```python
# -*- coding: utf-8 -*-
"""
Created on Mon Jun 16 10:22:16 2025

@author: ugims
"""

def calcular_todos_hq():
    import numpy as np
    from scipy.interpolate import interp1d

    # Table Data
    T_values = [100, 150, 200, 250, 300, 350, 400, 450, 500, 550,
                600, 650, 700, 750, 800, 850, 900, 950, 1000, 1050, 1100]
    nu_values = [2.00, 4.426, 7.590, 11.44, 15.89, 20.92, 26.41, 32.39, 38.79, 45.57,
                 52.69, 60.21, 68.10, 76.37, 84.93, 93.80, 102.9, 112.2, 121.6, 131.2, 141.8]
    alpha_values = [2.54, 5.84, 10.3, 15.9, 22.5, 29.9, 38.3, 47.2, 56.7, 66.7,
                    76.9, 87.3, 98.0, 109, 120, 131, 143, 155, 165, 177, 195]
    Pr_values = [0.786, 0.758, 0.737, 0.720, 0.707, 0.700, 0.690, 0.686, 0.684, 0.683,
                 0.685, 0.690, 0.695, 0.702, 0.709, 0.716, 0.711, 0.707, 0.705, 0.702, 0.700]
    k_values = [9.34, 13.8, 18.1, 22.3, 26.3, 30.0, 33.8, 37.3, 40.7, 43.9,
                46.9, 49.7, 52.4, 54.9, 57.3, 59.6, 62.0, 64.3, 66.5, 69.0, 71.5]

    # Interperlators
    nu_interp = interp1d(T_values, nu_values, kind='linear')
    alpha_interp = interp1d(T_values, alpha_values, kind='linear')
    pr_interp = interp1d(T_values, Pr_values, kind='linear')
    k_interp = interp1d(T_values, k_values, kind='linear')

    # Geometric and Thermic data (in -> mm, BєC)
    print("Insert new dimensions  --In Milimeters--:")
    altura = float(input("Height of plate (mm): ")) / 1000
    largura = float(input("Width of plate (mm): ")) / 1000
    comprimento = float(input("depth of plate (mm): ")) / 1000
    Ts_C = float(input("Surface temperature (B°C): "))
    Tamb_C = float(input("Ambient temperature (B°C): "))

    Ts = Ts_C + 273.15
    Tamb = Tamb_C + 273.15
    Tfilm = (Ts + Tamb) / 2
    delta_T = Ts - Tamb
    g = 9.81
    beta = 1 / Tfilm

    # Properties Interpolation
    nu = nu_interp(Tfilm) * 1e-6
    alpha = alpha_interp(Tfilm) * 1e-6
    Pr = pr_interp(Tfilm)
    k = k_interp(Tfilm) * 1e-3

    # Lengths and Areas
    A = largura * comprimento
    P = 2 * (largura + comprimento)
    L_horizontal = A / P
    L_vertical = altura

    def calcular_ral_nul_hq(L, tipo):
        RaL = (g * beta * delta_T * L**3) / (nu * alpha)
        if tipo == "vertical":
            NuL = 0.68 + (0.670 * RaL**(1/4)) / ((1 + (0.492 / Pr)**(9/16))**(4/9))
        elif tipo == "horizontal_sup":
            NuL = 0.54 * RaL**(1/4) if RaL <= 1e7 else 0.15 * RaL**(1/3)
        elif tipo == "horizontal_inf":
            NuL = 0.27 * RaL**(1/4)
        else:
            return None, None, None
        hq = NuL * k / L
        return RaL, NuL, hq

    # Main results
    print("\n--- General Results ---")
    print(f"T_film: {Tfilm:.2f} K")
    print(f"OS = {nu:.2e} mBI/s | O± = {alpha:.2e} mBI/s | Pr = {Pr:.4f} | k = {k:.4f} W/mB·K")

    Ra_v, Nu_v, hq_v = calcular_ral_nul_hq(L_vertical, "vertical")
    print("\n[1] Vertical:")
    print(f"  L = {L_vertical:.4f} m")
    print(f"  Ra_L = {Ra_v:.2e} | Nu_L = {Nu_v:.2f} | h_q = {hq_v:.2f} W/mBIB·K")

    Ra_hs, Nu_hs, hq_hs = calcular_ral_nul_hq(L_horizontal, "horizontal_sup")
    print("\n[2] Horizontal - upper hot surface (or lower cold):")
    print(f"  L = {L_horizontal:.4f} m")
    print(f"  Ra_L = {Ra_hs:.2e} | Nu_L = {Nu_hs:.2f} | h_q = {hq_hs:.2f} W/mBIB·K")

    Ra_hi, Nu_hi, hq_hi = calcular_ral_nul_hq(L_horizontal, "horizontal_inf")
    print("\n[3] Horizontal - lower hot surface (ou upper cold):")
    print(f"  L = {L_horizontal:.4f} m")
    print(f"  Ra_L = {Ra_hi:.2e} | Nu_L = {Nu_hi:.2f} | h_q = {hq_hi:.2f} W/mBIB·K")

  # Calculation for additional details
    resposta = input("\nAny additional mould details (y/n): ").strip().lower()
    if resposta == 'y':
        num = int(input("How many details? "))
        for i in range(num):
            print(f"\n[Details {i+1}]")
            tipo = input("Orientation (vertical / horizontal_sup / horizontal_inf: ").strip().lower()
            if tipo == "vertical":
                L = float(input("Detail's height (mm): ")) / 1000
            else:
                largura_d = float(input("Detail's width (mm): ")) / 1000
                comprimento_d = float(input("Detail's depth (mm): ")) / 1000
                A_d = largura_d * comprimento_d
                P_d = 2 * (largura_d + comprimento_d)
                L = A_d / P_d

            Ra, Nu, hq = calcular_ral_nul_hq(L, tipo)
            print(f"  L = {L:.4f} m")
            print(f"  Ra_L = {Ra:.2e} | Nu_L = {Nu:.2f} | h_q = {hq:.2f} W/mBIB·K")

    # NEW FUNCTIONALITY: lower channel with lower heating (formula 9.47)
    resposta_canal = input("\nCalculate convention coefficient for a lower closed horizontal channel? (y/n): ").strip().lower()
    if resposta_canal == 'y':
        print("\n--- Natural convention calculation (horizontal channel вЂ” formula 9.47 by Azevedo & Sparrow) ---")
        try:
            S = float(input("Spacing between plates S (mm): ")) / 1000
            L_canal = float(input("Channel depth L (mm): ")) / 1000

            # Number of Rayleigh based in S
            Ra_S = (g * beta * delta_T * S**3) / (nu * alpha)
            ratio_SL = S / L_canal
            Nu_S = 0.645 * (Ra_S * ratio_SL)**0.25
            h_canal = (Nu_S * k) / S
```

```
            print(f"\n[Canal horizontal]")
            print(f"  Ra_S = {Ra_S:.2e}")
            print(f"  S/L = {ratio_SL:.4f}")
            print(f"  Nu_S = {Nu_S:.2f}")
            print(f"  h (coef. de convecГ§Гo) = {h_canal:.2f} W/mBIB·K")
        except Exception as e:
            print(f"Erro no cГЎlculo do canal horizontal: {e}")

# Main Function:
calcular_todos_hq()
```