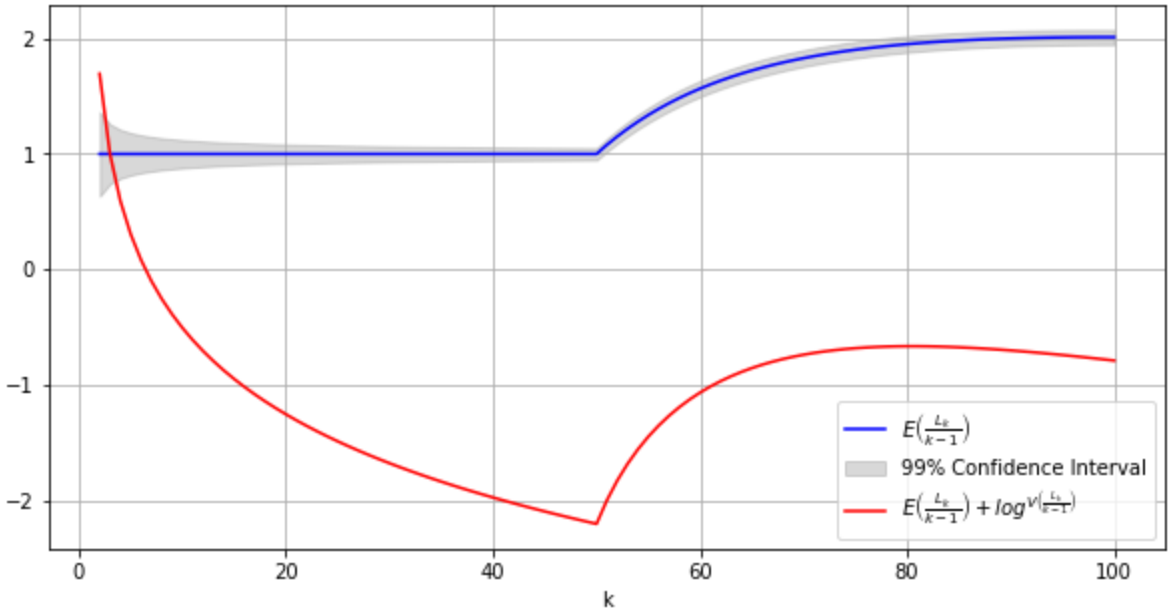
In [4]:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.optimize import minimize
from scipy import signal

Z_ALPHA = 2.576
n = 100
m = 50

def Thm1_plot_Lk(mu1, mu2, sigma):
    delta = np.abs(mu2-mu1)
    k_values = np.arange(1, n + 1)

    E_Lk = np.zeros(n)
    V_Lk = np.zeros(n)

    for k in k_values:
        if k < m:
            E_Lk[k-1] = ((k-1) * sigma**2)/(k-1)
            V_Lk[k-1] = (2*(k-1) * sigma**4) / (k-1)**2
        else:
            E_Lk[k-1] = ((k-1) * sigma**2 + m*(delta**2)*(1-m/k))/ (k-1)
            V_Lk[k-1] = (2*(k-1) * sigma**4 + 4*m*(delta**2)*(sigma**2)*(1-m

    lower_bound = E_Lk - Z_ALPHA * np.sqrt(V_Lk/n)
    upper_bound = E_Lk + Z_ALPHA * np.sqrt(V_Lk/n)

    new_E_Lk = E_Lk + np.log(V_Lk)

    plt.figure(figsize=(10, 5))

    plt.xlabel('k')
    plt.plot(k_values, E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right)$', color
    plt.fill_between(k_values, lower_bound, upper_bound, color='gray', alpha

    plt.plot(k_values, new_E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right) + lo

    plt.legend()
    plt.grid(True)
    plt.show()

def Thm2_plot_Lk(mu, sigma1, sigma2):
    gamma = (sigma2/sigma1)**2
    k_values = np.arange(1, n + 1)

    E_Lk = np.zeros(n)
    V_Lk = np.zeros(n)

    for k in k_values:
        if k < m:
            E_Lk[k-1] = ((k-1) * sigma1**2)/(k-1)
            V_Lk[k-1] = (2*(k-1) * sigma1**4) / (k-1)**2
        else:
```

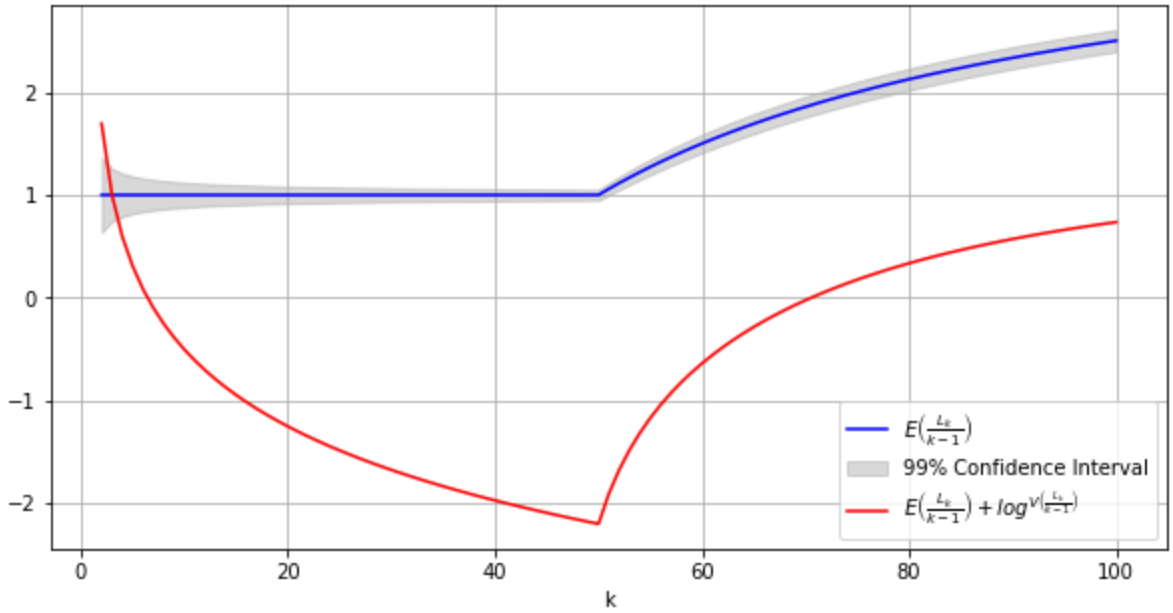
```
            E_Lk[k-1] = ((k-1) * sigma1**2 + (gamma-1)*sigma1**2*(k-1)*(1-m/
            V_Lk[k-1] = (2*(k-1) * sigma1**4
                         + 2*(gamma**2-1)*sigma1**4*k
                         + 4*m*(gamma-1)*sigma1**4*(1/k)
                         + 2*m**2*(gamma-1)**2*sigma1**4*(1/k**2)
                         - 2*(gamma**2-1)*(m+1)*sigma1**4) / (k-1)**2

    lower_bound = E_Lk - Z_ALPHA * np.sqrt(V_Lk/n)
    upper_bound = E_Lk + Z_ALPHA * np.sqrt(V_Lk/n)

    new_E_Lk = E_Lk + np.log(V_Lk)

    plt.figure(figsize=(10, 5))

    plt.xlabel('k')
    plt.plot(k_values, E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right)$', color
    plt.fill_between(k_values, lower_bound, upper_bound, color='gray', alpha

    plt.plot(k_values, new_E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right) + lo

    plt.legend()
    plt.grid(True)
    plt.show()

def Thm3_plot_Lk(mu1, mu2, sigma1, sigma2):
    delta = np.abs(mu2-mu1)
    gamma = (sigma2/sigma1)**2
    k_values = np.arange(1, n + 1)

    E_Lk = np.zeros(n)
    V_Lk = np.zeros(n)

    for k in k_values:
        if k < m:
            E_Lk[k-1] = ((k-1) * sigma1**2)/(k-1)
            V_Lk[k-1] = (2*(k-1) * sigma1**4) / (k-1)**2
        else:
            E_Lk[k-1] = ((k-1) * sigma1**2
                         + (gamma-1)*sigma1**2*k
                         + (m*(gamma-1)*sigma1**2 - (m**2*delta**2))*(1/k)
                         + (1-gamma)*(1+m)*sigma1**2 + m*delta**2)/ (k-1)
            V_Lk[k-1] = (2*(k-1) * sigma1**4
                        + 2*(gamma-1)*sigma1**4*k
                        + (4*m*(gamma-2)*delta**2*sigma1**2 - 4*m*sigma1**4)
                        + (4*m**2*(1-gamma)*delta**2*sigma1**2 + 2*m**2*(1-g
                        + (2*sigma1**4*(1+m-m*gamma+gamma) + 4*delta**2*sigm


    lower_bound = E_Lk - Z_ALPHA * np.sqrt(V_Lk/n)
    upper_bound = E_Lk + Z_ALPHA * np.sqrt(V_Lk/n)

    new_E_Lk = E_Lk + np.log(V_Lk)

    plt.figure(figsize=(10, 5))

    plt.xlabel('k')
```

```
    plt.plot(k_values, E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right)$', color
    plt.fill_between(k_values, lower_bound, upper_bound, color='gray', alpha

    plt.plot(k_values, new_E_Lk, label=r'$E\left(\frac{L_k}{k-1}\right) + lo

    plt.legend()
    plt.grid(True)
    plt.show()

Thm1_plot_Lk(0, 2, 1)
Thm2_plot_Lk(0, 1, 2)
Thm3_plot_Lk(0, 2, 1, 2)
```

```
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:20: RuntimeWarning: invalid value encountered in long_scalars
  E_Lk[k-1] = ((k-1) * sigma**2)/(k-1)
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:21: RuntimeWarning: invalid value encountered in long_scalars
  V_Lk[k-1] = (2*(k-1) * sigma**4) / (k-1)**2
```




```
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:52: RuntimeWarning: invalid value encountered in long_scalars
  E_Lk[k-1] = ((k-1) * sigma1**2)/(k-1)
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:53: RuntimeWarning: invalid value encountered in long_scalars
  V_Lk[k-1] = (2*(k-1) * sigma1**4) / (k-1)**2
```




```
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:89: RuntimeWarning: invalid value encountered in long_scalars
  E_Lk[k-1] = ((k-1) * sigma1**2)/(k-1)
/var/folders/jr/dh2sym6s07l4mjz0kr64zr4m0000gn/T/ipykernel_2339/4100556691.p
y:90: RuntimeWarning: invalid value encountered in long_scalars
  V_Lk[k-1] = (2*(k-1) * sigma1**4) / (k-1)**2
```