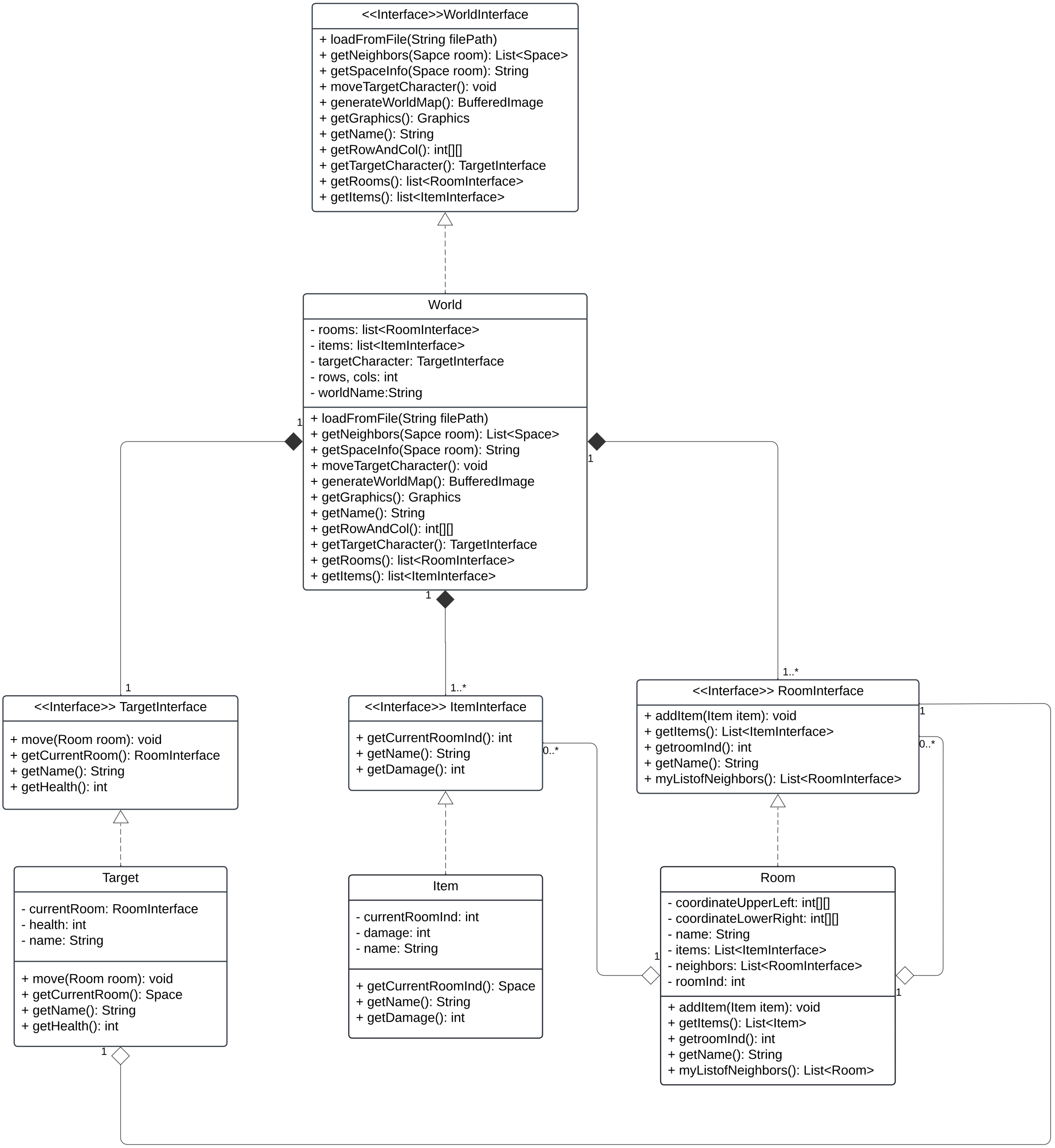
<<Interface>>WorldInterface
+ loadFromFile(String filePath)
+ getNeighbors(Sapce room): List<Space>
+ getSpaceInfo(Space room): String
+ moveTargetCharacter(): void
+ generateWorldMap(): BufferedImage
+ getGraphics(): Graphics
+ getName(): String
+ getRowAndCol(): int[][]
+ getTargetCharacter(): TargetInterface
+ getRooms(): list<RoomInterface>
+ getItems(): list<ItemInterface>
World
- rooms: list<RoomInterface>
- items: list<ItemInterface>
- targetCharacter: TargetInterface
- rows, cols: int
- worldName:String
+ loadFromFile(String filePath)
+ getNeighbors(Sapce room): List<Space>
+ getSpaceInfo(Space room): String
+ moveTargetCharacter(): void
+ generateWorldMap(): BufferedImage
+ getGraphics(): Graphics
+ getName(): String
+ getRowAndCol(): int[][]
+ getTargetCharacter(): TargetInterface
+ getRooms(): list<RoomInterface>
+ getItems(): list<ItemInterface>
1
1
1
1
1
1..*
1..*
<<Interface>> TargetInterface
+ move(Room room): void
+ getCurrentRoom(): RoomInterface
+ getName(): String
+ getHealth(): int
<<Interface>> ItemInterface
+ getCurrentRoomInd(): int
+ getName(): String
+ getDamage(): int
0..*
<<Interface>> RoomInterface
+ addItem(Item item): void
+ getItems(): List<ItemInterface>
+ getroomInd(): int
+ getName(): String
+ myListofNeighbors(): List<RoomInterface>
1
0..*
Target
- currentRoom: RoomInterface
- health: int
- name: String
+ move(Room room): void
+ getCurrentRoom(): Space
+ getName(): String
+ getHealth(): int
Item
- currentRoomInd: int
- damage: int
- name: String
+ getCurrentRoomInd(): Space
+ getName(): String
+ getDamage(): int
Room
- coordinateUpperLeft: int[][]
- coordinateLowerRight: int[][]
- name: String
- items: List<ItemInterface>
- neighbors: List<RoomInterface>
- roomInd: int
+ addItem(Item item): void
+ getItems(): List<Item>
+ getroomInd(): int
+ getName(): String
+ myListofNeighbors(): List<Room>
1
1
1
1

Milestone 1 - Preliminary Design Testing Plan

World Class

Testing construction and loadFromFile()	Input	Expected Value
Valid file input	Path to a valid txt file	The world is correctly loaded, with all rooms, items, and target character data populated
Invalid file path	Path to a non-existent file	Exception thrown with an error indicating the file cannot be found
Malformed file	A file with incorrect formatting or missing data	Exception thrown with an error indicating invalid file format

Testing getNeighbors(Space room)	Input	Expected Value
Room with neighbors	A room surrounded by other rooms (Kitchen)	List of neighboring rooms correctly returned

Testing getSpaceInfo(Space room)	Input	Expected Value
Valid room input	Armory	Correct information about the Armory, including its name and items
Room without items	Piazza	Room name of Piazza and confirmation that no items are present

Testing moveTargetCharacter()	Input	Expected Value
Valid sequential movement	Call moveTargetCharacter() while the target is in room 0	The target moves to room 1

Move from last room	Call moveTargetCharacter() while the target is in the last room	Target stays in the last room

Testing generateWorldMap()	Input	Expected Value
Correct map generation	Call the method after loading a world	BufferedImage object correctly representing the layout of the world
No world loaded	Call the method without loading a world	Exception or empty BufferedImage generated

Testing getGraphics()	Input	Expected Value
Verify that the correct graphics object is returned correctly	Call method after generateWorldMap()	Valid Graphics object returned

Room Class

Testing Constructor Room(int[][] coordinateUpperLeft, int[][] coordinateLowerRight, String name)	Input	Expected Value
Valid input	Room([0][3], [5][8], "Lancaster Room")	Room object is successfully created with the correct coordinates and room name
Invalid coordinates (upper-left corner larger than lower-right corner)	Room([5][8], [0][3], "Invalid Room")	Exception thrown with an error for invalid coordinates (upper-left should always be smaller than lower-right)
Null or empty room name	Room([0][3], [5][8], null)	Exception with error handling for null or empty room name

Testing addItem(Item item)	Input	Expected Value
Add valid item	Add an item “Revolver” to the Armory	Item is added successfully, reflected in room’s item list
Add duplicate item	Add the same item multiple times	Either item added multiple times or prevented, depending on design choice

Testing getItems()	Input	Expected Value
Room with items	Call method for the Armory	List of items ("Revolver") correctly returned
Room without items	Call method for a room with no items (Winter Garden)	Empty list returned

Testing getRoomId()	Input	Expected Value
Room with valid ID	Room ID request for a specific room (Room 0)	Correct room ID 0 returned

Testing myListOfNeighbors()	Input	Expected Value
Room with multiple neighbors	Call method for Dining Hall	List of all neighboring rooms (including Tenessee Room, Trophy Room, Billiard Room, rmory, Drawing Room, Wine Cellar, Kitchen, and Parlor) returned
Room with 1 neighbor	Call method for Green House	List of room (Hedge Maze) returned

Testing addNeighbor(Room room)	Input	Expected Value
Add a valid neighbor	Add the Parlor to the Kitchen	Parlor is added successfully

Add a non-adjacent room as neighbor	Add the Armory to the Kitchen	Operation fails or is prevented

Item Class

Testing Constructor Item(int roomId, int damage, String name)	Input	Expected Value
Valid input	Item(0, 10, "Revolver")	Item object is successfully created with the correct room ID, damage value, and name
Negative damage value	Item(0, -5, "Broken Item")	Exception thrown with an error handling for negative damage values (damage should not be negative)
Null or empty item name	Item(0, 10, null)	Exception thrown with an error handling for null or empty item name

Testing getCurrentRoomId()	Input	Expected Value
Valid item in a room	Call method for item “Revolver”	Correct room ID where the item is located

Testing getName()	Input	Expected Value
Retrieve item name	Call method for an item “Revolver”	Revolver

Testing getDamage()	Input	Expected Value
Item with valid damage value	Call method for "Revolver"	Correct damage value is returned

Target Class

Testing Constructor Target(RoomInterface currentRoom, int health, String name)	Input	Expected Value
Valid input	Target(new Room(), 50, "Doctor Lucky")	Target object is successfully created with the correct initial room, health, and name
Invalid health value (negative or zero)	Target(new Room(), -10, "Sick Doctor")	Exception with error handling for invalid health value (health should be positive)
Null or empty name	Target(new Room(), 50, null)	Exception with error handling for null or empty name

Testing move(Room room)	Input	Expected Value
Move target to a valid room	Move target to the Armory	Target successfully moves to the specified room
Move to non-existent room	Move target to an invalid room ID	Exception or error returned

Testing getCurrentRoom()	Input	Expected Value
Target in a room	Call method after moving target to Library	The correct room of Library is returned

Testing getHealth()	Input	Expected Value
Retrieve target health	Call method after world is initialized	Health value of 50 (based on the mansion.txt)