 10,
51:                                  0.01)
52:     print("Calculated")
53:     fig, ax = pylab.subplots()
54:     line, = ax.plot(X[0], Y[0], 'ko')
55:     pylab.ylim((-1.1, 1.1))
56:     pylab.xlim((-1.1, 1.1))
57:
58:     def init():
59:         line.set_data(X[0], Y[0])
60:         return line,
61:
62:     def anim(i):
63:         line.set_data(X[i], Y[i])
```

```
64:         return line,
65:
66:     ani = animation.FuncAnimation(
67:         fig, anim, init_func=init, interval=20, frames=range(len(T)))
68:     pylab.show()
69:
70:
71: if __name__ == "__main__":
72:     main()
```