Week 1:

Describe at least 2 desirable unit tests and what "cuts" would be needed (see lecture topic 2.5), that would cover main paths and/or boundary condition handling, explaining how you could unit test utilising these cuts.

1. Unit Test for Drone's Energy Consumption by Disruptor Activation

Desired Unit Test (UT): Verify that the energy consumption when activating the quantum disruptor deducts the correct amount of energy from the drone's energy store.

- Cut Identified: The cut isolates the energy deduction logic upon disruptor activation from the actual user input and physical energy store update mechanism. This cut allows testing the logic that calculates and applies the energy deduction without external dependencies.
- Using the Cut: The unit test would mock the disruptor activation process, providing a predefined energy level in the drone's energy store and simulating the activation. The test checks if the energy store's value decreases by the expected amount upon activation.
- Isolation: This UT isolates the disruptor's energy consumption calculation from the physical button press for activation and the real-time energy recharge mechanism. It specifically targets the logic that handles energy deduction to ensure accuracy and reliability in energy management.

2. Unit Test for Wall Collision Detection

Desired Unit Test (UT): Ensure the collision detection system accurately identifies when the drone collides with a wall within the maze.

- Cut Identified: The cut for this test isolates the collision detection logic from the actual movement and position update system of the drone. By focusing on collision detection, we avoid the complexities of real-time drone movement and environmental interactions.
- Using the Cut: This unit test would simulate the drone being at various positions relative to a wall (including edge cases like barely touching or just missing the wall) and verify that the collision detection logic correctly identifies whether a collision has occurred.
- Isolation: The UT isolates the wall collision detection logic from the rest of the drone's navigation and movement control system. It aims to ensure that the method responsible for determining collisions is accurate, regardless of how the drone's position is updated in the broader system.

Where the project stands:

This week I did the project planning, created my task diagram, and identified 2 cutting points for unit testing. I also created a work log in excel to map out work ahead of me and track time put into the project for each week. I also created a risk register to identify what I think the risks with the highest impact/probability will be. In upcoming weeks I will be updating the work log and risk register once I start working on the project. As of this week I am not entirely sure how I want to start the implementation so I am keeping things outlined but not very specific. I estimate that the project will take me about **57.5** hours in total to complete. Each week I estimate that I have **11.5** hours of work to stay on track. So far I have completed **0%** of my estimated work in **100%** of the budgeted total project time. For the work that has been completed, I took **0x** (**0**/**57.5**) as much time as I estimated for the total and **0x** (**0**/**11.5**) as much time as I estimated weekly. Next week I would like to get started on the Physics task and disruptor task as I predict these will be the most important to get up and running. Then I can implement the Map task and LED task. I probably will want to get a basic LCD task up and running as well just to see what is happening on the display.