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

# Given parameters
L = 1.2           # Arm length (m)
h = 1.0           # Distance from pivot to actuator base (m)
stroke_max = 1.2  # Max actuator stroke (m)
F_max = 35000     # Max actuator force (N)

# Actuator length as function of theta (degrees)
def actuator_length(theta_deg):
    theta = np.radians(theta_deg)
    return np.sqrt(L**2 + h**2 - 2 * L * h * np.cos(theta))

# Calculate stroke for min and max angles
theta_min = 20  # assumed min arm angle (degrees)
ell_min = actuator_length(theta_min)

# Angle range to test
theta_range = np.linspace(theta_min, 120, 1000)

# Initialize variables to track max height and max weight
max_height = 0
max_weight = 0
opt_theta_height = 0
opt_theta_weight = 0

# Calculate load W = F_max * sin(alpha)
# Where alpha is angle between actuator and arm:
# cos(alpha) = (L^2 + ell^2 - h^2) / (2 * L * ell)
def load(theta_deg):
    theta = np.radians(theta_deg)
    ell = actuator_length(theta)

    # Check stroke limit
    if ell - ell_min > stroke_max:
        return 0  # Invalid angle, stroke exceeded

    # Calculate alpha using law of cosines in triangle formed by arm,
actuator, and offset h
    cos_alpha = (L**2 + ell**2 - h**2) / (2 * L * ell)
    # Clamp cos_alpha between -1 and 1 to avoid numerical errors
    cos_alpha = np.clip(cos_alpha, -1, 1)
    alpha = np.arccos(cos_alpha)

    # Load W = F_max * sin(alpha)
    return F_max * np.sin(alpha)

for theta in theta_range:
    ell = actuator_length(theta)
```

```python
    if ell - ell_min <= stroke_max:
        height = L * np.sin(np.radians(theta))
        w = load(theta)

        if height > max_height:
            max_height = height
            opt_theta_height = theta

        if w > max_weight:
            max_weight = w
            opt_theta_weight = theta

print(f"Max Height: {max_height:.3f} m at θ =
{opt_theta_height:.2f}°")
print(f"Max Weight: {max_weight/1000:.2f} kN at θ =
{opt_theta_weight:.2f}°")

# Calculate and print height and weight for a specific angle
theta_specific_1 = 92.525
height_specific_1 = L * np.sin(np.radians(theta_specific_1))
weight_specific_1 = load(theta_specific_1) / 1000

print(f"\nAt θ = {theta_specific_1:.3f}°:")
print(f"  Height: {height_specific_1:.3f} m")
print(f"  Weight: {weight_specific_1:.2f} kN")

# Calculate and print stroke length for a specific angle
theta_stroke = 92.525
ell_stroke = actuator_length(theta_stroke)
stroke_at_theta = ell_stroke - ell_min

print(f"\nAt θ = {theta_stroke:.3f}°:")
print(f"  Stroke Length: {stroke_at_theta:.3f} m")


# Optional: plot height and load vs theta
plt.figure(figsize=(10,5))
plt.plot(theta_range, L*np.sin(np.radians(theta_range)), label="Height
(m)")
plt.plot(theta_range, [load(t)/1000 for t in theta_range], label="Load
(kN)")
plt.axvline(opt_theta_height, color='blue', linestyle='--',
label=f"Max Height θ={opt_theta_height:.1f}°")
plt.axvline(opt_theta_weight, color='orange', linestyle='--',
label=f"Max Load θ={opt_theta_weight:.1f}°")
plt.axvline(theta_specific_1, color='red', linestyle='-',
label=f"Specific Angle θ={theta_specific_1:.1f}°")

plt.xlabel("Arm Angle θ (degrees)")
plt.ylabel("Height (m) / Load (kN)")
```

```
plt.title("Height and Load vs Arm Angle")
plt.legend()
plt.grid(True)
plt.show()
```

```
Max Height: 1.200 m at θ = 89.97°
Max Weight: 5.00 kN at θ = 95.08°

At θ = 92.525°:
  Height: 1.199 m
  Weight: 4.87 kN

At θ = 92.525°:
  Stroke Length: 1.166 m
```