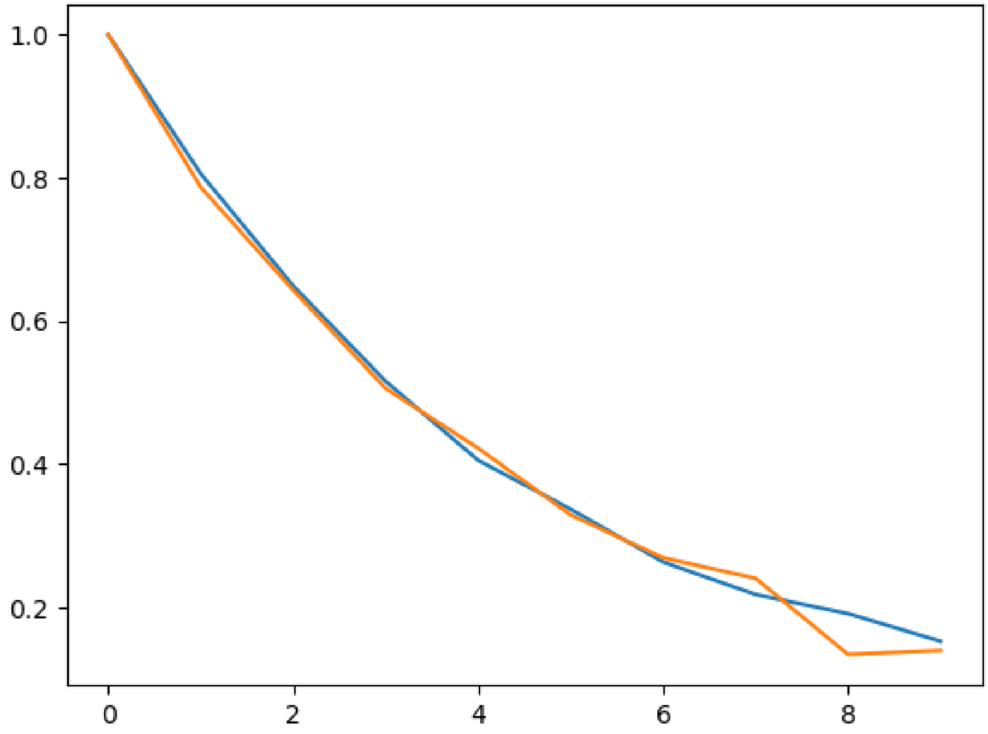
In [27]:

```
import numpy as np
import matplotlib.pyplot as plt

number_of_steps = 10

f_real = np.array([[0.8]])

residual = []
variance = 0.01
for j in range(1000):
    x = np.zeros((1, 1, number_of_steps))
    real_measurement = np.zeros((1, 1, number_of_steps))
    measurement = np.zeros((1, 1, number_of_steps))
    x[0, 0, 0] = 1 #cm
    real_measurement[0, 0, 0] = 1 #cm
    measurement[0, 0, 0] = 1 #cm
    for i in range(1, number_of_steps):
        real_measurement[:, :, i] = f_real @ real_measurement[:, :, i-1]
        x[:, :, i] = f_real @ x[:, :, i-1] + np.random.normal(0, variance)
        measurement[:, :, i] = real_measurement[:, :, i] + np.random.normal(0, variance)
        residual.append(measurement[:, :, i] - x[:, :, i])


plt.plot(x[0, 0, :])
plt.plot(measurement[0, 0, :])
print(np.mean(residual))
```

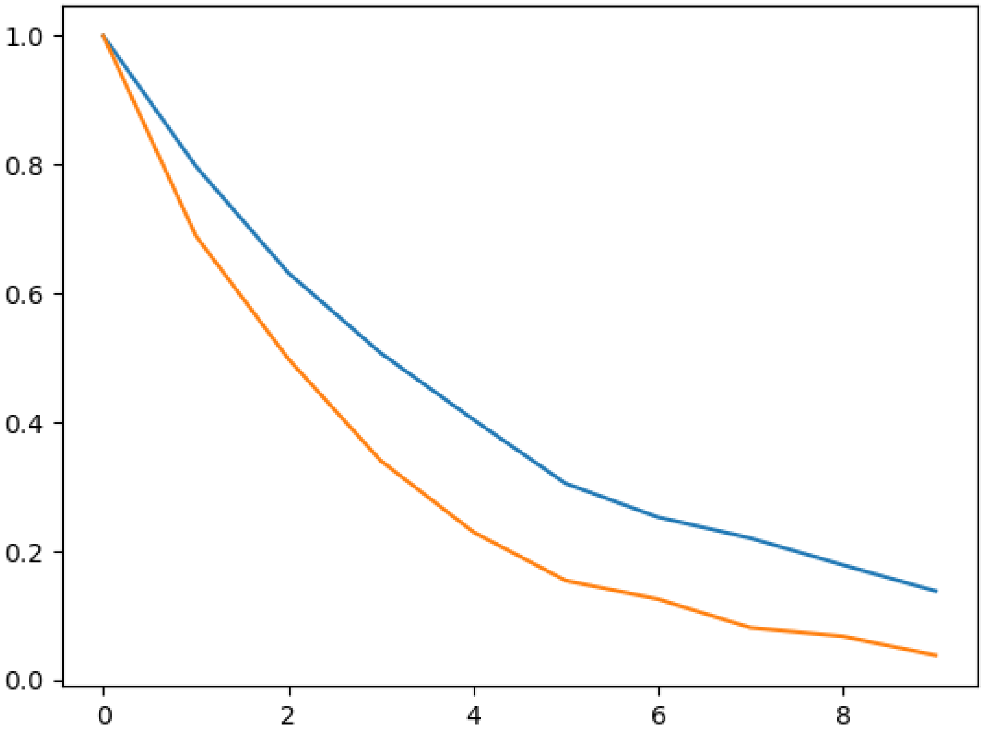
```
0.00048721837802322664
```



In [28]:

```
f = np.array([[0.7]])

residual = []
variance = 0.01
for j in range(1000):
    x = np.zeros((1, 1, number_of_steps))
    real_measurement = np.zeros((1, 1, number_of_steps))
    measurement = np.zeros((1, 1, number_of_steps))
    x[0, 0, 0] = 1 #cm
    real_measurement[0, 0, 0] = 1 #cm
    measurement[0, 0, 0] = 1 #cm
    for i in range(1, number_of_steps):
        real_measurement[:, :, i] = f @ real_measurement[:, :, i-1]
        x[:, :, i] = f_real @ x[:, :, i-1] + np.random.normal(0, variance)
        measurement[:, :, i] = real_measurement[:, :, i] + np.random.normal(0, variance)
        residual.append(measurement[:, :, i] - x[:, :, i])


plt.plot(x[0, 0, :])
plt.plot(measurement[0, 0, :])
print(np.mean(residual))
```

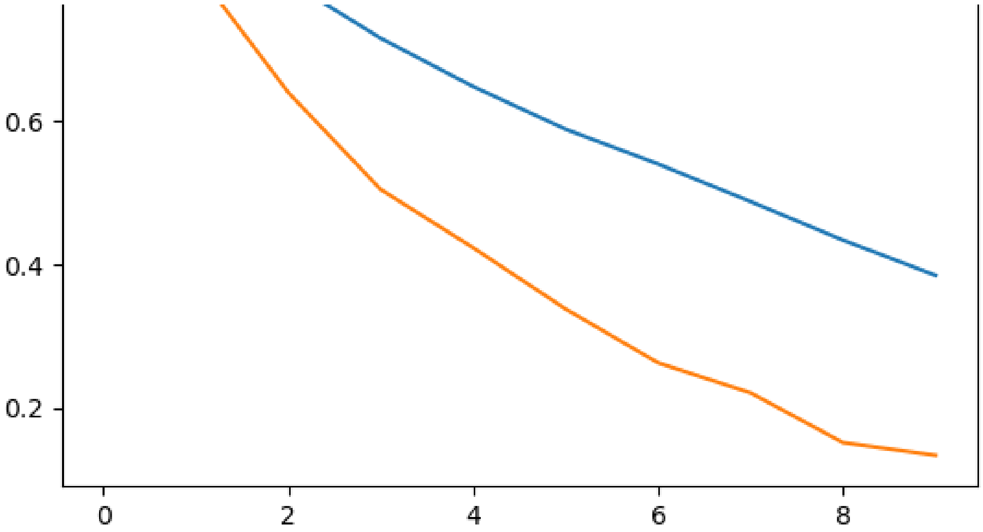
```
-0.1349684323112019
```



In [29]:

```
f = np.array([[0.9]])

residual = []
variance = 0.01
for j in range(1000):
    x = np.zeros((1, 1, number_of_steps))
    real_measurement = np.zeros((1, 1, number_of_steps))
    measurement = np.zeros((1, 1, number_of_steps))
    x[0, 0, 0] = 1 #cm
    real_measurement[0, 0, 0] = 1 #cm
    measurement[0, 0, 0] = 1 #cm
    for i in range(1, number_of_steps):
        real_measurement[:, :, i] = f_real @ real_measurement[:, :, i-1]
        x[:, :, i] = f @ x[:, :, i-1] + np.random.normal(0, variance)
        measurement[:, :, i] = real_measurement[:, :, i] + np.random.normal(0, variance)
        residual.append(measurement[:, :, i] - x[:, :, i])


plt.plot(x[0, 0, :])
plt.plot(measurement[0, 0, :])
print(np.mean(residual))
```

```
-0.22845663603506333
```


0.6
0.4
0.2
0
2
4
6
8