1. Give pseudocode for PID control. You do not have to worry about integrator anti-windup. Your code does not have to copy the notes; it only needs to correctly implement PID control. The beginning of the code is given to you.

\[ E_{\text{integrated}} = 0; \quad e_{\text{previous}} = 0; \]

Every \( dt \) seconds do {

- Take measurement (1 pt)
- Ref val (1 pt)
- Calculate error (ref-measurement) (2 pts.)
- Calculate \( d_{\text{error}} = (e - e_{\text{previous}})/dt \) (2 pts.)
- Calculate \( e_{\text{integrated}} = e_{\text{integrated}} + \text{error} \); (2 pts.)
- Calculate \( u = kp \times d + ki \times e_{\text{integrated}} + kd \times (e - e_{\text{previous}}) \) (2 pts.)
- Send control (1 pts.)
- Update \( e_{\text{previous}} = e \) (1 pt.)

2. Explain what integrator anti-windup is.

Sets max/min value for \( e_{\text{integrated}} \) to limit the oscillation caused when it is allowed to build up to a large number. (2 pts.)

3. To turn a PID controller into a PD controller, what do you do?

Set \( Ki = 0; \) (2 pts.)

4. You have chosen gains KP and Kd for a PD controller
   a. Your overshoot is too large. Which gain do you increase?
      i. \( Kd \) (2 pts.)
   b. Your steady-state error is too large. Which gain do you increase? (Ki would work if it was a PID controller) (2 pts.)