```
 1: import numpy as np
 2: import pylab
 3:
 4: m = 9.1094e-31
 5: hbar = 1.0546e-34
 6: e = 1.6022e-19
 7: V0 = 50 * e
 8: a = 1e-11
 9: N = 1000
10: L = 20 * a
11: h = L / N
12:
13:
14: def V1(x):
15:     return V0 * (x**2) / (a**2)
16:
17:
18: def V2(x):
19:     return V0 * (x**4) / (a**4)
20:
21:
22: def f(r, x, E, V):
23:     psi = r[0]
24:     phi = r[1]
25:     fpsi = phi
26:     # print(V(x)-E)
27:     fphi = (2 * m / hbar**2) * (V(x) - E) * psi
28:     # if x == -L/2 or x == L/2:
29:     #     fphi = 0
30:     return np.array([fpsi, fphi], float)
31:
32:
33: def solve(E, V):
34:     psi = 0.0
35:     phi = 1.0
36:     r = np.array([psi, phi], float)
37:     y = []
38:
39:     for x in np.arange(-L / 2, L / 2, h):
40:         k1 = h * f(r, x, E, V)
41:         k2 = h * f(r + 0.5 * k1, x + 0.5 * h, E, V)
42:         k3 = h * f(r + 0.5 * k2, x + 0.5 * h, E, V)
43:         k4 = h * f(r + k3, x + h, E, V)
44:         r += (k1 + 2 * k2 + 2 * k3 + k4) / 6
45:         y.append(r[0])
46:
47:     return r[0], y
48:
49:
50: def get_state(e1, e2, V):
51:     E1 = e1
52:     E2 = e2
53:     psi2, plt = solve(E1, V)
54:
55:     target = e / 1000
56:     while abs(E1 - E2) > target:
57:         psi1 = psi2
58:         psi2, plt = solve(E2, V)
59:         E1, E2 = E2, E2 - psi2 * (E2 - E1) / (psi2 - psi1)
60:     return E2 / e, plt
61:
62: def integrate(data):
63:     return h * ((0.5 * (np.fabs(data[0])**2 + np.fabs(data[-1])**2)) + sum([np.fabs(x)*
```

```
*2 for x in data]))
   64:
   65: def main():
   66:     print("E0 {}".format(get_state(0.0, e, V1)[0]))
   67:     print("E1 {}".format(get_state(e, 300 * e, V1)[0]))
   68:     print("E2 {}".format(get_state(300 * e, 500 * e, V1)[0]))
   69:     print("E0 {}".format(get_state(0.0, 300*e, V2)[0]))
   70:     print("E1 {}".format(get_state(300*e, 900 * e, V2)[0]))
   71:     print("E2 {}".format(get_state(900 * e, 1200 * e, V2)[0]))
   72:     global L
   73:     L = 10*a
   74:     _, E1 = get_state(0.0, 300*e, V2)
   75:     _, E2 = get_state(300*e, 900*e, V2)
   76:     _, E3 = get_state(900*e, 1200*e, V2)
   77:     scale_1 = integrate(E1)
   78:     scale_2 = integrate(E2)
   79:     scale_3 = integrate(E3)
   80:     X = np.arange(-L/2, L/2, h)
   81:     E1 = [np.fabs(x)**2/scale_1 for x in E1]
   82:     E2 = [np.fabs(x)**2/scale_2 for x in E2]
   83:     E3 = [np.fabs(x)**2/scale_3 for x in E3]
   84:     pylab.plot(X, E1)
   85:     pylab.plot(X, E2)
   86:     pylab.plot(X, E3)
   87:     pylab.show()
   88:
   89: if __name__ == "__main__":
   90:     main()
```