

SE 3XA3: Test Plan Rogue Reborn

Group #6, Team Rogue++

| | |
|-------------------|---------|
| Ian Prins | prinsij |
| Mikhail Andrenkov | andrem5 |
| Or Almog | almogo |

Due Wednesday, Dec 7th, 2016

Contents

| | | |
|----------|--|-----------|
| 1 | Functional Requirements Evaluation | 4 |
| 2 | Nonfunctional Requirements Evaluation | 5 |
| 2.1 | Usability | 5 |
| 2.2 | Performance | 5 |
| 2.3 | etc. | 5 |
| 3 | Comparison to Existing Implementation | 6 |
| 4 | Unit Testing | 7 |
| 5 | Changes Due to Testing | 8 |
| 6 | Automated Testing | 9 |
| 6.1 | Automated Testing Strategy | 9 |
| 6.2 | Specific System Tests | 9 |
| 6.3 | Automated Testing Strategy | 9 |
| 6.4 | Specific System Tests | 9 |
| 7 | Trace to Requirements | 24 |
| 8 | Trace to Modules | 25 |
| 9 | Code Coverage Metrics | 27 |

Table 1: **Revision History**

| Date | Version | Notes |
|----------|---------|------------------------------------|
| 12/06/16 | 0.1 | Initial Draft |
| 12/06/16 | 0.2 | Automated Tests To PlayerChar |
| 12/06/16 | 0.3 | Functional Requirements Evaluation |

List of Tables

| | | |
|---|-----------------------------|----|
| 1 | Revision History | 1 |
| 3 | Module Hierarchy | 25 |
| 4 | Test-Module Trace | 26 |

List of Figures

1 Functional Requirements Evaluation

Overall, an evaluation of functional requirements reveals near, if not complete coverage. The tests written for the projects turned out to be quite useful, as many caught bugs or business-errors that would have otherwise gone unnoticed. Those will be discussed below. As for the rest of the functional requirements, many were mundane, general, or crucial enough to have already been satisfied earlier. Those will not be discussed, as their complete satisfaction has already been verified countless times.

The list below refers to each functional requirement by its numerical identifier, as listed in the System Requirements Specification. Please refer to the SRS if any confusion arises due to this.

FR.16: When performing level tests, a strange anomaly led to one test constantly failing. The test revealed that the player, in fact, did not begin at the first level. Due to an off-by-one error and slight miscommunication between developers, the current level depth the player was on was i in some places and $i + 1$ in others. As soon as the test revealed this, the problem was remedied globally.

FR.19: Whenever the player uncovers a new dungeon level (including the very first level), an algorithm decides on a position in which to place the user initially. This algorithm while appearing flawless, actually had a very slight chance of placing the player in an unreachable location, surrounded by walls, doomed forever. With the automatic tests running thousands upon thousands of simulations, the bug was quickly revealed, and remedied.

FR.39: Working with C++ has its benefits, but also its drawbacks. An anomaly in the way C++ handles integers revealed a very serious bug in the code, in which player armor could reach utterly ridiculous values, rendering the player effectively invincible. By simulating every possibility of armor that can be made, this bug was caught and patched. To elaborate, the reason the bug even existed was because an unsigned integer was allowed to be reduced to a negative value, which of course means that it was not reduced to a negative number and instead went to the highest value an integer can be.

2 Nonfunctional Requirements Evaluation

Mikhail

2.1 Usability

Mikhail

2.2 Performance

Mikhail

2.3 etc.

Mikhail

3 Comparison to Existing Implementation

Ori

4 Unit Testing

Mikhail

5 Changes Due to Testing

Mikhail

6 Automated Testing

6.1 Automated Testing Strategy

For this project we elected not to use a 3rd party testing library. We made this decision to ease configuration/installation problems and reduce our dependencies, as we judged it would not be necessary. Instead a series of files (labeled `test.foobar.cpp`) in the repository hold tests, which are run by our custom test runner. These automated tests are run on command by executing the produced executable, or by the continuous integration script run whenever changes are pushed to the central repository. The results of these tests are automatically reported, resulting in a failed or successful build.

6.2 Specific System Tests

The following is a list of all system tests in the project.

6.3 Automated Testing Strategy

For this project we elected not to use a 3rd party testing library. We made this decision to ease configuration/installation problems and reduce our dependencies, as we judged it would not be necessary. Instead a series of files (labeled `test.foobar.cpp`) in the repository hold tests, which are run by our custom test runner. These automated tests are run on command by executing the produced executable, or by the continuous integration script run whenever changes are pushed to the central repository. The results of these tests are automatically reported, resulting in a failed or successful build.

6.4 Specific System Tests

The following is a list of all system tests in the project.

| | |
|-------------------------|--------------------------------------|
| Name: | Amulet Construction |
| Initial State: | None |
| Input: | Coordinate, context value |
| Expected Output: | Amulet object in valid initial state |

| | |
|-------------------------|--|
| Name: | Armor Construction 1 |
| Initial State: | None |
| Input: | Coordinate |
| Expected Output: | Armor object in valid initial state |
| Name: | Armor Construction 2 |
| Initial State: | None |
| Input: | Coordinate, context value, type value |
| Expected Output: | Armor object in valid initial state |
| Name: | Armor Identification |
| Initial State: | Cursed Armor |
| Input: | None |
| Expected Output: | Verification that armor is identified |
| Name: | Armor Identification |
| Initial State: | Cursed Armor |
| Input: | None |
| Expected Output: | Verification that armor is identified |
| Name: | Armor Curse |
| Initial State: | Cursed Armor |
| Input: | None |
| Expected Output: | Verification that armor is cursed |
| Name: | Armor Enchantment |
| Initial State: | Cursed Armor |
| Input: | Curse level |
| Expected Output: | Verification that armor enchantment is correct |
| Name: | Armor Rating |
| Initial State: | Cursed Armor |
| Input: | None |
| Expected Output: | Verification that armor rating is correct |
| Name: | Coordinate Ordering |
| Initial State: | None |
| Input: | (0,0) coordinate and (1,1) coordinate |
| Expected Output: | Verification that (0,0) \leq (1,1) |
| Name: | Coordinate Equality |
| Initial State: | None |
| Input: | Two (0,0) coordinates |
| Expected Output: | Verification that the two inputs are equal |
| Name: | Coordinate Inequality |

| | |
|-------------------------|---|
| Initial State: | None |
| Input: | (0,0) coordinate and (1,1) coordinate |
| Expected Output: | Verification that the two inputs are not equal |
| Name: | Coordinate Addition |
| Initial State: | None |
| Input: | (2,3) coordinate and (1,2) coordinate |
| Expected Output: | (3,5) coordinate |
| Name: | Coordinate Subtraction |
| Initial State: | None |
| Input: | (2,3) coordinate and (1,2) coordinate |
| Expected Output: | (1,1) coordinate |
| Name: | Feature Construction |
| Initial State: | None |
| Input: | Symbol, coordinate, visibility, color |
| Expected Output: | Feature object in valid initial state |
| Name: | Feature Symbol Check |
| Initial State: | Feature with given symbol |
| Input: | Symbol |
| Expected Output: | Verification that feature's symbol matches given |
| Name: | Feature Invisibility Check |
| Initial State: | Invisible feature |
| Input: | None |
| Expected Output: | Verification that feature is invisible |
| Name: | Feature Visibility Check |
| Initial State: | Visible feature |
| Input: | None |
| Expected Output: | Verification that feature is visible |
| Name: | Feature Location Check |
| Initial State: | Feature with given location |
| Input: | Coordinate |
| Expected Output: | Verification that feature's location matches given coordinate |
| Name: | Food Construction |
| Initial State: | None |
| Input: | Coordinate and context value |
| Expected Output: | Food object in valid initial state |
| Name: | Food Eating |
| Initial State: | Food and player objects |

| | |
|-------------------------|--|
| Input: | None |
| Expected Output: | Verification that food has increased the player's food life by an appropriate amount |
| Name: | GoldPile Construction |
| Initial State: | None |
| Input: | Coordinate, gold amount value |
| Expected Output: | GoldPile object in valid initial state |
| Name: | GoldPile Quantity Check |
| Initial State: | GoldPile with given amount of gold |
| Input: | Amount of gold value |
| Expected Output: | Verification that gold's amount matches given amount |
| Name: | Item Construction 1 |
| Initial State: | None |
| Input: | Symbol, coordinate, context value, item class specifier, name value, psuedo name |
| Expected Output: | Item object in valid initial state |
| Name: | Item Construction 2 |
| Initial State: | None |
| Input: | Symbol, coordinate, context value, item class specifier, name value, psuedo name |
| Expected Output: | Item object in valid initial state |
| Name: | Name Vector Check |
| Initial State: | None |
| Input: | Vector of item names |
| Expected Output: | Shuffled vector of item names |
| Name: | Item Curse Check |
| Initial State: | Uncursed item |
| Input: | None |
| Expected Output: | Verification that item is uncursed |
| Name: | Item Curse/Effect Check 1 |
| Initial State: | Uncursed item to which the cursed effect has been applied |
| Input: | None |
| Expected Output: | Verification that item is cursed |
| Name: | Item Curse/Effect Check 2 |
| Initial State: | Cursed item whose curse effect has been removed |
| Input: | None |
| Expected Output: | Verification that item is uncursed |
| Name: | Item Unidentified Check |
| Initial State: | Identified item |
| Input: | None |

| | |
|-------------------------|--|
| Expected Output: | Verification that item is unidentified |
| Name: | Item Identified Check |
| Initial State: | Unidentified item |
| Input: | None |
| Expected Output: | Verification that item is identified |
| Name: | Item Display-Name Check 1 |
| Initial State: | Unidentified item |
| Input: | Psuedoname |
| Expected Output: | Verification that item's display name matches psuedoname |
| Name: | Item Display-Name Check 2 |
| Initial State: | Identified item |
| Input: | True name |
| Expected Output: | Verification that item's display name matches true name |
| Name: | ItemZone Containment Check 1 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that ItemZone contains the first item |
| Name: | ItemZone Containment Check 2 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that ItemZone contains the second item |
| Name: | ItemZone Empty Check |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that ItemZone is not empty |
| Name: | ItemZone Size Check |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that ItemZone's size is 2 |
| Name: | ItemZone Keybind Check 1 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that first item is bound to 'a' key |
| Name: | ItemZone Keybind Check 2 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Verification that second item is bound to 'b' key |

| | |
|-------------------------|---|
| Name: | ItemZone Contents Retrieval 1 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Item map with exactly 1 copy of first item |
| Name: | ItemZone Contents Retrieval 2 |
| Initial State: | ItemZone with 2 items |
| Input: | None |
| Expected Output: | Item map with exactly 1 copy of second item |
| Name: | ItemZone Removal |
| Initial State: | ItemZone with 2 items |
| Input: | Removal command |
| Expected Output: | ItemZone with only second item |
| Name: | ItemZone Keybind Persistence |
| Initial State: | ItemZone with first item removed |
| Input: | None |
| Expected Output: | Verification that second item is still bound to 'b' |
| Name: | ItemZone Weight Enforcement |
| Initial State: | Empty ItemZone |
| Input: | Attempt to add 500 pieces of armor to ItemZone |
| Expected Output: | ItemZone with max-weight worth of armor |
| Name: | Level Construction |
| Initial State: | None |
| Input: | Depth, player object |
| Expected Output: | Level object in valid initial state |
| Name: | Level Depth Check |
| Initial State: | Level with given depth |
| Input: | Depth value |
| Expected Output: | Verification that level's depth matches given value |
| Name: | Level BFSPerp Diagonal Small |
| Initial State: | Empty level object |
| Input: | Pair of coordinates diagonally adjacent |
| Expected Output: | Path between coordinates with expected length, utilizing taxicab movement |
| Name: | Level BFSPerp Horizontal |
| Initial State: | Empty level object |
| Input: | Pair of coordinates with equal y-values |
| Expected Output: | Path between coordinates with expected length, utilizing taxicab movement |
| Name: | Level BFSPerp Vertical |

| | |
|-------------------------|--|
| Initial State: | Empty level object |
| Input: | Pair of coordinates with equal x-values |
| Expected Output: | Path between coordinates with expected length, utilizing taxicab movement |
| Name: | Level BFSDiag Horizontal |
| Initial State: | Empty level object |
| Input: | Pair of coordinates with equal y-values |
| Expected Output: | Path between coordinates with expected length, utilizing orthogonal movement |
| Name: | Level BFSDiag Vertical |
| Initial State: | Empty level object |
| Input: | Pair of coordinates with equal x-values |
| Expected Output: | Path between coordinates with expected length, utilizing orthogonal movement |
| Name: | Level BFSPerp Diagonal |
| Initial State: | Empty level object |
| Input: | Pair of coordinates on diagonal line |
| Expected Output: | Path between coordinates with expected length, utilizing taxicab movement |
| Name: | Level Starting Position |
| Initial State: | Empty level object |
| Input: | None |
| Expected Output: | Valid starting position coordinate |
| Name: | Level getAdjPassable |
| Initial State: | Empty level object |
| Input: | Coordinate |
| Expected Output: | List of coordinates orthogonally adjacent to given coordinate |
| Name: | Level Path Generation |
| Initial State: | Player object and generated level |
| Input: | Series of path requests between random coordinates |
| Expected Output: | Valid paths between locations |
| Name: | Level Connectedness |
| Initial State: | Player object and generated level |
| Input: | Series of path requests between all rooms in the level |
| Expected Output: | Valid paths between each room |
| Name: | Level Staircase Check |
| Initial State: | Player object and generated level |
| Input: | None |
| Expected Output: | Verification that level contains a staircase |
| Name: | Level GoldPile Check |
| Initial State: | Player object and generated level |

| | |
|-------------------------|--|
| Input: | None |
| Expected Output: | Verification that level contains at least one goldpile |
| Name: | Monster Construction |
| Initial State: | None |
| Input: | Symbol, coordinate, armor value, HP value, exp value, level value, maxHP |
| Expected Output: | Monster object in valid initial state |
| Name: | Dice-Math 1 |
| Initial State: | None |
| Input: | 1 1-sided die |
| Expected Output: | Sum of values of 1 |
| Name: | Dice-Math 2 |
| Initial State: | None |
| Input: | 2 1-sided die |
| Expected Output: | Sum of values of 2 |
| Name: | Dice-Math 3 |
| Initial State: | None |
| Input: | 1 2-sided die |
| Expected Output: | 1 j= Sum of values j= 2 |
| Name: | Dice-Math 4 |
| Initial State: | None |
| Input: | 3 4-sided die |
| Expected Output: | 3 j= Sum of values j= 12 |
| Name: | Mob Armor Check |
| Initial State: | Mob object |
| Input: | None |
| Expected Output: | Verification mob armor is in valid range |
| Name: | Mob HP Check 1 |
| Initial State: | Mob with given HP value |
| Input: | HP value |
| Expected Output: | Verification mob has correct HP value |
| Name: | Mob MaxHP Check |
| Initial State: | Mob with given MaxHP value |
| Input: | MaxHP value |
| Expected Output: | Verification mob has correct MaxHP value |
| Name: | Mob Level Check |
| Initial State: | Mob with given level value |
| Input: | Level value |

| | |
|-------------------------|---|
| Expected Output: | Verification mob has correct level value |
| Name: | Mob Location Check |
| Initial State: | Mob with given location |
| Input: | Coordinate |
| Expected Output: | Verification mob has correct location |
| Name: | Mob Name Check |
| Initial State: | Mob with given name |
| Input: | Name value |
| Expected Output: | Verification mob has correct name |
| Name: | Mob setMaxHP |
| Initial State: | Mob with default MaxHP |
| Input: | setMaxHP command with MaxHP value |
| Expected Output: | mob with given MaxHP value |
| Name: | Mob setCurrentHP |
| Initial State: | Mob with default currentHP |
| Input: | setCurrentHP command with currentHP value |
| Expected Output: | mob with given currentHP value |
| Name: | Mob Dead Check 1 |
| Initial State: | Living Mob object |
| Input: | None |
| Expected Output: | Verification mob is alive |
| Name: | Mob HP Check 2 |
| Initial State: | Living Mob object |
| Input: | Hit command for i mob's current HP |
| Expected Output: | Verification mob has HP $j = 0$ |
| Name: | Mob Dead Check 2 |
| Initial State: | Dead mob object |
| Input: | None |
| Expected Output: | Verification mob is dead |
| Name: | Monster Construction |
| Initial State: | None |
| Input: | Symbol, coordinate |
| Expected Output: | Monster object in valid initial state |
| Name: | Monster Flag/Invisibility |
| Initial State: | Visible monster object |
| Input: | SetFlag command to make monster invisible |
| Expected Output: | Invisible monster object |

| | |
|-------------------------|---|
| Name: | Monster Aggrevate |
| Initial State: | Idling, sleeping monster object |
| Input: | Aggrevate command |
| Expected Output: | Awake, chasing monster object |
| Name: | Monster Damage Calculation |
| Initial State: | Monster object |
| Input: | calculateDamage command |
| Expected Output: | Correct amount of damage |
| Name: | Monster Hit Chance |
| Initial State: | Monster and player objects |
| Input: | calculateHitChange command |
| Expected Output: | Hit chance in valid range |
| Name: | Monster Armor Check |
| Initial State: | Monster object |
| Input: | None |
| Expected Output: | Verification that monster armor is in valid range |
| Name: | Invisible Monster Name Check |
| Initial State: | Invisible uonster object |
| Input: | None |
| Expected Output: | Verification monster has hidden name |
| Name: | Visible Monster Name Check |
| Initial State: | Invisible monster object |
| Input: | RemoveFlag command to make monster invisible |
| Expected Output: | Verification monster has real name |
| Name: | Monster Symbol/Level Association |
| Initial State: | None |
| Input: | Depth value |
| Expected Output: | Set of symbols for monsters that are valid candidates for given depth |
| Name: | Monster Symbol/Treasure/Level Association |
| Initial State: | None |
| Input: | Depth value |
| Expected Output: | Set of symbols for monsters that are valid candidates for given depth for |
| Name: | PlayerChar Initial Amulet Check |
| Initial State: | Just initialized playerchar object |
| Input: | None |
| Expected Output: | Verification the game does not believe the player has the amulet |
| Name: | PlayerChar Initial HP Check |

| | |
|-------------------------|--|
| Initial State: | Just initialized playerchar object |
| Input: | None |
| Expected Output: | Verification playerchar has full hp |
| Name: | PlayerChar Level-Up Exp |
| Initial State: | Playerchar object at initial level |
| Input: | Exp input into playerchar object |
| Expected Output: | Playerchar object with increased level |
| Name: | PlayerChar Level-Up Manual |
| Initial State: | Playerchar object |
| Input: | Level-up command |
| Expected Output: | Playerchar object with increased level |
| Name: | PlayerChar Damage |
| Initial State: | Playerchar object at full hp |
| Input: | Series of damage commands applied to playerchar object |
| Expected Output: | Playerchar object with less than full hp |
| Name: | PlayerChar UnArmed 1 |
| Initial State: | Unarmed playerchar object |
| Input: | calculateDamage command |
| Expected Output: | 0 damage value |
| Name: | PlayerChar Armed |
| Initial State: | Playerchar object armed with weapon |
| Input: | calculateDamage command |
| Expected Output: | Damage value i , 0 |
| Name: | PlayerChar Stow Weapon |
| Initial State: | Playerchar object armed with uncursed weapon |
| Input: | removeWeapon command |
| Expected Output: | PlayerChar object unarmed |
| Name: | PlayerChar UnArmed 2 |
| Initial State: | Armed playerchar object |
| Input: | removeWeapon command, then calculateDamage |
| Expected Output: | 0 damage value |
| Name: | PlayerChar Remove Non-Armor |
| Initial State: | Playerchar object with no armor |
| Input: | removeArmor command |
| Expected Output: | Boolean indicating failure to remove armor |
| Name: | PlayerChar Remove Armor |
| Initial State: | Playerchar object with uncursed armor |

| | |
|-------------------------|---|
| Input: | removeArmor command |
| Expected Output: | Playerchar object without armor |
| Name: | Potion Construction 1 |
| Initial State: | None |
| Input: | Coordinate |
| Expected Output: | Potion object in valid initial state |
| Name: | Potion Construction 2 |
| Initial State: | None |
| Input: | Coordinate, item context value, item type specifier |
| Expected Output: | Potion object in valid initial state |
| Name: | Potion of Strength |
| Initial State: | Player object |
| Input: | Potion of strength |
| Expected Output: | Player with strength increased by 1 |
| Name: | Potion of Restore Strength |
| Initial State: | Player object with reduced strength |
| Input: | Potion of restore strength |
| Expected Output: | Player object with pre-reduction strength |
| Name: | Potion of Healing |
| Initial State: | Player object with full hp |
| Input: | Potion of healing |
| Expected Output: | Player object with maxHP increased by 1 |
| Name: | Potion of Extra Healing |
| Initial State: | Player object with full hp |
| Input: | Potion of extra healing |
| Expected Output: | Player object with maxHP increased by 2 |
| Name: | Potion of Poison |
| Initial State: | Player object with strength ≥ 0 |
| Input: | Potion of poison |
| Expected Output: | Player object with reduced strength |
| Name: | Potion of Raise Level |
| Initial State: | Player object with less than max level |
| Input: | Potion or raise level |
| Expected Output: | Player object with level + 1 |
| Name: | Potion of Blindness |
| Initial State: | Player object without the blindness condition |
| Input: | Potion of blindness |

| | |
|-------------------------|---|
| Expected Output: | Player object with the blindness condition |
| Name: | Potion of Hallucination |
| Initial State: | Player object without the hallucination condition |
| Input: | Potion of hallucination |
| Expected Output: | Player object with the hallucination condition |
| Name: | Potion of Detect Monster |
| Initial State: | Player object without the detect-monsters condition |
| Input: | Potion of detect monsters |
| Expected Output: | Player object with the detect-monsters condition |
| Name: | Potion of Detect Object |
| Initial State: | Player object without the detect-objects condition |
| Input: | Potion of detect objects |
| Expected Output: | Player object with the detect-objects condition |
| Name: | Potion of Confusion |
| Initial State: | Player object without the confusion condition |
| Input: | Potion of confusion |
| Expected Output: | Player object with the confusion condition |
| Name: | Potion of Confusion |
| Initial State: | Player object without the confusion condition |
| Input: | Potion of confusion |
| Expected Output: | Player object with the confusion condition |
| Name: | Potion of Levitation |
| Initial State: | Player object without the levitation condition |
| Input: | Potion of levitation |
| Expected Output: | Player object with the levitation condition |
| Name: | Potion of Haste |
| Initial State: | Player object without the haste condition |
| Input: | Potion of haste |
| Expected Output: | Player object with the haste condition |
| Name: | Potion of See-Invisible |
| Initial State: | Player object without the invisible-sight condition |
| Input: | Potion of invisible |
| Expected Output: | Player object with the invisible-sight condition |
| Name: | Random Range 1 |
| Initial State: | None |
| Input: | Upper and lower bounds 0,0 |
| Expected Output: | 0 |

| | |
|-------------------------|--|
| Name: | Random Range 2 |
| Initial State: | None |
| Input: | Upper and lower bounds 5,5 |
| Expected Output: | 5 |
| Name: | Random Range 3 |
| Initial State: | None |
| Input: | Upper and lower bounds 0,60, repeated 40 times |
| Expected Output: | 0 j= result j= 60 |
| Name: | Random Float |
| Initial State: | None |
| Input: | 40 repeats |
| Expected Output: | 0 j= result j= 1 |
| Name: | Random Boolean |
| Initial State: | None |
| Input: | 10 repeats |
| Expected Output: | Both true and false are generated |
| Name: | Random Percent |
| Initial State: | None |
| Input: | 40 repeats |
| Expected Output: | 0 j= result j= 100 |
| Name: | Random Position |
| Initial State: | None |
| Input: | Two coordinates, as top-left and bottom-right of rectangle, 10 repeats |
| Expected Output: | Random coordinates within the bounds |
| Name: | Ring Construction 1 |
| Initial State: | None |
| Input: | Coordinate |
| Expected Output: | Ring object with valid initial state |
| Name: | Ring Construction 2 |
| Initial State: | None |
| Input: | Coordinate, item context value, type identifier |
| Expected Output: | Ring object with valid initial state |
| Name: | Ring of Stealth |
| Initial State: | Player object without stealth condition |
| Input: | Activate ring of stealth |
| Expected Output: | Player object with the stealth condition |
| Name: | Ring of Teleportation |

| | |
|-------------------------|---|
| Initial State: | Player object without random teleportation condition |
| Input: | Activate ring of teleportation |
| Expected Output: | Player object with the random teleportation condition |
| Name: | Ring of Regeneration |
| Initial State: | Player object without regeneration condition |
| Input: | Activate ring of regeneration |
| Expected Output: | Player object with the regeneration condition |
| Name: | Ring of Digestion |
| Initial State: | Player object without digestion condition |
| Input: | Activate ring of digestion |
| Expected Output: | Player object with the digestion condition |
| Name: | |
| Initial State: | |
| Input: | |
| Expected Output: | |

7 Trace to Requirements

Ori

8 Trace to Modules

The following table re-iterates the modules of the project, along with their respective domain and module ID. The module IDs are used to refer to modules in the trace. More about the modules can be found in the Module Guide.

Table 3: **Module Hierarchy**

| Level 1 | Level 2 | |
|-----------------------------|------------------|------------|
| Hardware-Hiding Module | BasicIO | M1 |
| | Doryen | M2 |
| | Input Format | M3 |
| Behaviour-Hiding Module | External | M4 |
| | Item | M5 |
| | Level | M6 |
| | LevelGen | M7 |
| | MainMenu | M8 |
| | Monster | M9 |
| | PlayerChar | M10 |
| | RipScreen | M11 |
| | PlayState | M12 |
| | UIState | M13 |
| Software Decision Module | Coord | M14 |
| | Feature | M15 |
| | ItemZone | M16 |
| | MasterController | M17 |
| | Mob | M18 |
| | Random | M19 |
| | Terrain | M20 |

The following table maps test files, which implement tests, to specific modules, given by their IDs.

Table 4: **Test-Module Trace**

| File | Related Module(s) |
|---------------------|---|
| test.amulet.cpp | M7, M12, M13 |
| test.armor.cpp | M5, M10, M18 |
| test.coord.cpp | M2, M5, M6, M7, M14, M19 |
| test.feature.cpp | M5, M15, M16, M10 |
| test.food.cpp | M5, M6, M7, M10, M12 |
| test.goldpile.cpp | M5, M6, M7, M9, M10, M15, M16 |
| test.item.cpp | M5, M15 |
| test.itemzone.cpp | M5, M6, M14, M15, M16 |
| test.level.cpp | M5, M6, M9, M10, M14, M15, M19 |
| test.levelgen.cpp | M5, M6, M9, M14, M15, M19, M20 |
| test.main.cpp | None (Puts everything together) |
| test.mob.cpp | M9, M10, M12, M13, M14, M18 |
| test.monster.cpp | M9, M14, M18 |
| test.playerchar.cpp | M5, M6, M10, M11, M12, M13, M14, M15, M16, M17, M18 |
| test.potion.cpp | M5, M6, M7, M9, M10, M15, M16 |
| test.ring.cpp | M5, M6, M7, M9, M10, M15, M16 |
| test.room.cpp | M6, M7, M14, M19 |
| test.scroll.cpp | M5, M6, M7, M9, M10, M15, M16 |
| test.stairs.cpp | M7, M15, M17, M20 |
| test.terrain.cpp | M6, M7, M19, M20 |
| test.testable.cpp | Defines test-suite |
| test.testable.h | Defines test-suite |
| test.trap.cpp | M6, M7, M10, M13, M15 |
| test.tunnel.cpp | M6, M5, M14 |
| test.uistate.cpp | M4, M8, M11, M12, M13, M17 |
| test.wand.cpp | M5, M6, M7, M9, M10, M15, M16 |
| test.weapon.cpp | M5, M6, M7, M9, M10, M15, M16 |

9 Code Coverage Metrics

Ori