Elastic properties:
$Y_A = 67$ GPa
$Y_M = 26$ GPa

Transformation Temperature
$M_f = 9°C$
$M_s = 18°C$
$A_s = 35°C$
$A_f = 49°C$

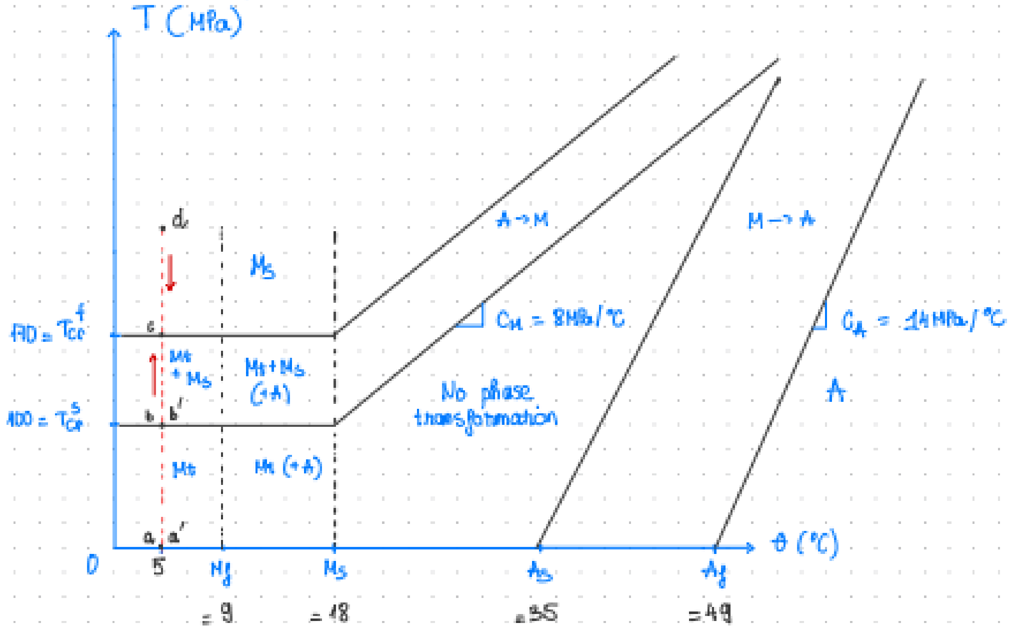
Transformation Constraints
$C_M = 8$ MPa/°C
$C_A = 14$ MPa/°C
$\tau_{cr}^{s} = 100$ MPa
$\tau_{cr}^{f} = 170$ MPa

Max Recovery Strain
$\varepsilon_L = 0.07$

Material Heat
$\theta = 5°C$

① What is the SMAs behavior at the given?
$\theta = 5°C < M_f = 9°C \Rightarrow$ The SMA behavior can be shape memory effect (SME)

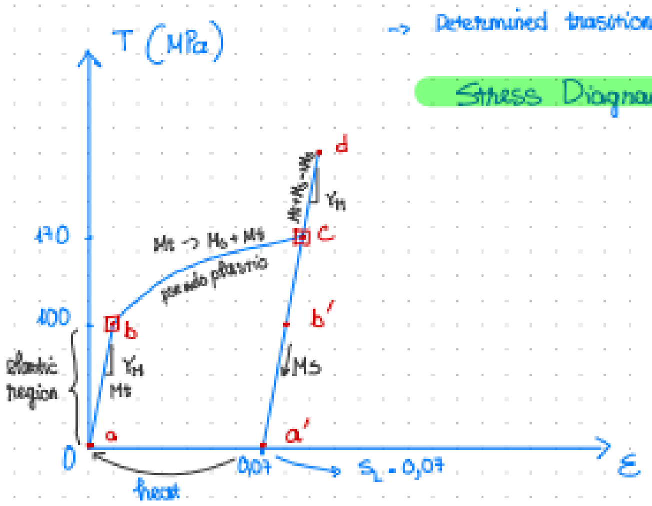
② Draw stress cycle on phase diagram? (Phase Diagram)



③ Sketch graphically T-S curve with different specific regions?

→ Determined transitional points: b, c

Stress Diagram



④ Evaluate stress at the transitional points in the T-S curve?

- $\tau_b = \tau_{cr}^{s} = 100$ MPa
- $\tau_c = \tau_{cr}^{f} = 170$ MPa

⑤

Code File:

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

class DataInput:
    def __init__(self, YA, YM, Ms, Mf, As, Af, CM, CA, TCRS, TCRF, SL, temp, shiS0, shiT0):
        self.YA = YA
        self.YM = YM
        self.Ms = Ms
        self.Mf = Mf
        self.As = As
        self.Af = Af
        self.CM = CM
        self.CA = CA
        self.TCRS = TCRS
        self.TCRF = TCRF
        self.SL = SL
        self.temp = temp
        self.shiS0 = shiS0
        self.shiT0 = shiT0
        self.shi_lst = []
    def stress_calculated(self):
        global Ta, Tb, Tc
        Ta = 0
        Tb = self.TCRS
        Tc = self.TCRF

    def predicted_through_stress(self):
        global Tb_predicted, Tc_predicted, Td_predicted
        Tb_predicted = np.linspace(Ta, Tb, 20)
        Tc_predicted = np.linspace(Tb, Tc, 20)
        for i in Tc_predicted:
            shiS = ((1 - self.shiS0) / 2) * math.cos((math.pi / (self.TCRS - self.TCRF)) * (i - self.TCRF)) + ((1 + self.shiS0) / 2)
            self.shi_lst.append(shiS)
        self.shi_lst = np.array(self.shi_lst)
        Td_predicted = np.linspace(Tc,Ta,num=20, endpoint=True)
        return Tb_predicted, Tc_predicted, Td_predicted
```

```
    def predicted_through_strain(self):
        global Sb_predicted, Sc_predicted, Sd_predicted
        Sb_predicted = Tb_predicted/self.YM
        Sc_predicted = (Tc_predicted/self.YM) + self.SL*self.shi_lst
        Sd_predicted = (Td_predicted/self.YM) + self.SL
        return Sb_predicted, Sc_predicted, Sd_predicted

    def plot_T_S_curve(self):
        strain_combined = np.concatenate((Sb_predicted, Sc_predicted, Sd_predicted))
        stress_combined = np.concatenate((Tb_predicted, Tc_predicted, Td_predicted))

        plt.figure(figsize=(8, 6))
        plt.plot(strain_combined, stress_combined, label='Stress-Strain Curve', color='blue')
        plt.xlabel('Strain')
        plt.ylabel('Stress (MPa)')
        plt.title('Stress-Strain Relationship')
        plt.legend()
        plt.grid(True)
        plt.show()

def main():
    Mf = 9  # °C
    Ms = 18  # °C
    As = 35  # °C
    Af = 49  # °C
    CM = 8  # MPa/°C (slope for martensite)
    CA = 14  # MPa/°C (slope for austenite)
    YM = 26000
    YA = 67000
    TCRS = 100  # MPa (start transformation stress)
    TCRF = 170  # MPa (finish transformation stress)
    shiT0 = 1
    shiS0 = 0
    SL = 0.07
    temp = 5
    proc = DataInput(YA, YM, Ms, Mf, As, Af, CM, CA, TCRS, TCRF, SL, temp, shiS0, shiT0)
    proc.stress_calculated()
    proc.predicted_through_stress()
```

```
    proc.predicted_through_strain()
    proc.plot_T_S_curve()
if __name__ == "__main__":
    main()
```

Result: