testing.pdf

Holly's tests: The vast majority of my tests below are integration tests (unless specified otherwise) because they depend on player movement, boulders, walls, etc.

Ticket: spider spawn and movement

- Test spider spawning [all tests below are unit tests unless specified otherwise]:
    - Test spider doesn't spawn on a boulder [integration test]
    - Test that the correct number of spiders spawn when spawn_rate is 1 tick
    - Test that the correct number of spiders spawn when spawn_rate is 0 ticks. This is a [Usability test] as well.
    - Test that the correct number of spiders spawn when spawn_rate is 5 ticks and ensure they spawn within the map boundaries
- Test spider movement [all tests below are integration tests]:
    - Test spider running into boulder and then reversing direction
    - Test a combination of spiders running into boulders multiple times, resulting in them reversing directions many times [Also a usability test]
    - Test spider can't move at all if there is a boulder above it
    - Test spider can move if a boulder above it has been pushed by the player

Ticket: zombie toast spawn and movement

- Test zombie toast spawn [all tests below are integration tests since they depend on zombie_toast_spawner]:
    - Test zombies spawn on a cardinally adjacent (up, down, left, right) "open square". Also ensure that no zombies can spawn on top of the spawner.
    - Test zombies do not spawn from a zombie toast spawner if they are completely surrounded by walls, boulders, locked doors.
    - Test that the correct number of zombies spawn when zombie_spawn_rate = 0. This is a [Usability test] as well.
    - Test that the correct number of zombies spawn when zombie_spawn_rate = 1
    - Test that the correct number of zombies spawn when zombie_spawn_rate = 10
    - Test that zombies can't spawn without a zombie spawner. Also, test that a zombie can already exist in the dungeon JSON map.
    - Test that zombies can spawn from many different spawners.
- Test zombie movement [integration tests]:
    - Test zombies can't move into a wall, boulder, locked doors (i.e. testing that zombies exhibit player movement behaviour)
    - Test zombies can move through open doors, everything else (note: this is commented out because the implementation for doors/keys is incomplete)
    - Test zombies can go up, down, left, right or stay at their current location [unit test]
    - Test that zombies can only spawn and move in one position. I.e. there is only one "open square."

Ticket: mercenary movement [integration tests]

- Test that the mercenary can't walk into boulders, closed doors and walls
- Test mercenary movement when they are close to the player. This also tests the mercenary's enemy movement, where mercenaries can only move towards the player.
- Test that the mercenary can walk through open doors (note: this is commented out because the implementation for doors/keys is incomplete)
- Test mercenary ally movement. This is also a [Usability test]: test a mercenary's movement after the player collects 3 coins and bribes the mercenary on the frontend.

Ticket: enemy battling [integration tests]

- Test that a zombie toast losing against an unarmed player
- Test that a spider losing against an unarmed player
- Test that a zombie can walk through an open door and meet up with the player to battle. The zombie wins.
- Please note: enemy mercenaries winning and losing against a player has already been tested in ExampleTests.java.
- Test ally mercenaries provide attack and defence bonus for the player
- [System test] Test the following:
    - 1. Player collects 3 treasure
    - 2. Player bribes the mercenary
    - 3. Player battles and wins against a zombie
    - 4. Player pushes a boulder
    - 5. Player exits and wins.
    - Please note: I would've liked to test the player crafting a bow/shield, but those features haven't been implemented unfortunately.

<u>Joseph's Tests:</u> All tests are integration tests unless specified because many of the tests require other functionality to be implemented (i.e movement, inventory,etc).

Collectable Entities:
Ticket Arrows:

- Test arrow is placed into a map
- Test arrow can be picked up

Ticket Bomb:

- Bomb can be placed into map (unit testing)
- Bomb can be picked up
- Bomb has a variable radius (Unit testing)
- Bomb removes entities when it explodes
- Bomb Does not remove entities out fo its range (System testing)
- Bomb does not activate unless next to an active switch (System testing)
- Bomb does not remove player (System testing)

Ticket Key:

- Key can be placed into map (unit testing)
- Key can be picked up
- Key disappears when used in crafting or door (System testing)

Ticket Sword:

- Sword can be placed into map (unit testing)
- Sword can be picked up
- Sword has durability thats decreases after battle (System testing)
- Sword has additive effect to player damage (System Testing)

Ticket Treasure:

- Treasure can be placed into map (unit testing)
- Treasure can be picked up

Ticket Wood:

- Wood can be placed into map (unit testing)
- Wood can be picked up

Ticket Invisibility Potion:

- Pot can be placed into map (unit testing)
- Pot can be picked up
- Pot can be used
- Pot causes battles not to occur (System Testing)
- Pot causes mercenary movement to be random (System Testing)

Ticket Invincibility Potion:

- Pot can be placed into map (unit testing)
- Pot can be picked up
- Pot can be used
- Pot causes Zombies and mercenaries to run away (System Testing)
- Pot causes player not to take damage in rounds of battle (System Testing)
- Test exceptions

Laura's Tests - unit and integration tests are indicated at the start of each line.

Ticket Player Movement

- [Unit] Test Player creation response.
- [Unit] Player moves up.
- [Unit] Player moves down.
- [Unit] Player moves right.
- [Unit] Player moves left.
- [Unit] Player can't move through walls.
- [Integration] Player auto-pickup items.

Ticket Mercenary Bribery

- [Unit] Invalid id given to mercenary bribery function.
- [Unit] Mercenary is too far away to bribe.
- [Unit] Insufficient gold to bribe mercenary.
- [Integration] Mercenary bribery success.

Ticket Player Battling

- [Integration] Player battles monster with correct calculations - player dies.
- [Integration] Player battles monster with correct calculations - player wins.
- [Integration, untested] Player attacks with sword bonus calculations.
- [Integration, untested] Player attacks with bow bonus calculations.
- [Integration, untested] Player attacks with sword and bow bonus calculations.
- [Integration, untested] Player defence aided by shield calculations.
- [Integration] Player attacks with aid of ally calculations (both atk and def).

Ticket Zombie Spawner Destruction

- [Unit] Player is not cardinally adjacent to spawner.
- [Unit] Player tries to destroy without sword.
- [Unit] Player succeeds in destroying spawner.

Ticket Statistics and Basic Goals

- [Integration] Response when no goals have been completed.
- [Integration] Response when enemies goal has been completed.
- [Integration] Response when enemies (with spawner) goal has been completed.
- [Integration] Response when treasure goal has been completed.
- [Integration] Boulders goal completed, and that it can become incomplete again.
- [Integration] Response when exit goal has been completed.
- [Integration] Response when large treasure goal has been completed.

Ahmed's Tests:

Ticket: Wall

- [Unit] Test wall has been created.
- [Integration] Tests if walls block player movement.

Ticket: Boulder

- [Integration] Tests if a player can push a boulder to the right.
- [Integration] Tests if a player can push a boulder up.

Ticket: FloorSwitch

- [Integration] Tests whether or not the floor switch gets activated when a boulder is pushed on to it and ONLY when it is pushed on to it.
- [Integration] Tests whether or not the floor switch gets deactivated once a boulder that has already been pushed on to it is pushed off of it.

Ticket: Exit

- [Unit] Test exit has been created.
- [Integration] Tests whether or not the exit state of the exit changes once a player steps into it.

Ticket: Portal

- [Unit] Tests portal has been created.

- [Integration] Tests if a player successfully teleports to the correct portal pair out of two pairs of portals and into the correct cardinally adjacent position.
- [Integration] Tests the same as above but with a different pair of portals.

Ticket: Door
- [Unit] Tests door has been created.
- [Integration] Tests door and key interaction through creating two door and key pairs and then testing individually if the player can walk through the doors with and without the appropriate keys.

Luke's Tests:

Ticket: Bow
- [Integration] Testing that a bow has been created.

Ticket: Shield
- [Integration] Testing that a shield can be made from 2 wood and 1 treasure
- [Integration] Testing that a shield can be created from 2 wood and 1 key