Modelação de Sistemas Físicos - Aula Prática nº2

Regressão linear pelo método dos mínimos quadrados:

Para um dado conjunto de N pares de valoes $\{(x_i, y_i)\}$, com $i = 1, \ldots, N$, a reta $y(x) = mx + b$ que minimiza a *distância* $\sum_i [y_i - y(x_i)]^2$ é obtida quando,

$$m = \frac{N \sum_i x_i y_i - \sum_i x_i \sum_i y_i}{N \sum_i x_i^2 - (\sum_i x_i)^2},$$

$$b = \frac{\sum_i x_i^2 \sum_i y_i - \sum_i x_i \sum_i x_i y_i}{N \sum_i x_i^2 - (\sum_i x_i)^2}$$

sendo a qualidade do ajuste dada pelo coeficiente de correlação,

$$r^2 = \frac{(N \sum_i x_i y_i - \sum_i x_i \sum_i y_i)^2}{[N \sum_i x_i^2 - (\sum_i x_i)^2][N \sum_i y_i^2 - (\sum_i y_i)^2]}.$$

O erro associado ao declive e ordenada no orígem é dado por,

$$\Delta m = |m| \sqrt{\frac{(1/r^2) - 1}{N - 2}}$$

$$\Delta b = \Delta m \sqrt{\frac{\sum_i x_i^2}{N}}$$

Exercício 1

Escreva uma função em python que calcule as quantidades anteriores, assumindo que são fornecidos os valores de x_i e y_i.

Como passo intermédio, calcule as somas:

$$\sum_{i=1}^{N} x_i y_i \quad \sum_{i=1}^{N} x_i \quad \sum_{i=1}^{N} y_i \quad \sum_{i=1}^{N} x_i^2 \quad \sum_{i=1}^{N} y_i^2$$

A função deve retornar as quantidades m, b, r^2 , Δm e Δb.

In [1]:
```
# Função minimos quadrados
# Argumentos de entrada:
#     x: cadeia de valores x_i
#     y: cadeia de valores y_i
# Argumentos de saída:
#     m: declive
#     b: ordenada na orígem
#     r2: coeficiente de correlação ** 2
#     dm: erro do declive
#     db: erro da ordenada na orígem
def minimos_quadrados(x, y):
```

```
    # numero de pares {(x_i, y_i)}. Assumimos que x.size = y.size
    N = x.size

    # Definir somas elementares (escalares)
    sum_x = np.sum(x)
    sum_y = np.sum(y)
    sum_x2 = np.sum(x ** 2)
    sum_y2 = np.sum(y ** 2)
    sum_xy = np.sum(x * y)

    # Calcular declive
    m = (N * sum_xy - sum_x * sum_y) / (N * sum_x2 - sum_x ** 2)

    # Calcular ordenada no origem
    b = (sum_x2 * sum_y - sum_x * sum_xy) / (N * sum_x2 - sum_x ** 2)

    # Calcular coeficiente de correlação
    r2 = (N * sum_xy - sum_x * sum_y) ** 2 / ((N * sum_x2 - sum_x ** 2) * (N * sum_y2

    # Calcular erro do declive
    dm = np.absolute(m) * np.sqrt((1 / r2 - 1) / (N - 2))

    # Calcular erro da ordenada na orígem
    db = dm * np.sqrt((sum_x2) / N)

    # Devolver os argumentos desejados
    return m, b, r2, dm, db
```

Pergunta 1:

Como podemos testar se a função está a funcionar corretmente?

Aplicação de regressão linear a um conjunto de dados

Numa experiência de difração de um feixe de luz por uma fenda única foram medidos 7 pares de valores da distância da fonte de luz ao alvo e correspondente distância entre máximos luminosos consecutivos $\{L_i, X_i\}$ (ver [link]).

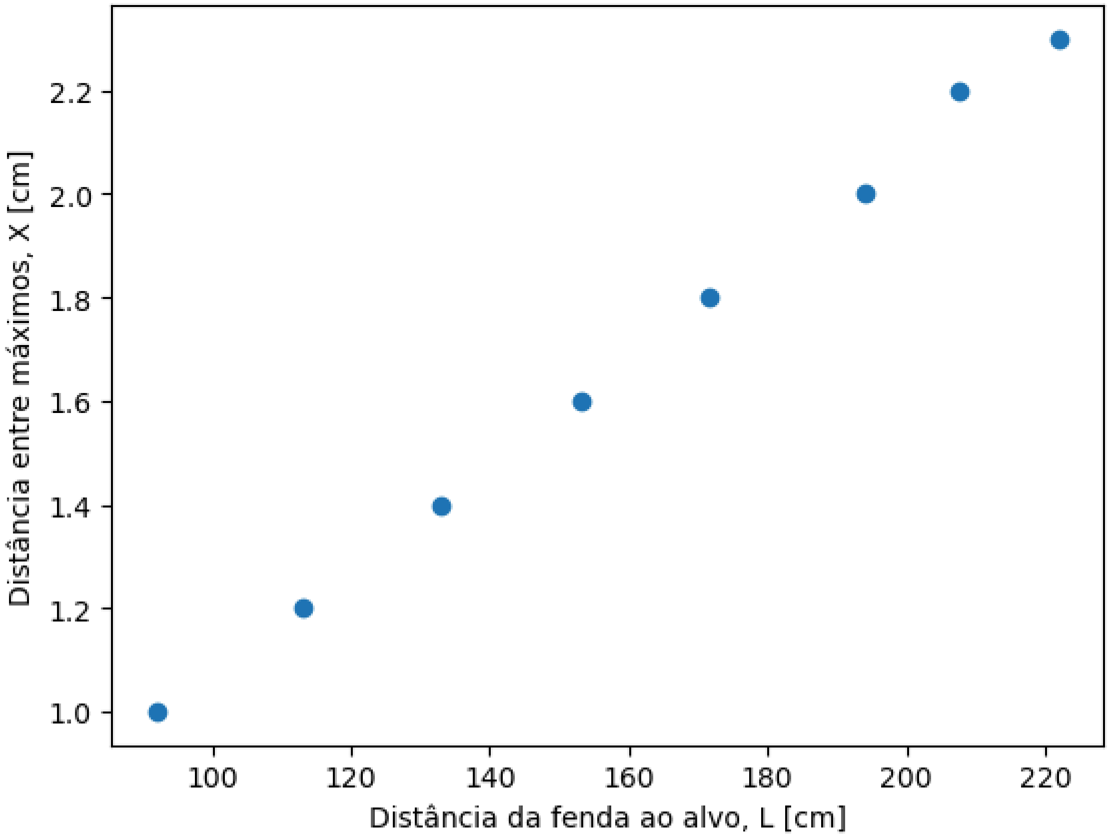
L_i (cm)	222.0	207.5	194.0	171.5	153.0	133.0	113.0	92.0
X_i (m)	2.3	2.2	2.0	1.8	1.6	1.4	1.2	1.0

Exercício 2. Escreva um código em Python que:

a) Representa os dados experimentais num gráfico.

In [2]:
```
# Importar os pacotes "numpy" e "matplotib"
import numpy as np
import matplotlib.pyplot as plt

# Definir cadeias de valores {x_i} e {y_i}
L_i = np.array([222.0, 207.5, 194.0, 171.5 , 153.0, 133.0, 113.0, 92.0]) # [cm]
X_i = np.array([2.3, 2.2, 2.0, 1.8, 1.6, 1.4, 1.2, 1.0])                # [cm]

# Representar dados num grafico (usando o matplotlib)
plt.scatter(L_i, X_i)
plt.xlabel("Distância da fenda ao alvo, L [cm]")
```

```
plt.ylabel("Distância entre máximos, X [cm]")
plt.show()
```



b) Calcule as quantidades m, b, r^2, Δm, e Δb para este conjunto de dados.

In [3]:
```
# Executar a função (calcular melhor reta)
m, b, r2, dm, db = minimos_quadrados(L_i, X_i)

# Imprimir resultados
print("m = {0:.4f}".format(m))
print("b = {0:.2f} cm".format(b))
print("r² = {0:.4f}...".format(r2))
print("Δm = {0:.4f}".format(dm))
print("Δb = {0:.2f} cm".format(db))
```

```
m = 0.0102
b = 0.06 cm
r² = 0.9985...
Δm = 0.0002
Δb = 0.03 cm
```

Compare com os valores:

$$m = 0.01015505; \quad \Delta m = 0.000162973$$

$$b = 0.05507544; \quad \Delta b = 0.02713077$$

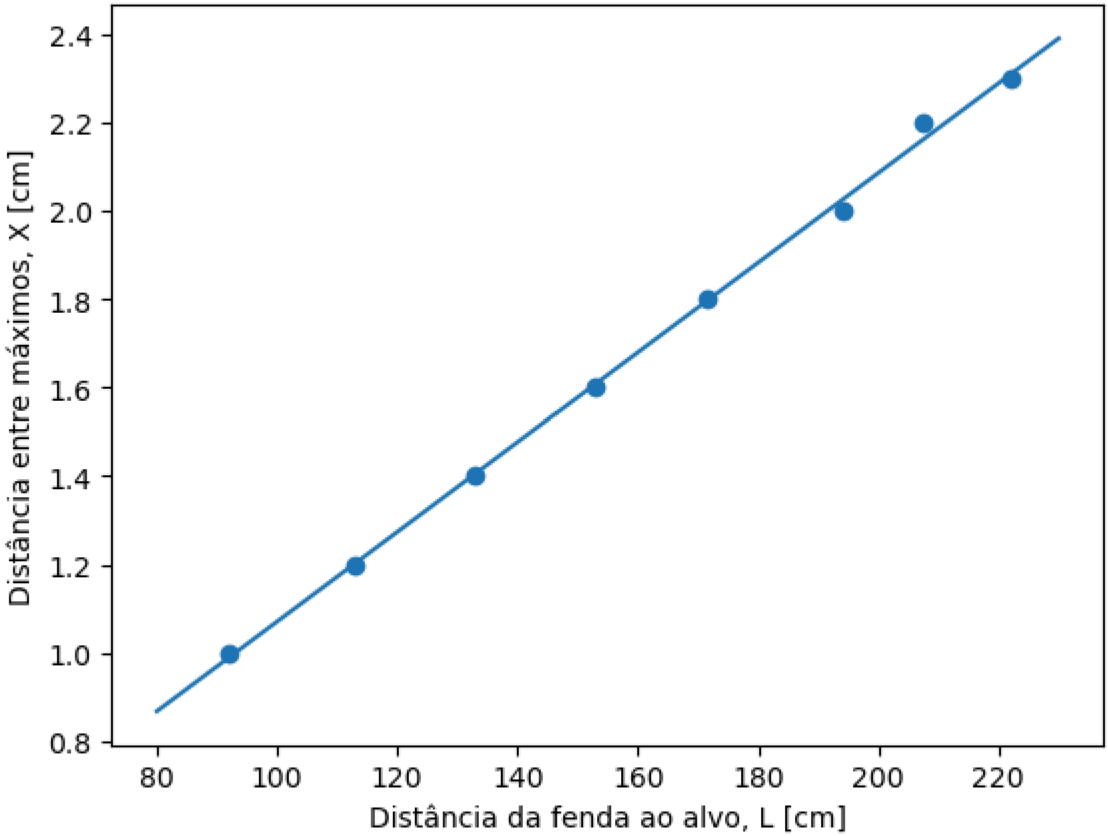
$$r^2 = 0.99845714$$

De seguida:

c) Representa a reta $y = mx + b$ no gráfico.

In [4]:
```
# Definir dois pontos (x0, y0) e (x1, y1) para representar melhor reta
x = np.array([80.0, 230.0])
y = m * x + b
```

```
# Representar dados num grafico (usando o matplotlib)
plt.scatter(L_i, X_i)
plt.plot(x, y)
plt.xlabel("Distância da fenda ao alvo, L [cm]")
plt.ylabel("Distância entre máximos, X [cm]")
plt.show()
```



d) **Interpolação:** Encontre o valor de X, quando $L = 165.0$ cm. Use a reta determinada pela regressão linear.

In [5]:
```
print("X(L=165.0 cm) = {0:.2f} cm".format(m * 165.0 + b))
```

```
X(L=165.0 cm) = 1.73 cm
```

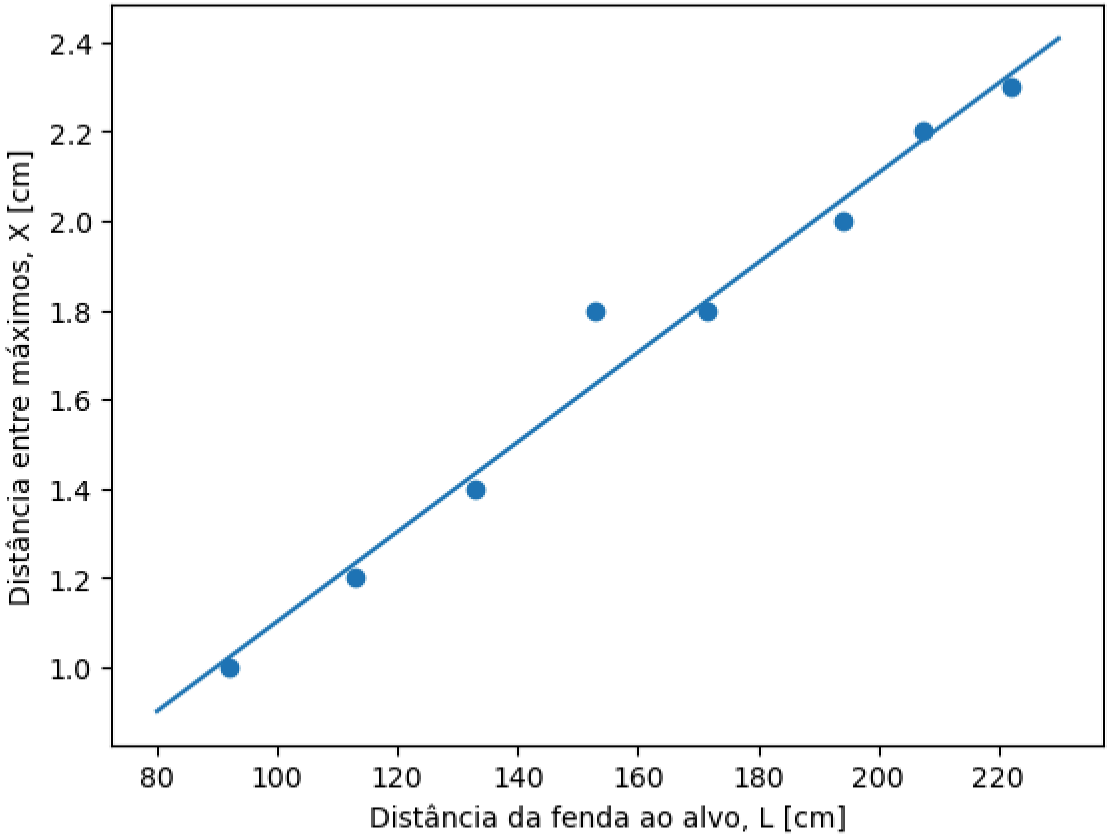
e) Afaste da reta encontrada um dos valores medidos de X. Compare o coeficiente de determinação com o valor anterior. Faça um gráfico com os novos pontos experimentais e a nova reta.

In [6]:
```
# Afastei o valor perto do centro, X(153) de 1.6 para 1.8
X_i = np.array([2.3, 2.2, 2.0, 1.8, 1.8, 1.4, 1.2, 1.0])  # [cm]
```

f) Calcule de novo os valores de m, b e o coeficiente de determinação r^2 e compare com o valor anterior. Faça um gráfico com os novos pontos experimentais e a nova reta.

In [7]:
```
# Executar novamente a função (calcular melhor reta)
m, b, r2, dm, db = minimos_quadrados(L_i, X_i)

# Definir dois pontos (x1, y1) e (x2, y2) para representar melhor reta
L = np.array([80.0, 230.0])
X = m * L + b

# Representar dados num grafico (usando o matplotlib)
plt.scatter(L_i, X_i)
plt.plot(L, X)
```

```
plt.xlabel("Distância da fenda ao alvo, L [cm]")
plt.ylabel("Distância entre máximos, X [cm]")
plt.show()

# Imprimir resultados
print("m = {0:.4f} cm^2".format(m))
print("b = {0:.1f} cm".format(b))
print("r² = {0:.4f}...".format(r2))
print("Δm = {0:.4f} cm^2".format(dm))
print("Δb = {0:.1f} cm".format(db))
```




```
m = 0.0101 cm^2
b = 0.1 cm
r² = 0.9782...
Δm = 0.0006 cm^2
Δb = 0.1 cm
```

O coeficiente de correlação baixa de 0.998... para 0.978..., ou seja, a qualidade do ajuste piora.

O erro Δb aumenta uma ordem de grandeza

Pergunta 2:

Sugere uma maneira de estimar o erro no valor de X encontrado em d)

Exercício 3. Regressão linear pelo método dos mínimos quadráticos

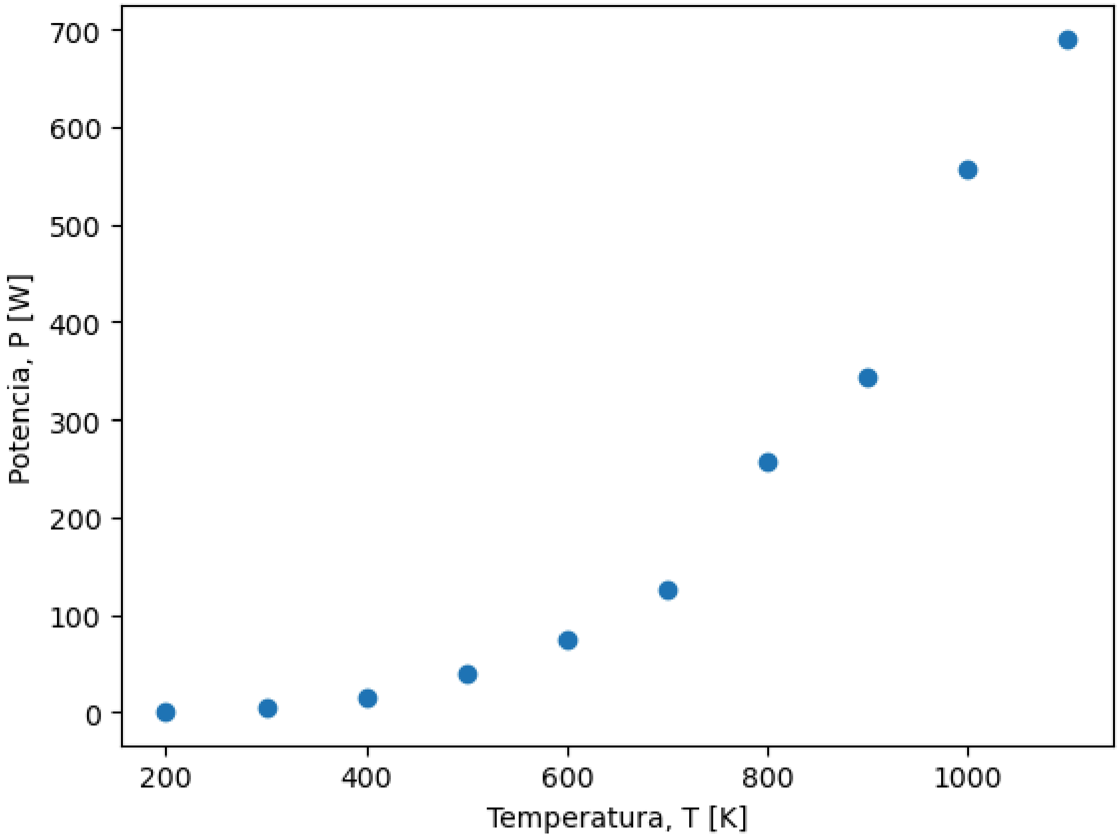
Foi medida a energia por segundo (potência) emitida por um corpo negro (corpo que absorve toda a energia que incide nele) com uma área superficial $A = 100\ \mathrm{cm}^2$ em função da temperatura absoluta, T, e registada na seguinte tabela,

Grandeza	Medições
$T\,(K)$	`200.0, 300.0, 400.0, 500.0, 600.0, 700.0, 800.0, 900.0, 1000.0, 1100.0`
$P\,(W)$	`0.6950, 4.363, 15.53, 38.74, 75.08, 125.2, 257.9, 344.1, 557.4, 690.7`

a) Apresente estas medições num gráfico. Ao analisar o gráfico, verifique se a relação entre a energia emitida e a temperatura é linear?

In [8]:

```
# Temperatura medida [K]
T_i = np.array([200.0, 300.0, 400.0, 500.0, 600.0, 700.0, 800.0, 900.0, 1000.0, 1100.

# Energia medida [W]
P_i = np.array([0.6950, 4.363, 15.53, 38.74, 75.08, 125.2, 257.9, 344.1, 557.4, 690.7

# Representar dados num grafico (usando o matplotlib)
plt.scatter(T_i, P_i)
plt.xlabel("Temperatura, T [K]")
plt.ylabel("Potencia, P [W]")
plt.show()
```

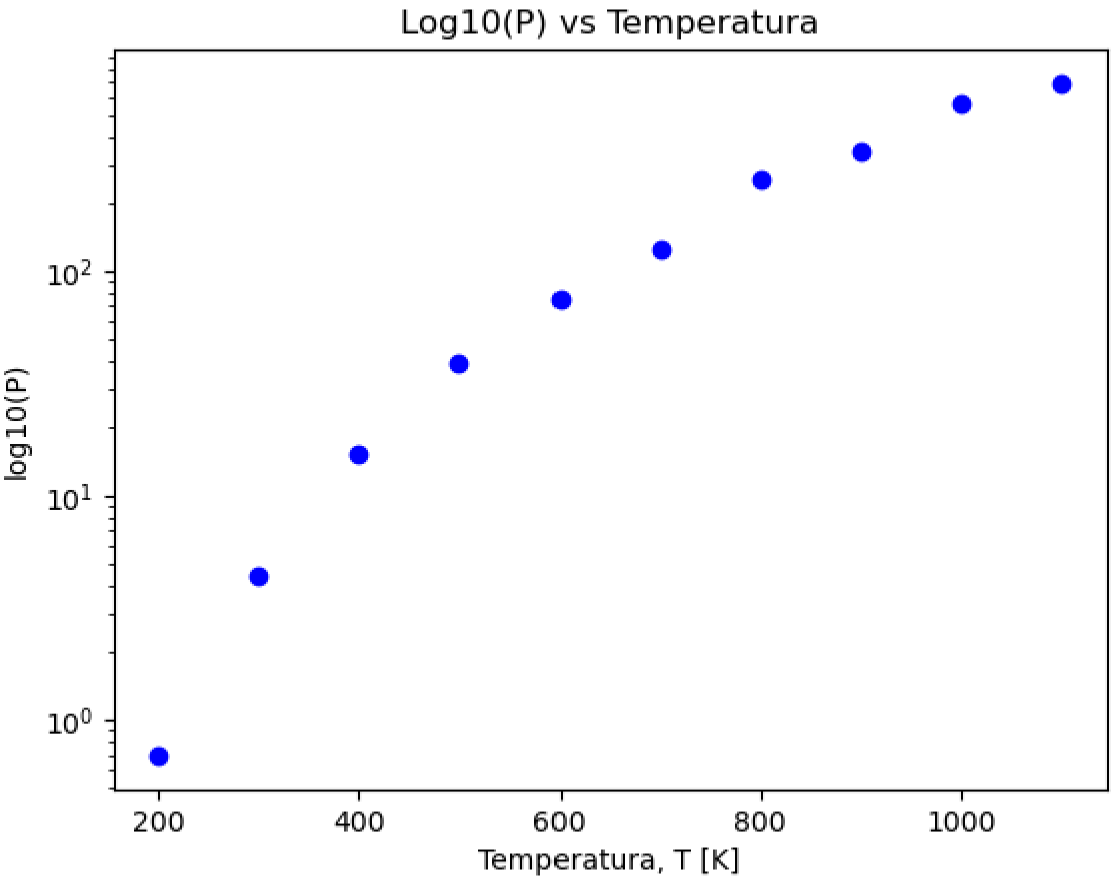


Claramente, a relação entre a potência dissipada e a temperatura do corpo negro **não é linear**.

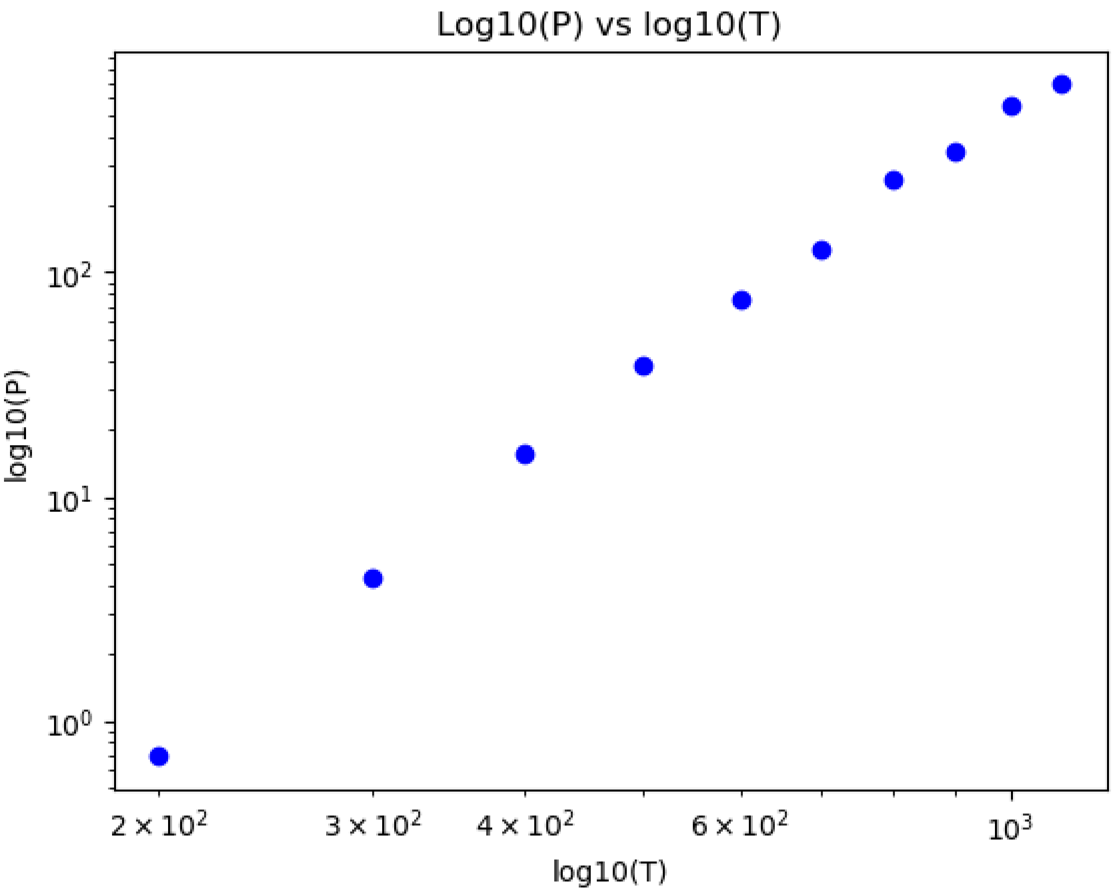
b) Apresente as medições num gráfico log-linear e num gráfico log-log, representando a temperatura no eixo horizontal.

In [9]:

```
# Representar dados num grafico (usando o matplotlib)
plt.semilogy(T_i, P_i, 'bo')
plt.title('Log10(P) vs Temperatura')
plt.xlabel("Temperatura, T [K]")
plt.ylabel("log10(P)")
plt.show()
```



In [10]:
```
# Representar dados num grafico (usando o matplotlib)
plt.loglog(T_i, P_i, 'bo')
plt.title('Log10(P) vs log10(T)')
plt.xlabel("log10(T)")
plt.ylabel("log10(P)")
plt.show()
```



c) Qual a dependência entre a potência emitida e a temperatura: lei de potência ou lei exponencial?

Estamos a lidar com uma lei de potência.

d) Transforme os dados de modo a que a relação seja linear (via linearização), e encontre a linha de melhor ajuste utilizando o método dos mínimos quadrados. Qual é o valor de r^2 ?

Vamos linearizar uma lei de potência $P = cT^n$,

$$P = cT^n \Leftrightarrow \ln P = \ln c + n \ln T$$

$$X = \ln T$$

$$Y = \ln P$$

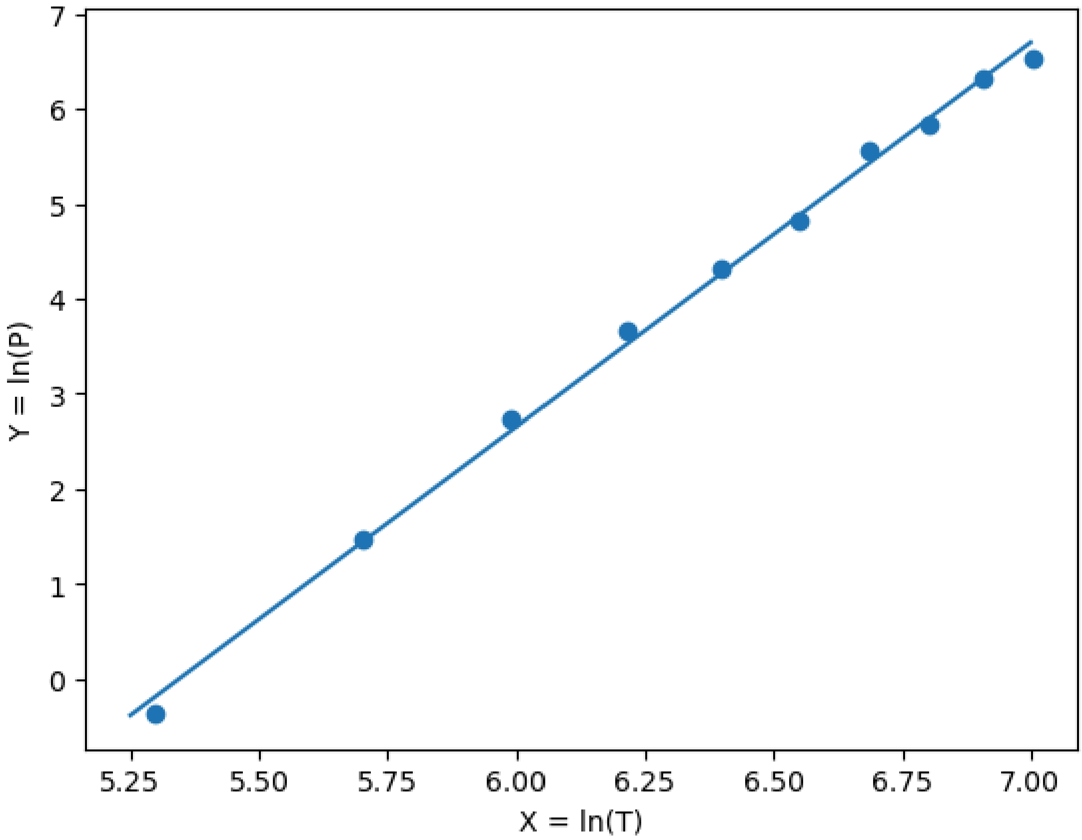
$$m = n$$

$$b = \ln c$$

In [11]:

```python
# Linearização:
# Atenção à diferença entre as funções do numpy:
# numpy.log = logarítmo natural (base "e")
# numpy.log10 = logaritmo de base decimal
X_i = np.log(T_i)
Y_i = np.log(P_i)

# Calcular melhor reta
m, b, r2, dm, db = minimos_quadrados(X_i, Y_i)

# Imprimir resultados
print("m = {0:.4f}".format(m))
print("b = {0:.2f} cm".format(b))
print("r² = {0:.4f}...".format(r2))
print("Δm = {0:.4f}".format(dm))
print("Δb = {0:.2f} cm".format(db))


# Definir dois pontos (x0, y0) e (x1, y1) para representar a melhor reta
x = np.array([5.25, 7.0])
y = m * x + b

# Representar dados num grafico (usando o matplotlib)
plt.scatter(X_i, Y_i)
plt.plot(x, y)
plt.xlabel("X = ln(T)")
plt.ylabel("Y = ln(P)")
plt.show()

print("n = m = ", m)
print("c = exp(b) = ", np.exp(b), "W/K^4")
```

```
m = 4.0483
b = -21.64 cm
r² = 0.9973...
Δm = 0.0744
Δb = 0.47 cm
```




```
n = m =  4.048269520127743
c = exp(b) =  4.0095023319769386e-10 W/K^4
```

e) Escreva a função que representa a relação entre T e P encontrada.

Tendo ja assumido que $P = cT^n$, podemos afirmar que:

$$P = 4.0 \times^{-10} \ T^{4.0}$$

De facto, segundo a lei de Stefan–Boltzmann, a potência dissipada por um corpo negro é proporcional a T^4.

Exercício 4. Regressão linear pelo método dos mínimos quadrados

Foi medida a atividade de uma amostra do isótopo radioativo $^{131}\mathrm{I}$. Foram registadas 10 medições separadas por intervalos de 5 dias. Os valores medidos são, em mCi (milicurie, em que a unidade Curie indica 3.7×10^{-10} desintegrações nucleares por segundo.):

```
9.676, 6.355, 4.261, 2.729, 1.862, 1.184, 0.7680, 0.4883, 0.3461,
0.2119
```

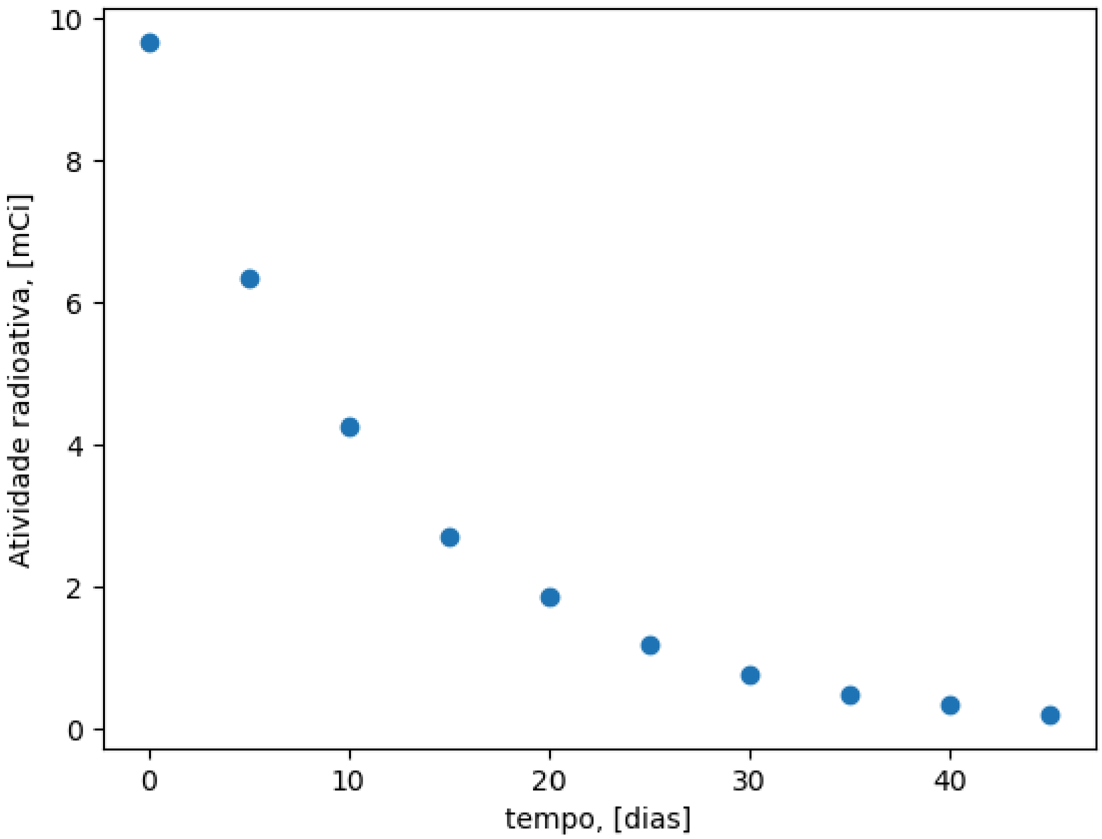
A metado do tempo de vida deste isótopo radioativo é $\tau_{1/2} = 8.0197 \text{ dias}$

Pergunta 3:

Quanto tempo demora para a atividade da amostra diminuir por um fator de 2? (Isto chama-se a semivida do isótopo)

a) Apresente estas medições num gráfico. A analisar o gráfico, a relação entre a atividade e o tempo é linear?

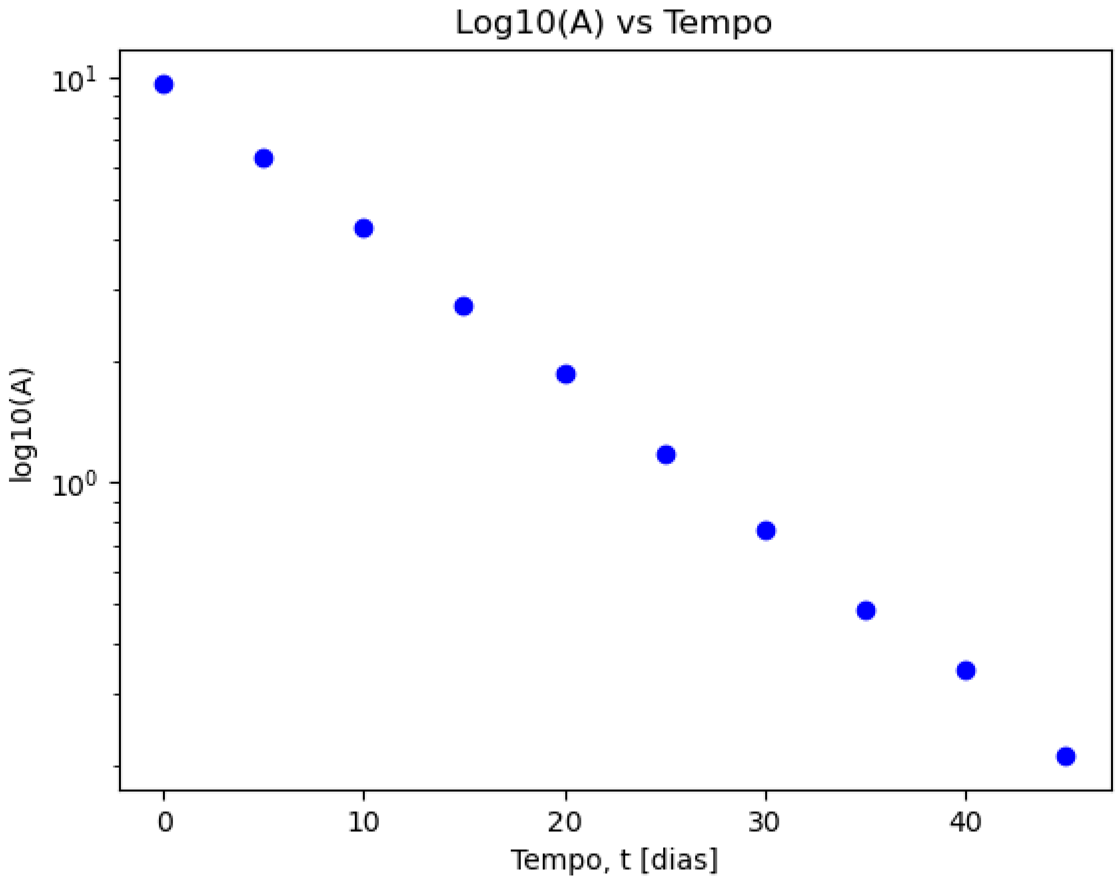
In [12]:
```python
# Tempo [dias]
t_i = np.array([0.0, 5.0, 10.0, 15.0, 20.0, 25.0, 30.0, 35.0, 40.0, 45.0])

# Atividade radioativa [mCi]
A_i = np.array([9.676 , 6.355, 4.261, 2.729, 1.862, 1.184, 0.7680, 0.4883, 0.3461, 0.

# Representar dados num grafico (usando o matplotlib)
plt.scatter(t_i, A_i)
plt.xlabel("tempo, [dias]")
plt.ylabel("Atividade radioativa, [mCi]")
plt.show()
```



Claramente, a relação entre a atividade radioativa e o tempo decorrido **não é linear**.

b) Apresente as medições num gráfico semilog. Como depende a atividade com o tempo?

In [13]:
```python
# Representar dados num grafico (usando o matplotlib)
plt.semilogy(t_i, A_i, 'bo')
plt.title('Log10(A) vs Tempo')
plt.xlabel("Tempo, t [dias]")
plt.ylabel("log10(A)")
plt.show()
```



O logaritmo da atividade depende linearmente do tempo: **Lei exponencial**

c) Encontre a função que relaciona a atividade radioativa com o tempo, incluindo os valores das constantes.

A lei que relaciona a atividade radioativa com o tempo é do tipo:

$$\log(A) = c_1 t + c_2 \quad (Y = c_1 X + c_2)$$

Portanto, trata-se de uma lei exponencial:

$$A = \log(c_2)\exp(c_1 t),$$

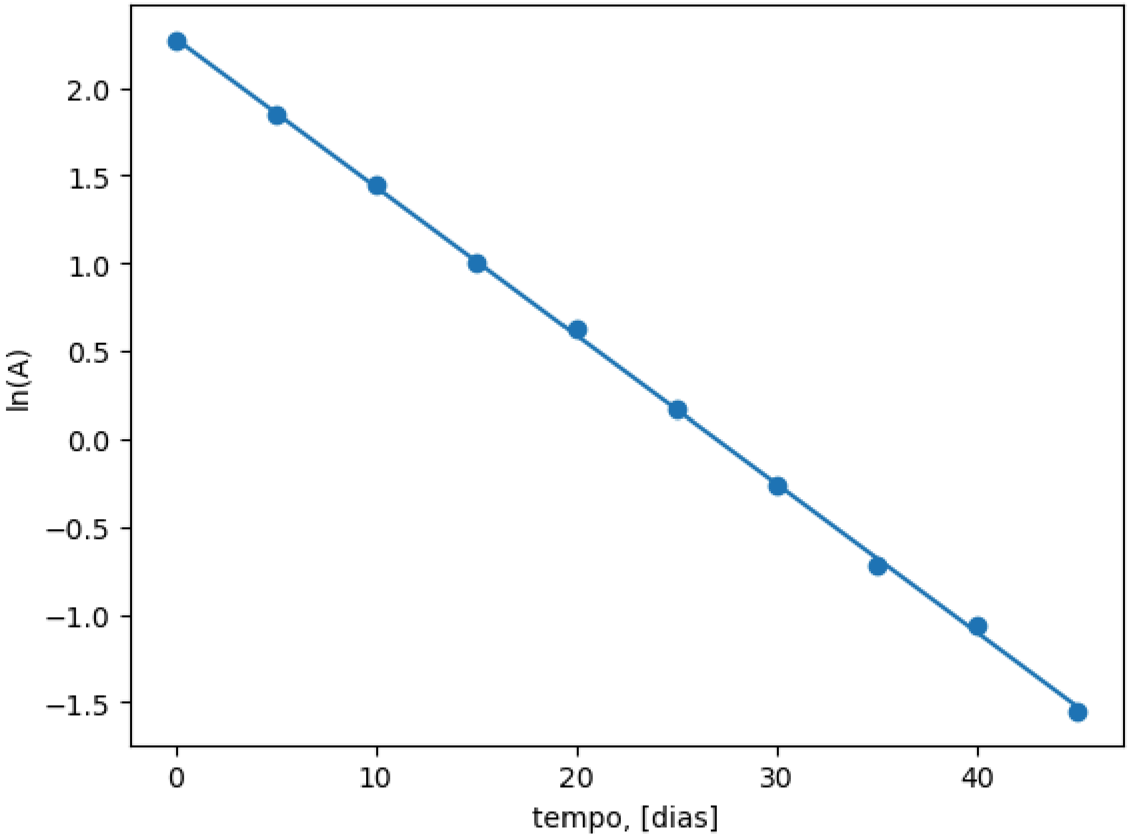
ou de uma forma mais clarificadora:

$$A(t) = A_0\,\mathrm{e}^{-t/\tau}$$

In [14]:
```
# Atenção à diferença entre o logaritmo natural "np.log" e o logarítmo de base 10 "np
X_i = t_i
Y_i = np.log(A_i)

# Calcular melhor reta
m, b, r2, dm, db = minimos_quadrados(X_i, Y_i)

# Imprimir resultados
print("m = {0:.4f}".format(m))
print("b = {0:.2f} cm".format(b))
print("r² = {0:.4f}...".format(r2))
print("Δm = {0:.4f}".format(dm))
print("Δb = {0:.2f} cm".format(db))

# Gerar valores do tempo e espaço percorrido para representação da reta
x = np.linspace(0.0, 45.0, 2)
```

```
y = m * x + b

# Representar dados num grafico (usando o matplotlib)
plt.scatter(X_i, Y_i)
plt.plot(x, y)
plt.xlabel("tempo, [dias]")
plt.ylabel("ln(A)")
plt.show()
```

```
m = -0.0847
b = 2.28 cm
r² = 0.9996...
Δm = 0.0006
Δb = 0.02 cm
```



De facto, a relação entre a atividade radioativa e o tempo decorrido é dada por $A(t) = A_0\, \mathrm{e}^{-t/\tau}$.

$\log A = \log A_0 - t/\tau$

$Y_i = \log A_i$

$X_i = t_i$

$m = -1/\tau$

$b = \log A_0$

```
In [15]: print("tau = -1/m = ", -1 / m, "dias")
```

```
tau = -1/m =  11.81075450149966 dias
```

A metade do tempo de vida $\tau_{1/2}$ relaciona-se com a constante de decaimento τ da seguite forma: Definindo um decaimento da radioativodade para metade $A = A0/2$ quando $t = \tau_{1/2}$, obtemos:

$\tau_{1/2} = \tau \ln 2$

In [16]:
```
print("tau_1/2 = ", -1 / m * np.log(2), "dias")
```

```
tau_1/2 =  8.186591183000171 dias
```

O que é próximo do valor de referência $\tau_{1/2} = 8.0197 \, \mathrm{dias}$