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

# PDE:
#  -u''(x) + a*u(x) = f(x)          x in (0,1)
#              u(0) = 0
#             u'(1) = \alpha*(g_b - u(1))

# parameters
a = 1
f = 3
alpha = 1
gb = 1
n = 10        # elements

# mesh
m = n+1       # nodes
h = 1.0/n
x = np.linspace(0,1,m)

# local stiffness matrix
K_loc = np.zeros((2,2))
A = (1.0/h) * np.array([[ 1,-1], [-1, 1]])
B = (a*h/6) * np.array([[ 2, 1], [ 1, 2]])
K_loc = A+B

# Assembling
K = np.zeros((m,m))
F = np.zeros(m)
for i in range(n):
    K[i:i+2,i:i+2] += K_loc
    F[i:i+2] += np.full(2, f*h/2)

# Boundary conditions
# Dirichlet: u(0) = 0
K[0,:] = 0
K[0,0] = 1
F[0]   = 0
# Neumann: u'(1) = \alpha*(g_b - u(1))
A[-1,-1] += alpha
F[-1] += alpha*gb

u = np.linalg.solve(K, F)
plt.plot(x, u, "-o", label="u_h")
plt.title(f"n = {n} | h = {h} | a = {a} | f(x) = {f} | alpha = {alpha} | gb = {gb}")
plt.xlabel("x")
plt.ylabel("u(x)")
plt.legend()
plt.grid(True)

print("K = ", K)
print("f = ", F)
print("u = ", u)

plt.tight_layout()
plt.savefig("../task_a.png", dpi=300)
plt.show()
```

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

# PDE:
#  -(lambda(x)u'(x))' = 0          x in (0,1)
#                u(0) = 0
#                u(1) = 1
#           lambda(x) = |  1      x in (0,1/sqrt(2))
#                       | 10      x in (1/sqrt(2),1)

# parameters
n  = 2              # elements (must be even)

# mesh
m = n+1             # nodes
nh = int(n/2)       # elements per subdomain
mh = nh+1           # nodes per subdomain
jump = 1/np.sqrt(2)
x1 = np.linspace(0,jump, mh)
x2 = np.linspace(jump, 1, mh)
x = np.concatenate((x1[:-1],x2))
h1 = jump/nh
h2 = (1-jump)/nh

# local stiffness matrix
K_loc1 = ( 1.0/h1) * np.array([[ 1,-1], [-1, 1]])
K_loc2 = (10.0/h2) * np.array([[ 1,-1], [-1, 1]])

# Assembling
K = np.zeros((m,m))
F = np.zeros(m)
for i in range(nh):
    K[i:i+2,i:i+2] += K_loc1
for i in range(nh,n):
    K[i:i+2,i:i+2] += K_loc2

# Boundary conditions
# Dirichlet: u(0) = 0
K[0,:] = 0
K[0,0] = 1
F[0]   = 0
# Dirichlet: u(1) = 1
K[-1,:] = 0
K[-1,-1] = 1
F[-1]   = 1

u = np.linalg.solve(K, F)
plt.plot(x, u, "-o", label="u_h")
plt.title(f"n = {n} (already exact)")
plt.xlabel("x")
plt.ylabel("u(x)")
plt.legend()
plt.grid(True)

print("K = ", K)
print("f = ", F)
print("u = ", u)

plt.tight_layout()
plt.savefig("../task_b.png", dpi=300)
plt.show()
```

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

# Peclet problem:
#  -u''(x) + pu'(x) = 0    x in (0,1)
#              u(0) = 0
#              u(1) = 1

# parameters
p = 70
n_values = [10,20,30,40,70]        # elements

# exact solution
x_exact = np.linspace(0,1,1000)
u_exact = (np.exp(p*x_exact)-1)/(np.exp(p)-1)

for n in n_values:
    # mesh
    m = n+1       # nodes
    h = 1.0/n
    x = np.linspace(0,1,m)

    # local stiffness matrix
    K_loc = np.zeros((2,2))
    A = (1.0/h) * np.array([[ 1,-1], [-1, 1]])
    B = (p/2) * np.array([[ -1, 1], [ -1, 1]])
    K_loc = A+B

    # Assembling
    K = np.zeros((m,m))
    F = np.zeros(m)
    for i in range(n):
        K[i:i+2,i:i+2] += K_loc

    # Boundary conditions
    # Dirichlet: u(0) = 0
    K[0,:] = 0
    K[0,0] = 1
    F[0]   = 0
    # Dirichlet: u(1) = 1
    K[-1,:] = 0
    K[-1,-1] = 1
    F[-1]   = 1

    u = np.linalg.solve(K, F)
    plt.plot(x, u, "-o", markersize=2, label=f"n = {n}")

plt.plot(x_exact, u_exact, "black", label="exact")
plt.title(f"p = {p}")
plt.xlabel("x")
plt.ylabel("u(x)")
plt.legend()
plt.grid(True)

print("K = ", K)
print("f = ", F)
print("u = ", u)

plt.tight_layout()
plt.savefig("../task_c.png", dpi=300)
plt.show()
```