```python
# I completed the exam independently using only permitted resources and fully
# observed the FIU student conduct and honor code.
# --- CATALINA CISNEROS PYTHON EXAM FALL 2025 ---

# 1 - done

# 2 - Import LT.box and numpy
import LT.box as B
import numpy as np

# 3 - Load resistivity data files and load them in the specified variables
w_resistivity = B.get_file('W_resistivity.data')
T, rho = w_resistivity['T'], w_resistivity['R']

# 4 - Calculate rel resistivity
rho_300 = rho[T == 300][0]
rho_r = rho / rho_300

# 5 - create figure
B.pl.figure()

# 6 - plot values
B.plot_exp(rho_r, T)

# 7 - polyfit 7th order
p7 = B.polyfit(rho_r, T, 7)
p7.plot()

# 8 - Label axis and give it a title
B.pl.xlabel('Relative Resistivity')
B.pl.ylabel('Temperature (K)')
B.pl.title('W temperature as a function of resistivity')

# 9 - Load the other file
dexp = B.get_file('SB_measurements.data')
I_l, V_l, PmV, dPmV = dexp['I'], dexp['V'], dexp['PmV'], dexp['sig_PmV']

# 10 - Get parameters
R_wire = dexp.par['Rw'] #r of the wires connecting the lamp to the power
R_l_cold = dexp.par['R_Lc'] #resistance of the cold lamp (meaning at 300K)

# 11 - From the cold lamp resistance subtract twice the wire resistance
R_l_cold_corr = R_l_cold - (2* R_wire)

# 12 - calculate the lamp
R_l = V_l / I_l # Use Ohm's Law

# 13 - subtract twice the wire resistance
R_l_corr = R_l - (2* R_wire)

# 14 - ratio of the corrected lamp resistance
```

```
R_l_rel = R_l_corr / R_l_cold_corr

# 15 - calculate the corresponding lamp temperature
T_L = p7.poly(R_l_rel)

# 15a - extra
B.pl.figure()
B.plot_exp(R_l_rel, T_L, marker='^', color='blue')
p7.plot()
B.pl.xlabel('Lamp Relative Resistivity')
B.pl.ylabel('Calculated W Temperature (K)')
B.pl.title('Calculated W Temperature vs Lamp Relative Resistivity')

# 16 - Calculate ln of PmV and its dPmV
logP = np.log(PmV)
dlogP = dPmV / PmV

# 17 - log of the lamp temp
logT_L = np.log(T_L)

# 18 - plot
B.pl.figure()
B.plot_exp(logT_L, logP, dlogP)

# 19 - fitted line
logf = B.linefit(logT_L, logP, dlogP)

# 20 - plot fitted line
logf.plot()

# 21 - label axis
B.pl.xlabel('Log of the lamp temperature [ln(K)]')
B.pl.ylabel('Log of Power [ln(mV)]')
B.pl.title('Test of the Stefan-Boltzmann Law')

# 22 - print fitted slope
B.pl.text(7.1, -1.2, f"My slope is {
    logf.slope:.2f} ± {logf.sigma_s:.2f} (expected 4.00)")


# 23 - New figure with the same graph as item 18
B.pl.figure()
B.plot_exp(logT_L, logP, dlogP)

# 24 - Create 3 arrays
tmin = np.zeros(8)
slopes = np.zeros(8)
dslopes = np.zeros(8)

# 34a - for step 34: create chisquared array
chi_red = np.zeros(8)
```

```python
# --- LOOP ---

# 25 - Start a for loop
for index in range (8) :

    # 26 - define starting point for each range
    tmin[index] = T_L.min() + 200 * index


    # 27 - T_L is larger than tmin[index], and up to T_L.max()
    sel_ht = B.in_between(T_L, tmin[index], T_L.max())

    # 28 - linear fit as 19 for the range above
    logf1 = B.linefit(logT_L[sel_ht], logP[sel_ht], dlogP[sel_ht])

    # 29 - store data in the arrays
    slopes[index] = logf1.slope
    dslopes[index] = logf1.sigma_s

    # 30 - Plot and print the results on the figure
    logf1.plot()
    B.pl.text(7.6, 0.8 - 0.3*index, f"T>{tmin[index]:.0f}K: slope = {
        logf1.slope:.2f} ± {logf1.sigma_s:.2f}")

    # 34b - for step 34: store the chisquared in another array
    chi_red[index] = logf1.chi_red

    # 31 - loop ends


# 32 - label the graphs accordingly
B.pl.xlabel('Log of the lamp temperature [ln(K)]')
B.pl.ylabel('Log of Power [ln(mV)]')
B.pl.title('Test of the Stefan-Boltzmann Law in different fitting ranges')

# 33 - new figure of slopes (w unc) usgin the 8 fitting resutls from the loop
B.pl.figure()
B.plot_exp(tmin, slopes, dslopes)
B.pl.xlabel('Minimum Temperature (K)')
B.pl.ylabel('Fitted Slope')
B.pl.title('Slopes vs Minimum Temperature')

# 34 - plot the chisquared in a new fig
B.pl.figure()
B.plot_exp(tmin, chi_red)
B.pl.xlabel('Minimum Temperature of Fit Range (K)')
B.pl.ylabel('Reduced Chi-squared')
B.pl.title('Reduced Chi-squared vs Minimum Temperature')
```