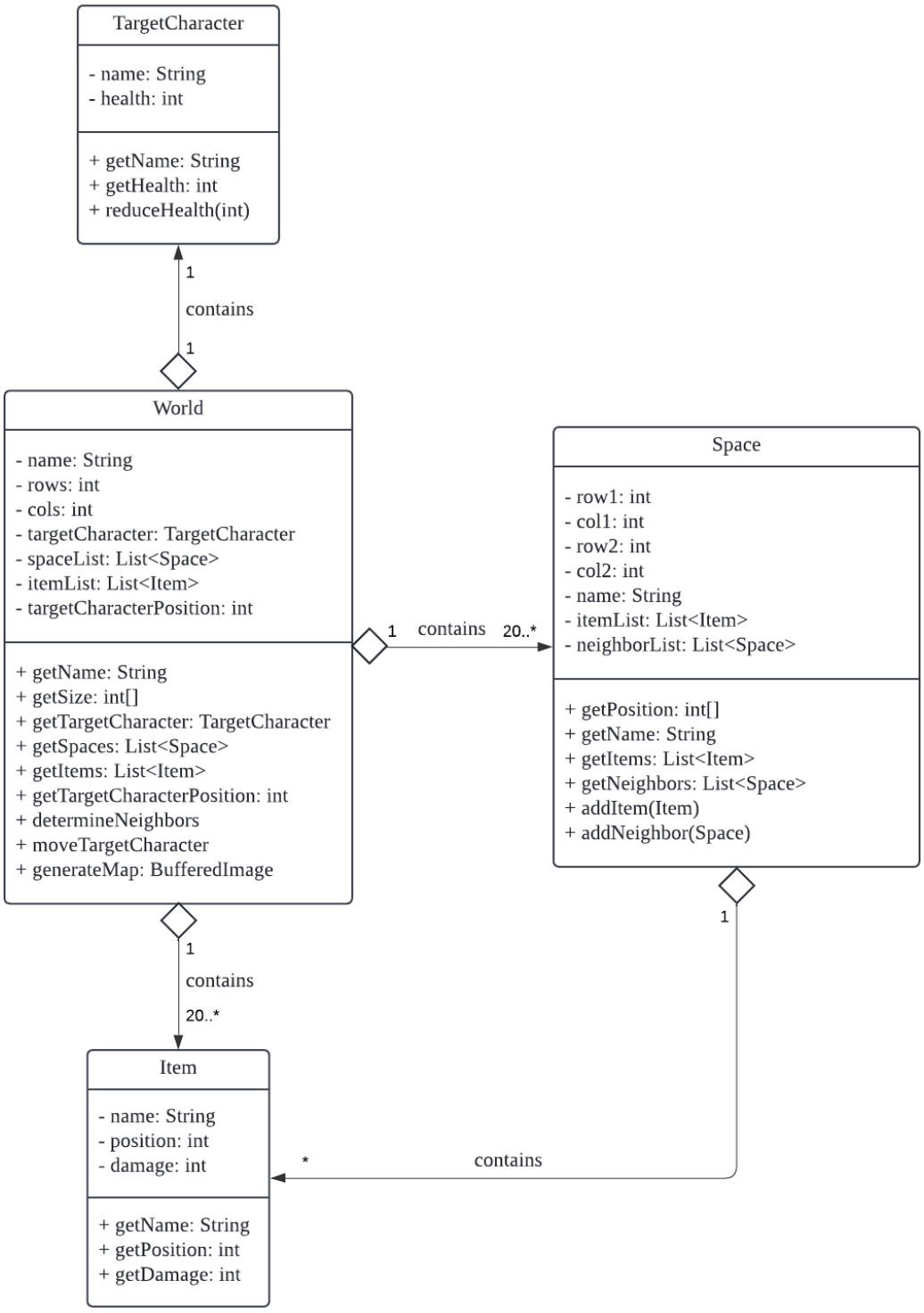
CS5010 Milestone1 Preliminary Design

By Linhao Qian (NUID: 002325915)

UML Diagram



Testing Plan

Testing design for TargetCharacter

Testing construction	Input	Expected Value
Constructor disallows empty name	TargetCharacter("", 50)	IllegalArgumentException
Constructor disallows non-positive health	TargetCharacter("doctor", 0)	IllegalArgumentException

Testing getName()	Input	Expected Value
TargetCharacter with normal name	TargetCharacter("Leo", 50)	"Leo"

Testing getHealth()	Input	Expected Value
TargetCharacter with positive health	TargetCharacter("Leo", 50)	50

Testing reduceHealth(int damage)	Input	Parameter	Actual Testing
Reduce positive health value	TargetCharacter("Leo", 50)	3	assertEquals(getHealth(), 47)

Testing design for Item

Testing construction	Input	Expected Value
Constructor disallows empty name	Item("", 0, 3)	IllegalArgumentException
Constructor disallows negative position	Item("Revolver", -1, 3)	IllegalArgumentException
Constructor disallows non-positive damage	Item("Revolver", 0, 0)	IllegalArgumentException

Testing getName()	Input	Expected Value
Item with normal name	Item("Revolver", 0, 3)	"Revolver"

Testing getPosition()	Input	Expected Value

Item with non-negative position	Item(“Revolver”, 0, 3)	0

Testing getDamage()	**Input**	**Expected Value**
Item with positive damage	Item(“Revolver”, 0, 3)	3

Testing design for Space

Testing construction	**Input**	**Expected Value**
Constructor disallows negative row1	Space (-1, 0, 3, 3, “Kitchen”)	IllegalArgumentException
Constructor disallows negative col1	Space (0, -1, 3, 3, “Kitchen”)	IllegalArgumentException
Constructor disallows the value of row2 to be less than the value of row1	Space (0, 0, -3, 3, “Kitchen”)	IllegalArgumentException
Constructor disallows the value of col2 to be less than the value of col1	Space (0, 0, 3, -3, “Kitchen”)	IllegalArgumentException
Constructor disallows empty name	Space (0, 0, 3, 3, “”)	IllegalArgumentException

Testing getPosition()	**Input**	**Expected Value**
Space with correct position	Space (0, 0, 3, 3, “Kitchen”)	new int[]{0, 0, 3, 3}

Testing getName()	**Input**	**Expected Value**
Space with normal name	Space (0, 0, 3, 3, “Kitchen”)	“Kitchen”

Testing addItem(Item item) and getItems()	**Operation**	**Actual Testing**

Add an item	addItem(new Item(“Revolver”, 0, 3))	assertEquals(getItems().get(0).getName(), “Revolver”);
		assertEquals(getItems().get(0).getPosition(), 0);
		assertEquals(getItems().get(0).getDamage(), 3);

Testing addNeighbor(Space space) and getNeighbors()	Operation	Actual Testing
Add a neighbor	addNeighbor(new Space(0, 3, 3, 6, “Armory”)	assertEquals(Arrays.equals(getNeighbors.get(0).getPosition(), new int[]{0, 3, 3, 6}), true); assertEquals(getNeighbors().get(0).getName(), “Armory”);

Testing design for World

Testing construction	Input	Expected Value
Constructor disallows empty name	World (“”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	IllegalArgumentException
Constructor disallows non-positive rows	World (“Mansion”, 0, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	IllegalArgumentException
Constructor disallows non-positive cols	World (“Mansion”, 40, 0, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	IllegalArgumentException

Testing construction	Testing ideas
Constructor disallows items with incorrect position	The position of an item should be less than the length of the spaceList.
Constructor disallows overlapping spaces	A space should not be overlapped by any other one. (i.e., the row1(row2) value of a space should not be between any other space’s row1 and row2 if its col1(col2) value is between any other space’s col1 and col2.
Constructor disallows space beyond boundaries	The row2 of a space should be less than the rows of the world, and the col2 of a space should be less than the cols of the world.

Testing getName()	Input	Expected Value
World with normal name	World (“Mansion”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20),	“Mansion”

	new ArrayList<Item>(20), 0)	

Testing getSize()	**Input**	**Expected Value**
World with positive size	World (“Mansion”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	new int[]{40, 30}

Testing getTargetCharacterPosition()	**Input**	**Expected Value**
The target character starts in space 0	World (“Mansion”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	0

Testing getTargetCharacter()	**Input**	**Actual Testing**
Get the target character	World (“Mansion”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	assertEquals (getTargetCharacter. getName(), “Leo”); assertEquals (getTargetCharacter. getHealth(), 50);

Testing moveTargetCharacter()	**Input**	**Operation**	**Actual Testing**
Move the target character from space to space in order	World (“Mansion”, 40, 30, new TargetCharacter(“Leo”, 50), new ArrayList<Space>(20), new ArrayList<Item>(20), 0)	moveTargetCharacter()	assertEquals (getTargetCharacterPosition(), 1)