**OPERATION CONTRACTS**

**Operations:**

openGame()

**Cross-reference:**

Initialize the Main Menu

**Pre-conditions:**

* The system gets the command to start the game.

**Post-conditions:**

* The player launches the game.
* A title screen shows up.
* Main menu shows up.

**Operations:**

startGame()

**Cross-reference:**

Initialize START Menu

**Pre-conditions:**

* The game must be started
* The main menu must be shown

**Post-conditions:**

* The player launches the game.
* Main menu shows up.
* User clicks the START button.
* A new game starts in build mode

**Operations:**

helpMenu()

**Cross-reference:**

Initialize HELP Menu

**Pre-conditions:**

* The game must be started
* The main menu must be shown

**Post-conditions:**

* The player launches the game.
* Main menu shows up.
* User clicks the HELP button.
* A HELP screen shows up

**Operations:**

exit()

**Cross-reference:**

Initialize EXIT Case

**Pre-conditions:**

* The game must be started
* The main menu must be shown

**Post-conditions:**

* The player launches the game.
* Main menu shows up.
* User clicks the EXIT button.
* System shu ts the game down

**Operations:**

pickupHallObject(Object object)

**Cross-reference:**

Building the Dungeon

**Pre-conditions:**

* The game must be in build mode

**Post-conditions:**

* The panel with objects is displayed
* User picks up an object

**Operations:**

dragHallObject(Object object)

**Cross-reference:**

Building the Dungeon

**Pre-conditions:**

* The game must be in build mode

**Post-conditions:**

* The player picks up an object from the panel
* Moves the object with a mouse

**Operations:**

releaseHallObject(Object object)

**Cross-reference:**

Building the Dungeon

**Pre-conditions:**

* The game must be in build mode

**Post-conditions:**

* The player picks up and drags an object
* The player releases the object

**Operations:**

movePlayer(Direction direction)

**Cross-reference:**

Player Movement

**Pre-conditions:**

* The game must be in play mode

**Post-conditions:**

* The player is at a random square in the current Hall.
* One of the arrows is clicked to move the player.
* The player moves toward the desired direction in one square magnitude.

**Operations:**

collectRune(Rune rune)

**Cross-reference:**

Collecting Runes, Winning the Game

**Pre-conditions:**

* Player must be in playing mode.
* The rune that player wants to collect must be hidden in one of the objects.

**Post-conditions:**

* The system hides the rune in one of the objects.
* The player goes near an object.
* The player left-clicks to search for the rune.
* The player repeats steps 2-3 until success.
* The current hall is marked as complete
* The door is opened for passage to the next hall.

**Operations:**

collectRune(Rune rune)

**Cross-reference:**

Collecting Runes, Winning the Game

**Pre-conditions:**

* Player must be in playing mode.
* The rune that player wants to collect must be hidden in one of the objects.

**Post-conditions:**

* The system hides the rune in one of the objects.
* The player goes near an object.
* The player left-clicks to search for the rune.
* The player repeats steps 2-3 until success.
* The current hall is marked as complete
* All the runes are collected, and the player wins

**Operations:**

initializeHero()

**Cross-reference:**

Initialize Play Mode

**Pre-conditions:**

* + The game is in Build Mode, and all required objects have been placed in each hall.
  + All halls meet the object placement criteria.

**Post-conditions:**

* + The game transitions to Play Mode.
  + The hero's position is initialized in the first hall (Hall of Earth).

**Operations:**

initializeObjects()

**Cross-reference:**

Initialize Play Mode

**Pre-conditions:**

* + The game is in Build Mode, and all required objects have been placed in each hall.
  + All halls meet the object placement criteria.

**Post-conditions:**

* + The game transitions to Play Mode.
  + All game elements, including monsters and enchantments, are initialized.

**Operations:**

initializeTimer()

**Cross-reference:**

Initialize Play Mode

**Pre-conditions:**

* + The game is in Build Mode, and all required objects have been placed in each hall.
  + All halls meet the object placement criteria.

**Post-conditions:**

* The game transitions to Play Mode.
* The timer for the first hall starts.

**Operations:**

addEnchantmentList(Enchantment enchantment)

**Cross-reference:**

Collecting Enchantments

**Pre-conditions:**

* Player must be in playing mode.
* The enhancement that player wants to collect must be visible in the screen.

**Post-conditions:**

* The system puts a random enhancement randomly in the grid every 12 seconds.
* The player left clicks the enhancement.
* The system takes the enhancement.
* The system updates the inventory.

**Operations:**

removeEnchantmentFromMap(Enchantment enchantment)

**Cross-reference:**

Collecting Enchantments

**Pre-conditions:**

* Player must be in playing mode.
* The enhancement that player wants to collect must be visible in the screen.

**Post-conditions:**

* The system puts a random enhancement randomly in the grid every 12 seconds.
* The player left clicks the enhancement.
* The system takes the enhancement.
* The system updates the inventory.
* The system removes the enchantment from the grid

**Operations:**

useEnchantment(Enchantment enchantment)

**Cross-reference:**

Enchantment Utilization

**Pre-conditions:**

* The game is in the Play Mode.
* The player has the reveal enchantment in the inventory

**Post-conditions:**

* The player presses the button R on the keyboard to use the enchantment.
* A 4x4 square containing the rune shows up and disappears after 10 seconds.

**Operations:**

useEnchantment(Enchantment enchantment)

**Cross-reference:**

Enchantment Utilization

**Pre-conditions:**

* The game is in the Play Mode.
* The player has the reveal enchantment in the inventory

**Post-conditions:**

* The player presses the button R on the keyboard to use the enchantment.
* The player is hidden from the archer monster for 20 seconds

**Operations:**

useEnchantment(Enchantment enchantment)

**Cross-reference:**

Enchantment Utilization

**Pre-conditions:**

* The game is in the Play Mode.
* The player has the reveal enchantment in the inventory

**Post-conditions:**

* The player presses the button B and one of the A, D, W, S buttons to determine the direction.
* The gem is successfully thrown at the aimed position and the fighter monster follows it.

**Operations:**

decreaseEnchantmentList(Enchantment enchantment)

**Cross-reference:**

Enchantment Utilization

**Pre-conditions:**

* The game is in the Play Mode.
* The player has the reveal enchantment in the inventory

**Post-conditions:**

* The player uses enchantment.
* The inventory is updated.

**Operations:**

pauseTimer()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The player clicks the pause button during gameplay.
* The system halts the timer
* The system displays a paused state or menu, indicating that the game is paused.

**Operations:**

pausePlayer()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The player clicks the pause button during gameplay.
* The system halts in-game the player
* The system displays a paused state or menu, indicating that the game is paused.

**Operations:**

pauseMonsters()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The player clicks the pause button during gameplay.
* The system halts all the monsters
* The system displays a paused state or menu, indicating that the game is paused.

**Operations:**

resumeTime()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The game is paused
* The player clicks the resume button.
* The system resumes the timer from the exact state before the pause.

**Operations:**

resumePlayer()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The game is paused
* The player clicks the resume button.
* The system resumes the player from the exact state before the pause.

**Operations:**

resumeMonsters()

**Cross-reference:**

Pause/Resume

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The game is paused
* The player clicks the resume button.
* The system resumes the monsters from the exact state before the pause.

**Operations:**

spawnMonster()

**Cross-reference:**

Player Movement

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The game is in Play Mode in a hall
* A random monster appears at a random location without intersecting any other object

**Operations:**

timerEquals(Time time)

**Cross-reference:**

Player Movement

**Pre-conditions:**

* The game is in Play Mode.

**Post-conditions:**

* The game is in Play Mode in a hall
* A random monster appears at a random location without intersecting any other object
* Eight seconds timer until the next spawn resets

**Operations:**

timerEquals(Time time)

**Cross-reference:**

Losing the game due to time

**Pre-conditions:**

* The player is in playing mode.
* The game is not paused.
* The game has not finished by another means
* The timer reaches zero.

**Post-conditions:**

* The player enters a hall.
* The timer initializes to a value of 5 seconds per object.
* The timer counts down by one every second.
* The timer reaches 0.
* The game finishes, the “Game Over” screen is displayed.

**Operations:**

initiateDamage(Monster monster, Player player)

**Cross-reference:**

Losing life because of a damage by a monster

**Pre-conditions:**

* The game must be in play mode.
* The player must have at least one life.
* There is at least one fighter or an archer monster in the hall.

**Post-conditions:**

* A fighter or an archer monster gets spawned in an empty location.
* Player has at least one out of three lives.
* Character moves to a location that is in the proximity of the monster.
* Monster attacks and the player loses a life.

**Operations:**

reduceHealth(Player player, Damage hitDamage)

**Cross-reference:**

Losing life because of a damage by a monster

**Pre-conditions:**

* The game must be in play mode.
* The player must have at least one life.
* There is at least one fighter or an archer monster in the hall.

**Post-conditions:**

* A fighter or an archer monster gets spawned in an empty location.
* Player has at least one out of three lives.
* Character moves to a location that is in the proximity of the monster.
* Monster attacks and the player loses a life.

**Operations:**

execute(WizardMonster wizard, PlayModeController controller)

**Cross-reference:**

Wizard Monster Behavior-phase 2

**Pre-conditions:**

* The game must be in play mode.
* The player must have at least one life.
* There is at least one wizard monster in the hall.

**Post-conditions:**

* The game continues after the wizard monster exhibits its behavior.

**Operations:**

Save()

**Cross-reference:**

Save Game

**Pre-conditions:**

* The game must be in play mode.
* The game must be paused.

**Post-conditions:**

* A file created storing the necessary information to continue the game later.

**Operations:**

Load(String path)

**Cross-reference:**

Load Game

**Pre-conditions:**

* The player must be in MainMenu.
* Save file should exist in the system.

**Post-conditions:**

* The game is started from the saved progression.