```python
import numpy as np

def CalcMAC(c_root, taper):
    """
    Calculates the Mean Aerodynamic Chord (MAC) for a trapezoidal wing.

    Parameters:
    c_root: Root chord length in feet (ft)
    taper (float): Taper ratio (tip chord / root chord)

    Returns:
    float: Mean Aerodynamic Chord (MAC) in feet (ft)
    """

    MAC = (2/3) * c_root * ((1 + taper + taper**2) / (1 + taper))
    return MAC

def CalcMAC_Y(b, taper):
    """Calculates the spanwise location of the mean aerodyamic chord

    Parameters:
    c_root: Root chord length in feet (ft)
    taper (float): Taper ratio (tip chord / root chord)

    Returns:
    float: Spanwise location of Mean Aerodynamic Chord (Y_bar) in feet (ft)"""

    Y_bar = (b/6) * (1+2*taper) / (1+ taper)
    return Y_bar



def quarter_chord_distance(x_w, x_h, c_r_w, c_r_h, taper_w, taper_h, sweep_w, sweep_h, b_w,
b_h):
    """
    Computes the quarter-chord positions of a swept wing and a swept horizontal stabilizer.

    Parameters:
        x_w: float - Leading-edge root position of the wing.
        x_h: float - Leading-edge root position of the horizontal stabilizer.
        c_r_w: float - Root chord of the wing.
        c_r_h: float - Root chord of the horizontal stabilizer.
        taper_w, taper_h: float - Taper ratio (tip chord/root chord) of the wing and
stabilizer.
        sweep_w, sweep_h: float - Quarter chord sweep angle (degrees) of the wing and
stabilizer.
        b_w, b_h: float - Span of the wing and stabilizer.

    Returns:
        tuple - Quarter-chord positions of the wing and stabilizer.
    """
    # Convert sweep angles to radians
    sweep_w = np.radians(sweep_w)
    sweep_h = np.radians(sweep_h)

    x_qc_w = x_w + c_r_w / 4 + CalcMAC_Y(b_w,taper_w) * np.tan(sweep_w)
    x_qc_h = x_h + c_r_h / 4 + CalcMAC_Y(b_h,taper_h) * np.tan(sweep_h)


    return x_qc_w, x_qc_h

def NPcalc(x_AC_wnon, CL_ah, CL_aw, eta_h, Sh, Sw, x_AC_hnon, dwash, Cmf_a):
    """Returns the location of the Neutral Point normalized by the MAC of the wing

    Parameters:
        x_AC_wnon: The normalized location of the aerodynamic center of the wing wrt to the
front of the aircraft
```

```
        CL_ah: The Coefficient of lift of the horizontal stabalizer
        CL_aw: The Coefficient of life of the wing
        eta_h: The efficiency of the horizontal stabalizer
        Sh: The area of the horizonal stabalizer in ft^2
        Sw: The area of the wing in ft^2
        x_AC_hnon: The normalzed location of the horizontal stabalizer
        dwash: The change in downwas wrt AOA
        Cmf_a: Pitching moment contribution from the fuselage wrt AOA"""

    num = x_AC_wnon + ((CL_ah/CL_aw)*eta_h*(Sh/Sw)*x_AC_hnon*(1-dwash)) - Cmf_a
    denom = 1+ ((CL_ah/CL_aw)*eta_h*(Sh/Sw)*(1-dwash))

    NP_non = num/denom

    return NP_non

def LiftSlope(AR, eta, sweep, M):
    """Returns the lift of the slope curve in radians of a trapazoidal swept wing
        Parameters:
            AR: Aspect ratio of lifting surface
            eta: Efficiency of lifting surface
            sweep: Quarter chord sweep of the lifting surface
            M: Predicted mach number """

    sweep = np.radians(sweep)

    num = 2* np.pi * AR
    denom = 2 + np.sqrt(((AR/eta)**2)*(1+ (np.tan(sweep)**2) - M**2)+4)

    return num/denom

def downwash(CL_aw, AR_w):
    """Approximates the downwash wrt changing AOA
    Parameters:
        CL_aw: The clope of the lift curve of the wing
        AR_w: The aspect ratio of the wing"""

    dwash = (2* CL_aw) / (np.pi * AR_w)

    return dwash

def FuselagePitchingMomentSlope(K_f, w_f, L_f, S_w, c_bar):

    Cmf_a = K_f*(w_f**2)*L_f/(c_bar*S_w)

    return Cmf_a



###################################### Calculating the Neutral Point
#############################################
x_w_LE = 9   # ft (Root leading-edge of the wing)
x_h_LE = 31.5 # ft (Root leading-edge of the stabilizer)
c_root_w = 5.18  # Root chord of wing in ft
c_root_h = 5.18 # Root chord of horizontal stabalizer in ft
taper_w = 0.7 # Taper ratio of the wing
taper_h = 0.7 # Taper ratio of the horizontal stabilizer
sweep_w = 5 # Quarter-chord sweep of the wing
sweep_h = -5 # Quarter-chord sweep of the horizontal stabalizer
b_w = 44  # ft span of wing
b_h = 44  # ft span of horizontal stabalizer
S = 193.6 # area of the wing in ft^2
Sh = 193.6 # area of the horizontal stabalizer in ft^2
AR_w = 10 # Aspect ratio of the wing
AR_h = 10 # Aspect ratio of the horizontal stabalizer
M = 0.162 # Predicted Mach at working speed
eta = 0.97 # Wing efficiency from metabook
```

```python
eta_h = 0.97 # Horizontal stabalizer efficiency (Yechout)
L_f = 35.5 # Length of the fuselage in ft
w_f = 4.39 # Max width of the fuselage in ft
K_f = 0.344 # From NACA TR 711 based on approximate x_AC_w/L_f = 0.30 (See Figure 8)
SM = 0.40

 # Desired CG

c_bar = CalcMAC(c_root_w, taper_w) # Calculating the MAC

# Calculate locations of the quarter chord location of lifting surfaces wrt front of aircraft
x_w, x_h = quarter_chord_distance(x_w_LE, x_h_LE, c_root_w, c_root_h, taper_w, taper_h,
sweep_w, sweep_h, b_w, b_h)


# Calculate slope of lift curve in radians of lifting surfaces
CL_aw = LiftSlope(AR_w, eta, sweep_w, M)
CL_ah = LiftSlope(AR_h, eta_h, sweep_h, M)

# Calculate the change in downwash wrt to AOA
dwash = downwash(CL_aw, AR_w)

# Normalize quarter chord locations
x_AC_wnon = x_w/c_bar
x_AC_hnon = x_h/c_bar

# Find pitching moment contribution from fuselage
Cmf_a = FuselagePitchingMomentSlope(K_f, w_f, L_f, S, c_bar)

# Calculate normalized NP location
x_AC_non = NPcalc(x_AC_wnon, CL_ah, CL_aw, eta_h, Sh, S, x_AC_hnon, dwash, Cmf_a)

# Calculate needed CG location for SM = 0.25
x_CG_non = x_AC_non - SM

X_AC = x_AC_non*c_bar # current NP location in ft
X_CG = x_CG_non*c_bar # desired CG location in ft

print(f"Non-dimensional Neutral Point location (x_AC_non): {x_AC_non:.4f}")
print(f"Non-dimensional Desired CG location (x_CG_non): {x_CG_non:.4f}")

print(f"Dimensional Neutral Point location (X_AC) in ft: {X_AC:.4f} ft")
print(f"Dimensional Desired CG location (X_CG) in ft: {X_CG:.4f} ft")
```