**Model test 1**: Checks that the character jumps, decelerates, falls, and hits ground (which sets vertical velocity to 0) accordingly based on pre-set gravitational acceleration. This test also checks the case of a parabolic jump (where character horizontal velocity isn’t 0).

**Lines**: L47

**Model test 2**: Checks that the character moves properly and that collisions (hitting walls horizontally and ceilings vertically) worked properly. Horizontal velocity is set to 0 when running against a wall which the character is already touching, and vertical velocity is set to 10 once the character touches the ceiling in-air.

**Lines**: L141

**Model test 3**: Checks that the game is won if both characters are at their respective doors, and doesn’t win if either character isn’t at their respective door.

**Lines**: L229

**Model test 4**: Checks that the game is lost if fireboy touches the blue pond, watergirl touches the red pond, or either character touches the green pond.

**Lines**: L264

**Model test 5**: Checks that both characters collect gems of their respective color accordingly and adds to the total score. If a gem is collected, it disappears from the map.

**Lines**: L338

**Functional Requirement 1:** A map is displayed upon beginning of the game, consisting of two playable characters (fireboy and watergirl) standing at the bottom-left of the screen, some obstacles and map features, and two gates at the top of the screen for them to escape.

**Explanation:** Simply load the game and you should see all the features outlined on the screen. Note 1: to initialize the game properly, I implemented 3 seconds of “flying frame”, where both characters will float in air; after this 3 seconds, gravitational acceleration will be applied and they should fall to the ground. Note 2: to close the game window quickly, press command+Q.

**Functional Requirement 2:** The player can control fireboy’s movement via the ←↑→↓keys.

**Explanation:** Once you enter the game, try moving around fireboy using ←↑→↓keys. (Backup: validation by test in test/model\_test/L47 & L141; validation by implementation in src/model/L93-105)

**Functional Requirement 3:** The player can control watergirl’s movement via the awsd keys

**Explanation:** Once you enter the game, try moving around watergirl using awsd keys. (Backup: validation by test in test/model\_test/L47 & L141; validation by implementation in src/model/L79-91)

**Functional Requirement 4:** A simple physical engine will be implemented such that there’s a constant gravitational acceleration applied to the playable characters when they jump or fall.

**Explanation:** Move fireboy or watergirl right to a relatively empty area, then hit ↑and ‘w’ to observe that they jump and fall accordingly. You can also place either character to a cliff and observe that they fall once you run off the cliff. (Backup: validation by test in test/model\_test/L47)

**Functional Requirement 5:** Characters may run on solid blocks, and they are stopped when they hit a solid wall/ceiling.

**Explanation:** Try moving around both characters in the map, and you will see that this is implemented. (Backup: validation by test in test/model\_test/L141)

**Functional Requirement 6:** Red, blue, and green pond are located sparsely on the map. If either character touches the pond of a different color then themselves (i.e. if fireboy touches blue or if watergirl touches red pond) or the green pond, they die and the player looses.

**Explanation:** Try playing the game and let fireboy and watergirl touch red/blue/geen ponds and the game should either continue or end according to this functional requirement. (Backup: validation by test in test/model\_test/L264)

**Functional Requirement 7:** Red and blue gems are located sparsely on the map, where each character may collect gems of their own color. The number of gems collected counts towards the final game score.

**Explanation:** Pretty much just play the game: try run fireboy and watergirl around in the map to where blue and red gems are collected. You will observe that fireboy can’t collect blue gems and watergirl can’t collect red gems. (Backup: validation by test in test/model\_test/L338)

**Functional Requirement 8:** If both player arrives at their respective door (a blue door and a red door), both doors open for the characters to pass and the player wins.

**Explanation:** You may either try win the game yourself (which might take some practice), or head to src/game\_config.cxx, comment the tree lines tagged “Bottom left of the screen” and uncomment the three lines tagged “Top of the screen”. This should set both fireboy and watergirl’s starting position very close to their doors, and you can easily walk them to their doors and win the game. (Backup: validation by test in test/model\_test/L229)

**Functional Requirement 9:** Trap doors are located on the map, which are initialized as either active or inactive. If a trap door is initialized as active, it behaves like a normal wall/block; if a trap door is initialized as inactive, it behaves like the background and allows the character to pass. 3 Pedals are located on the map. If any of these pedals are being stepped on by either character, it switches the trap doors’ states (i.e. if a trap door is initialized as active, it turns inactive; if a trap door is initialized as inactive, it turns inactive)

**Explanation:** Head to src/game\_config.cxx, comment the tree lines tagged “Bottom left of the screen” and uncomment the three lines tagged “Middle of the screen”. You will notice a trapdoor (initialized as inactive) located right to your left, and another trapdoor located on the right of the screen (initialized as active). Try stepping on the pedals and see that the trap doors change their states and try using this feature to win the game.

**Functional Requirement 10:** For each character, if it is standing still, the game window will render a standing-still, animated character figure; if it is moving left/right, the game window will render a left-running/right-running, animated character figure.

**Explanation:** Pretty much same as functional requirements 2 & 3