2511 Testing Plan

Before writing any functional code for the system we will write tests that will outline how we intend it to operate.

We will write unit tests that ensure each module is working in a self contained environment, checking that any operations it performs produce the expected result. An example of a unit test could be that if the player class had a method to move with a direction parameter, inputting 'DOWN' increases the player's y co-ordinate by 1.

We will write integration tests that ensure multiple modules of the system work well together. We expect these to pass when all the unit tests pass since the individual functionality should be assured for the combined functionality. An example of an integration test could be that if the player is supposed to pick up a potion when they share co-ordinates, if a player moves down from the co-ordinate above the potion they will pick it up.

We will write system tests that ensure the complete functionality of the system as a whole. We expect these to pass when all unit and integration tests pass. An example of a system test would be testing a complete game including all the players steps from starting the dungeon to either completing it or failing it.

Our tests will test for all input edge cases and ensure appropriate errors/exceptions are thrown when invalid input is given. We will also attempt to maintain the highest possible code coverage to improve code style and performance.