Controller Tests:

Testing method	Testing	Input Value	Expected Value
testAddPlayer()	For testing add player	String, int, int	String
testGraphicalRepresentationOfWorldMap()	testing graphical representation of world		String
testGetPlayerInformation()	For testing the player information	Int playerId	String
testDetailsOfSpecifiedRoom()	testing the display of details of specified room	Int roomId	String
testPickWeapon()	testing the pick up weapon action	Int weaponId	String
testLookAround()	testing whether the neighbouring rooms are displayed correctly or not		String
testMovePlayer()	testing for moving the player	Int roomId	String

Command Tests:

Testing method	Testing	Input Value	Expected Value
testRunGame()	For testing add human player command class	Mock model	String
testRunGame()	For testing add computer player command class	Mock Model	String
testRunGame()	For testing the graphical representation of the world command class	Mock Model	String
testRunGame()	For testing the display player details command class	Mock Model	String
testRunGame()	For testing the display room details command class	Mock Model	String
testRunGame()	For testing look around action command class	Mock Model	String
testRunGame()	For testing move player action command class	Mock Model	String

`testRunGame()`	For testing the pick up weapon action command class	Mock Model	String
`testRunGame()`	For testing the moving pet action		
`testRunGame()`	For testing the attack on target action by any player		

Model Tests:

Player class Tests:

Testing Method	**Testing**	**Input Value**	**Expected Value**
`testInvalidPlayerAttributes()`	For testing invalid attribute values	Player class variables	`IllegalArgumentException`
`testValidPlayerAttributes()`	For testing the valid attribute values	Player class variables	Returns the player variable values according to the respective assert statements
`testMovePlayer()`	For testing move player action	roomIndex value	Returns a string value
`testAddWeapon()`	For testing the add weapon action	weaponId	Returns a String value
`testAddWeaponNegative()`	For testing the negative scenario of add weapon	weaponId	Returns a String value

Room class Test:

Testing Method	**Testing**	**Input Value**	**Expected Value**
`testInvalidRoomValues()`	For testing invalid room values	Room class variables	`IllegalArgumentException`
`testInvalidRoomName()`	For testing invalid room name	Room class variable values	`IllegalArgumentException`
`testgetRoomId()`	For testing the roomId	Room object	`roomId`

testgetRowUpperLeft()	For testing the row upper left coordinate	Room object	Row upperleft coordinate
testgetColUpperLeft()	For testing the column upper left coordinate	Room object	Column upper left coordinate
testgetRowLowerRight()	For testing the lower row right corodinate	Room object	Lower row right corner coordinate
testgetColLowerRight()	For testing lower right column coordinate	Room object	Lower column right corner coordinate
testgetRoomName()	For testing the valid room name	Room object	Room name
testAddWeapons()	For testing the add weapons action	Room object	String
testAddPlayer()	For testing the add player action	Room object	String

Target class tests:

Testing method	Testing	Input Value	Expected Value
testInvalidWeaponValues()	For testing invalid target values	Target object	IllegalArgumentException
testInvalidWeaponName()	For testing invalid target name	Target object	IllegalArgumentException
testGetTargetName()	For testing valid target name	Target object	Target name
testGetCurrentRoom()	For testing the current room of the target	Target object	Target current location
testGetTargetHealth()	For testing the target health value	Target object	Target health value
testMoveTarget()	Testing the target movement	Target object	Target current location index

WeaponItem class tests:

Testing method	Testing	Input Value	Expected Value

testInvalidWeaponItemValues()	For testing invalid weaponitem values	WeaponItem Object	IllegalArgumentException
testInvalidWeaponName()	For testing invalid weapon name	WeaponItem object	IllegalArgumentException
testGetRoomIndex()	For testing the room index of the weapon	WeaponItem object	Room index
testGetWeaponName()	For testing the weapon name	WeaponItem object	Weapon Name
testGetDamageAmount()	For testing the damage amount for each weapon	WeaponItem object	Damage amount
World class Tests:			
testDetermineNeighbours()	Testing the determine neighbours method	Room index	List<Room>
testDetailsOfSpecifiedRoom()	Testing the displaying of details of the specified room	Room index	String
testMoveTargetAroundTheWorld()	Testing the movement of the target		String
testGraphicalRepresentationOfWorldMap()	Testing the graphical representation of the world		String
testGetWeaponItemsForSpecifiedRoom()	Testing the list of weapons in a specified room	Room index	List<WeaponItem>

testAddPlayer()	Testing the add player action	Player object	String
testAddPlayerNotDone()	Testing add player action not done	Player object	IllegalArgumentException
testMovePlayer()	Testing the move player action	Player object	String
testGetPlayerDetails()	Testing the display of player details	playerId	String
testPickWeapon()	Testing the pick weapon action	weaponId	String
testWeaponPickedForNoWeaponsInBag()	Testing scenario when no weapons are there in the bag	weaponId	String
testLookAround()	Testing the look around action		String
testGetPlayerList()	Testing the get playerlist method	Room index	List<Player>
testGetCountTurn()	Testing the turn count	Current room index	String
testGetTurn()	Testing the get turn method		Player
getTarget()	Testing the getTarget method		Target object
getWeaponItems()	Test to get list of weapons in the world		List<WeaponItem>
testGetNoOfWeaponItems()	Test to get number of weaponItems		Integer
testGetNoOfWorldName()	Test to get world name		String
testGetNoOfColumns()	Test to get number of columns		Integer

testGetNoOfRows()	Test to get number of rows		Integer
testGetNoOfRooms()	Test to get number of rooms		Integer
testMovingPet()	Test for moving pet		String
testAttackOnTraget	Test for attack on target	String playerName	String
damageToTragetHealth	Test for damage to target health		
testDetermineVisibilityOfRooms()	Test for visibility of rooms		
testDetermineVisibilityOfPlayers	Test for visibility of players		
testWanderingpet()	Testing for *“depth-first traversal” implementation for the wandering pet*		
testAfterTheAttempt()	Test for the result of after the attempt on the target by any player		

Target pet test class:

Testing method	**Testing**	**Input Value**	**Expected Value**
testTargetPetName	Test for the target pet name		Target pet name
testForCurrentLocationOfPet	Test for the current location of the pet		Target pet current location
testInvalidParamenters	Test for invalid parameters		IllegalArgumentException