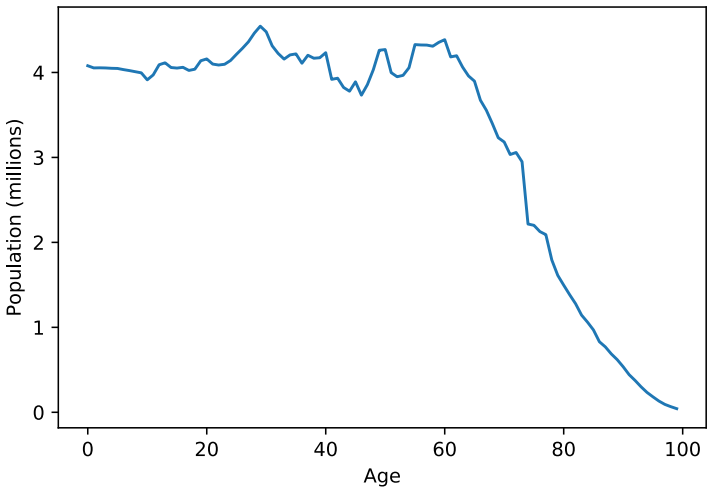

Figure 9.2.: Predicted age distribution in the US in 2020.

9.3. Epidemic dynamics

Let's implement the simulation of the epidemic dynamics from VMLS section 9.3.

In [ ]:
```
T = 210
A = np.array([[0.95,0.04,0,0], [0.05,0.85,0,0], [0,0.1,1,0],
↪  [0,0.01,0,1]])
x_1 = np.array([1,0,0,0])
# state trajectory
state_traj = np.column_stack([x_1, np.zeros((4, T-1))])
# dynamics recursion
for t in range(T-1):
    state_traj[:, t+1] = A @ state_traj[:,t]
import matplotlib.pyplot as plt
plt.ion()
plt.plot(np.arange(T), state_traj.T)
plt.xlabel('Time t')
plt.legend(['Susceptible', 'Infected', 'Recovered', 'Deceased'])
plt.show()
```

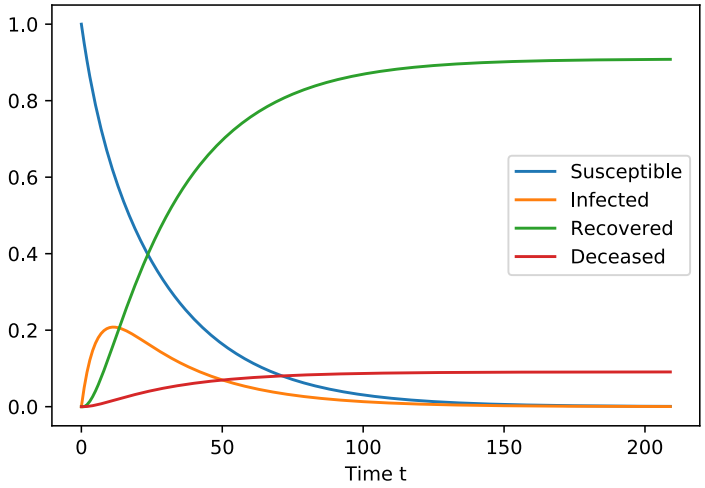

Figure 9.3.: Simulation of epidemic dynamics.

9.4. Motion of a mass

Let's simulate the discretized model of the motion of a mass in section 9.4 of VMLS.

In [ ]:
```
h = 0.01
m = 1
eta = 1
A = np.block([[1,h],[0, 1-h*eta/m]])
B = np.vstack([0,h/m])
x1 = np.array([0,0])
K = 600 #simulate for K*h = 6 seconds
f = np.zeros((K));
f[49:99] = 1
f[99:139] = -1.3
X = np.column_stack([x1, np.zeros((2, K-1))])
for k in range(K-1):
    X[:, k+1] = A @ X[:, k] + f[k]*B.T
import matplotlib.pyplot as plt
plt.ion()
plt.plot(X[0,:])
plt.xlabel('k')
plt.ylabel('Position')
```