Milestone 2 - Preliminary Design Testing Plan

Controller Class

Testing startGame()

Input	Expected Value
Game starts with 1 human, 1 computer, and valid world file	Game starts successfully, alternates turns between players, moves target, and executes actions for both human and computer players.
Game starts with no players	Exception or error message indicating that no players are available to start the game.
Game starts with 1 human and invalid room for the human player	Error indicating the player's starting room is invalid, game does not proceed.
Game starts with invalid file for the world	Exception or error indicating invalid world data, game does not start.
Game starts and reaches maximum turn limit	Game proceeds until the specified maximum number of turns is reached, then ends gracefully with a message.
Game starts but user manually quits early	Game stops when the user enters a "quit" command, ensuring any resources are cleaned up, and the game ends properly.
Game with multiple human players	Alternates turns between human players correctly, each player gets a chance to take an action in turn.
Game with only computer players	Game runs automatically, computer players take turns correctly and perform valid actions.
Game starts but target character's health reaches zero	Game stops when the target character dies or reaches zero health, displaying a message.
Player carries more than allowed items	Game allows players to pick up items only up to their carry limit, and denies further pickups when the limit is reached.

Testing addHumanPlayer(String playerName, IRoom startingRoom)

Input	Expected Value
Valid player name, valid room	Human player is added to the world successfully.
Valid player name, null room	Exception thrown for null room input.
Null player name, valid room	Exception thrown for null player name input.
Duplicate player name, valid room	Player with duplicate name is handled properly (either allowed or rejected, based on design).

Testing addComputerPlayer(String playerName, IRoom startingRoom)

Input	Expected Value
Valid player name, valid room	Computer player is added to the world successfully.
Valid player name, null room	Exception thrown for null room input.
Null player name, valid room	Exception thrown for null player name input.
Duplicate player name, valid room	Player with duplicate name is handled properly (either allowed or rejected, based on design).

Testing movePlayer(String playerName, IRoom room)

Input	Expected Value
Valid player, valid room	Player is moved to the specified room successfully.
Valid player, null room	Exception thrown for null room input.
Null player name, valid room	Exception thrown for null player name input.
Invalid room (non-neighboring)	Exception or handling for invalid room movement.
Invalid player name	Exception or handling for non-existent player.

Testing playerPickUpItem(String playerName, IItem item)

Input	Expected Value
Valid player, valid item	Item is picked up by the player successfully.

Valid player, null item	Exception thrown for null item input.
Null player name, valid item	Exception thrown for null player name input.
Valid player, item exceeding carry limit	Error message or exception for exceeding item carry limit

Testing lookAround(String playerName)

Input	Expected Value
Valid player	Information about neighboring rooms and items is displayed.
Player in a room with no neighbors	Appropriate message displayed indicating no neighboring rooms.
Null player name	Exception thrown for null player name input.
Invalid player name	Error message for non-existent player.

Testing displayPlayerDescription(String playerName)

Input	Expected Value
Valid player with items	Player description including their current location and items carried.
Valid player without items	Player description with no items, showing their location.
Null player name	Exception thrown for null player name input.
Invalid player name	Error message for non-existent player.

Testing saveWorldMap(String filePath)

Input	Expected Value
Valid file path	World map is successfully saved as a PNG image at the specified location.
Null file path	Exception thrown for null file path input.
Invalid file path (non-existent directory)	Exception or error message indicating that the directory does not exist, and the file cannot be saved.
File path to a restricted location (no write permissions)	Exception or error message indicating that the file cannot be saved due to permission issues.

Testing limitTurns(int maxTurns)

Input	Expected Value
Valid max turns	The game is limited to the specified number of turns.
Zero or negative turns	Exception or appropriate error handling for invalid turn input.
Exceeding the max turns	The game ends or provides an appropriate message when the max turns are reached.

World Class

Testing addPlayer(IPlayer player)

Input	Expected Value
Valid human player	Player is successfully added to the world.
Valid computer player	Player is successfully added to the world.
Null player input	Exception thrown for null input.
Duplicate player	Either player added again or prevented, depending on the design.

Testing removePlayer(IPlayer player)

Input	Expected Value
Valid human player	Player is successfully removed from the world.
Valid computer player	Player is successfully removed from the world.
Non-existent player	Exception or appropriate handling of invalid removal.
Null player input	Exception thrown for null input.

Testing getPlayers()

Input	Expected Value
World with multiple players	List of all players is returned.
World with no players	Empty list returned.

Testing movePlayer(String playerName, IRoom room)

Input	Expected Value
Valid player, valid room	Player is moved to the specified room successfully.
Invalid player	Exception or appropriate handling for invalid player.
Invalid room (non-neighboring)	Exception or handling for invalid room movement.
Null room or player input	Exception thrown for null input.

Room Class

Testing addPlayer(IPlayer player)

Input	Expected Value
Valid human player	Player is added to the room successfully.
Valid computer player	Player is added to the room successfully.
Null player input	Exception thrown for null input.
Duplicate player	Either player added again or prevented, depending on the design.

Testing removePlayer(IPlayer player)

Input	Expected Value
Valid human player	Player is removed from the room successfully.
Valid computer player	Player is removed from the room successfully.
Non-existent player	Exception or appropriate handling of invalid removal.
Null player input	Exception thrown for null input.

Testing getPlayers()

Input	Expected Value
Room with multiple players	List of players is returned.
Room with no players	Empty list returned

Player Class (HumanPlayer and ComputerPlayer)

Testing moveTo(IRoom room)

Input	Expected Value
Valid room	Player is moved to the specified room successfully.
Null room input	Exception thrown for null room input.

Testing pickUpItem(IItem item)

Input	Expected Value
Valid item within carry limit	Item is picked up by the player successfully.
Valid item exceeding carry limit	Item not picked up, or error message displayed.
Null item input	Exception thrown for null item input.

Testing lookAround()

Input	Expected Value
Player in a room with neighbors	Information about the neighboring rooms and items displayed.
Player in a room with no neighbors	Appropriate message displayed indicating no neighboring rooms.

Testing getDescription()

Input	Expected Value
Player with items	Detailed description of the player, their items, and location.
Player without items	Description of the player with no items.