Task-1: Robot Simulation:

- Derive dynamic equation for manipulator.
- Solve equations for acceleration and use these to create state-space simulator. (This should include graphic simulation of robotic manipulator.)
- Use robot manipulator initial angles of 10,20, and 30 degrees respectively.
- Use individual link lengths of 4, 3, and 2 meters respectively.
- Use individual link masses of 20, 15, and 10 kg respectively.
- Use a-axis inertial values of 0.5, 0.2, and 0.1 kg-m^2 respectively.
- Demonstrate dynamics of the robot through simulation.
    1. No actuator torque, no gravity
    2. No actuator torque, with gravity
    3. Joint-1 actuator torque in order to provide equilibrium, with gravity
    4. Joint-1 and Joint-2 actuator torque in order to provide equilibrium, with gravity
    5. Joint-1, Joint-2, and Joint-3 actuator torque in order to provide equilibrium, with gravity
- All simulations should present graphs of x,y, and alpha end-effector positions vs. time.
- Include any other graphs, gif's, etc. which you deem necessary to describe the demonstration.

Task-2 Robot simulation – with control partitioning

- Design a closed loop system by partitioning the robot dynamics into model-based and servo-based dynamics.
- Simulate the system by settting the initial joint angles to 10, 20, and 30 degresss respectively. Explain what happens during the simulation.
- Add joint-position control by adding an input vector which represents the joint desired positions. (Hint: This vector should be subtracted from the actual joint positions to form and error signal which should be subsequently fed into the K_p gain block.)
- Simulate the system with the above initial angeles and appropriate input command angles.
- One joint at a time, apply a step input change to the command angle. Monitor the response of the ssytem by examining the error signal.
- Experiment with various K_p gains (such as 1, 10, 100, etc.) and explain the results.
- How does the rise time change?
- Can you set the gain to give overshoot?
- Does the system need different K_p values for each joint?
- Does the step response look like a second order system?

Task-3: Robot simulation – path trajectory

Most industrial robot paths are programmed by teaching desired points along the path. This is done by using the teach pendant to maneuver to some desired point and orientation and then capturing the joints coordinate values. Given these joint coordinates and some desired time line we can create a reasonable trajectory through these points.

Given: The following timeline data

Time (seconds)	Theta-1 (degrees)	Theta-2 (degrees)	Theta-3 (degrees)
0	0	0	90
2	30	-10	70
4	45	130	-85
6	150	10	70
8	180	0	-90

- Create a spline fit curve through each set of points for each joint. Make sure the starting and ending slopes of the trajectory (joint velocity) are zero. (Use the spline function in MATLab)
- Write a function to feed the three trajectory curves to your RRR robot simulator.
- Set initial angles of system to match those at time zero and simulate the path for 8 seconds.
- Does the simulation make it to the end?
- Does the value of the selected gains (K_p and K_v) change the simulation?
- Does the actual path follow the designed trajectory?