		Class: HRSController				
Method	**#**	**Test Description**	**Sample input Data**	**Expected Output**	**Actual Output**	**P/F**
createHotel	1	Create a hotel with a unique name and valid room count	hotelName: "Hotel A", numOfRooms: 10	Success message: "Hotel 'Hotel A' successfully created with 10 rooms."	Success message: "Hotel 'Hotel A' successfully created with 10 rooms."	P
	2	Attempt to create a hotel with a name that already exists	hotelName: "Hotel A", numOfRooms: 10	Error message: "Hotel name 'Hotel A' already exists."	Error message: "Hotel name 'Hotel A' already exists."	P
	3	Attempt to create a hotel with an invalid room count	hotelName: "Hotel B", numOfRooms: 0	Error message: "'0' number of rooms is not allowed (must be from 1 to 50 only)."	Error message: "'0' number of rooms is not allowed (must be from 1 to 50 only)."	P
getHotelGeneralInfo	1	Get info for an existing hotel	hotelName: "Hotel A"	HashMap with hotel details	HashMap with hotel details	P
	2	Get info for a non-existing hotel	hotelName: "Hotel X"	Empty HashMap	Empty HashMap	P
	3	Get info for a hotel with no reservations	hotelName: "Hotel B"	HashMap with zero reservations	HashMap with zero reservations	P
checkIfHotelExists	1	Check existence of an existing hotel	hotelName: "Hotel A"	"Hotel A"	"Hotel A"	P
	2	Check existence of a non-existing hotel	hotelName: "Hotel X"	null	null	P
	3	Check with empty hotel list	hotelName: "Hotel Y"	null	null	P
checkIfRoomExists	1	Check existence of an existing room in a hotel	roomName: "Mystic", hotelName: "Hotel A"	"Mystic"	"Mystic"	P
	2	Check existence of a non-existing room in a hotel	roomName: "Room X", hotelName: "Hotel A"	null	null	P
	3	Check room in a non-existing hotel	roomName: "Mystic", hotelName: "Hotel X"	null	null	P
checkIfReservationExists	1	Check existence of an existing reservation	hotelName: "Hotel A", roomName: "Mystic", reservationId: "1"	"1"	"1"	P
	2	Check existence of a non-existing reservation	hotelName: "Hotel A", roomName: "Mystic", reservationId: "999"	null	null	P
	3	Check reservation in a non-existing hotel	hotelName: "Hotel X", roomName: "Mystic", reservationId: "1"	null	null	P
getAllRoomInfoOnHotel	1	Get room info for a hotel with no rooms	hotelName: "Hotel A"	Empty ArrayList	Empty ArrayList	P
	2	Get room info for a hotel with one room	hotelName: "Hotel B"	ArrayList with one room's HashMap info	ArrayList with one room's HashMap info	P
	3	Get room info for a hotel with multiple rooms	hotelName: "Hotel C"	ArrayList with multiple rooms' HashMap info	ArrayList with multiple rooms' HashMap info	P
getAllDatesOnRoom	1	Get dates for a room in a hotel with no dates	hotelName: "Hotel A", roomName: "Mystic"	Empty ArrayList	Empty ArrayList	P
	2	Get dates for a room with one available date	hotelName: "Hotel B", roomName: "Golden"	ArrayList with one date's HashMap info	ArrayList with one date's HashMap info	P
	3	Get dates for a room with multiple dates	hotelName: "Hotel C", roomName: "Radiant"	ArrayList with multiple dates' HashMap info	ArrayList with multiple dates' HashMap info	P
getAllReservationInfoOnRoom	1	Get reservations for a room with no reservations	hotelName: "Hotel A", roomName: "Mystic"	Empty ArrayList	Empty ArrayList	P
	2	Get reservations for a room with one reservation	hotelName: "Hotel B", roomName: "Golden"	ArrayList with one reservation's HashMap info	ArrayList with one reservation's HashMap info	P
	3	Get reservations for a room with multiple reservations	hotelName: "Hotel C", roomName: "Radiant"	ArrayList with multiple reservations' HashMap info	ArrayList with multiple reservations' HashMap info	P
getAllRoomsAvailableOnDate	1	Get available rooms on a date range with no rooms available	hotelName: "Hotel A", checkInDate: 1, checkOutDate: 2	Empty ArrayList	Empty ArrayList	P
	2	Get available rooms on a date range with one room available	hotelName: "Hotel B", checkInDate: 1, checkOutDate: 2	ArrayList with one room's HashMap info	ArrayList with one room's HashMap info	P
	3	Get available rooms on a date range with multiple rooms available	hotelName: "Hotel C", checkInDate: 1, checkOutDate: 2	ArrayList with multiple rooms' HashMap info	ArrayList with multiple rooms' HashMap info	P
getSpecificRoomInfo	1	Get specific room info for an existing room	hotelName: "Hotel A", roomName: "Mystic"	HashMap with room details	HashMap with room details	P
	2	Get specific room info for a non-existing room	hotelName: "Hotel A", roomName: "Room X"	Empty HashMap	Empty HashMap	P
	3	Get room info for a room in a non-existing hotel	hotelName: "Hotel X", roomName: "Mystic"	Empty HashMap	Empty HashMap	P
getSpecificDateInfo	1	Valid hotel, room, and date number	"Hotel1", "Room1", 1	{"Date Number":"1", "Is Available":"Available", "Base Price":"100.0", "Modified Price":"90.0"}	{"Date Number":"1", "Is Available":"Available", "Base Price":"100.0", "Modified Price":"90.0"}	P
	2	Invalid hotel name	"InvalidHotel", "Room1", 1	null	null	P
	3	Valid hotel and room, but invalid date number	"Hotel1", "Room1", 999	null	null	P
getSpecificReservationInfo	1	Valid hotel, room, and reservation ID	"Hotel1", "Room1", 1	{"Id":"1", "Guest Name":"John Doe", "Check In Date":"1", "Check Out Date":"3", "Total Price":"300.0", "Discount C	{"Id":"1", "Guest Name":"John Doe", "Check In Date":"1", "Check Out Date":"3", "Total Price":"300.0", "Discount C	P
	2	Invalid hotel name	"InvalidHotel", "Room1", 1	null	null	P
	3	Valid hotel and room, but invalid reservation ID	"Hotel1", "Room1", 999	null	null	P
getRoomBookedDatesList	1	Valid hotel and room	Hotel1", "Room1"	["1", "2", "3"]	["1", "2", "3"]	P
	2	Invalid hotel name	"InvalidHotel", "Room1"	null	null	P
	3	Valid hotel, but room with no bookings	"Hotel1", "EmptyRoom	[]	[]	P
getRoomAvailableDatesList	1	Valid hotel and room	"Hotel1", "Room1"	["4", "5", "6"]	["4", "5", "6"]	P
	2	Invalid hotel name	"InvalidHotel", "Room1"	null	null	P
	3	Valid hotel, but room with no availability	"Hotel1", "FullRoom"	[]	[]	P
getPriceBreakdownOnReservationList	1	Valid hotel, room, and reservation ID	"Hotel1", "Room1", "1"	["Day 1-2: 100.0", "Day 2-3: 100.0"]	["Day 1-2: 100.0", "Day 2-3: 100.0"]	P
	2	Invalid hotel name	"InvalidHotel", "Room1", "1"	null	null	P
	3	Valid hotel and room, but invalid reservation ID	"Hotel1", "Room1", "999"	null	null	P
changeHotelName	1	Valid hotel name change	"OldHotelName", "NewHotelName"	TRUE	TRUE	P
	2	Change to an existing hotel name	"OldHotelName", "ExistingHotelName"	FALSE	FALSE	P
	3	Invalid hotel name	"InvalidHotel", "NewHotelName"	FALSE	FALSE	P
addRoom	1	Valid hotel and room type	"Hotel1", "Deluxe"	Success message	Success message	P
	2	Hotel name doesn't exist or invalid	"InvalidHotel", "Deluxe"	Error message	Error message	P
	3	Invalid room type	"Hotel1", "InvalidType"	Error message	Error message	P
removeRoom	1	Valid hotel, room with no reservations	"Hotel1", "Room1"	Success message	Success message	P
	2	Valid hotel, room with reservations	"Hotel1", "RoomWithReservations"	Error message	Error message	P
	3	Invalid hotel name	"InvalidHotel", "Room1"	Error message	Error message	P
updateBasePrice	1	Valid hotel, no reservations, valid base price	"Hotel1", 150.0	Success message	Success message	P
	2	Valid hotel, no reservations, invalid base price	"Hotel1", 50.0	Error message	Error message	P
	3	Valid hotel, with reservations	"HotelWithReservations", 150.0	Error message	Error message	P
removeReservation	1	Valid hotel, room, and reservation ID	"Hotel1", "Room1", 1	TRUE	TRUE	P
	2	Valid hotel, room, but invalid reservation ID	"Hotel1", "Room1", 999	FALSE	FALSE	P
	3	Invalid hotel name	"InvalidHotel", "Room1", 1	FALSE	FALSE	P
removeHotel	1	Valid hotel with no reservations	"Hotel1"	TRUE	TRUE	P
	2	Valid hotel with reservations	"HotelWithReservations"	FALSE	FALSE	P
	3	Invalid hotel name	"InvalidHotel"	FALSE	FALSE	P
datePriceModifier	1	Valid hotel, room, and date, valid modifier	"Hotel1", "Room1", 1, 20	TRUE	TRUE	P
	2	Valid hotel, room, but invalid date	"Hotel1", "Room1", 999, 20	FALSE	FALSE	P
	3	Invalid hotel name	"InvalidHotel", "Room1", 1, 20	FALSE	FALSE	P
areDatesValid	1	Valid date range	1, 3	TRUE	TRUE	P
	2	Invalid date range (start date after end date)	3, 1	FALSE	FALSE	P
	3	Invalid date (negative start date)	-1, 3	FALSE	FALSE	P
isARoomAvailable	1	Valid Date Range, Available Room	hotelName="Grand Hotel", roomType="Standard", checkInDate=1, checkOutDate=2	TRUE	TRUE	P
	2	Invalid Date Range (Check-in after Check-out)	hotelName="Grand Hotel", roomType="Deluxe", checkInDate=5, checkOutDate=3	FALSE	FALSE	P
	3	No Room Available	hotelName="Grand Hotel", roomType="Executive", checkInDate=1, checkOutDate=5	FALSE	FALSE	P
bookRoom	1	Valid Booking, No Discount	hotelName="Grand Hotel", guestName="John Doe", roomType="Standard", checkInDate=2, chec	TRUE, Success Message with Room Name and Dates	TRUE, Success Message with Room Name and Dates	P
	2	Valid Booking with Valid Discount Code ("STAY4_GET1")	hotelName="Grand Hotel", guestName="John Doe", roomType="Deluxe", checkInDate=5, check	TRUE: Success Message with Room Name, Dates, and Discount Applied	TRUE: Success Message with Room Name, Dates, and Discount Applied	P
	3	Booking with Invalid Discount Code	hotelName="Grand Hotel", guestName="John Doe", roomType="Single", checkInDate=16, check	FALSE: Error Message with Invalid Discount Code	FALSE: Error Message with Invalid Discount Code	P
isDiscountCodeValid	1	Valid Discount Code ("I_WORK_HERE")	discountCode="I_WORK_HERE"	TRUE	TRUE	P
	2	Invalid Discount Code	discountCode="NOT_VALID"	FALSE	FALSE	P
	3	Empty Discount Code	discountCode=""	FALSE	FALSE	P
isDiscountApplied	1	Valid Discount Code ("STAY4_GET1") with Stay Longer than 5 Days	Check In: 5, Check Out: 11, Discount Code: "STAY4_GET1"	TRUE	TRUE	P
	2	Valid Discount Code ("PAYDAY") with Reservation Covering Day 15	Check In: 14, Check Out: 17, Discount Code: "PAYDAY"	TRUE	TRUE	P
	3	Invalid Discount Code ("STAY4_GET1") with Stay Less than 5 Days	Check In: 25, Check Out: 27, Discount Code: "STAY4_GET1"	FALSE	FALSE	P
createTemporaryReservation	1	Valid Reservation with No Discount	hotelName="Grand Hotel", roomType="Standard", guestName="John Doe", checkInDate=1, chec	HashMap containing Reservation Details	HashMap containing Reservation Details	P
	2	Valid Reservation with Valid Discount Code ("I_WORK_HERE")	hotelName="Grand Hotel", roomType="Standard", guestName="John Doe", checkInDate=7, chec	HashMap containing Reservation Details with Discount Code applied	HashMap containing Reservation Details with Discount Code applied	P
	3	Reservation for Non-Existent Hotel	hotelName="Non-Hotel", roomType="Standard", guestName="John Doe", checkInDate=1, check	null	null	P
findHotelByName	1	Hotel found	hotelName="Grand Hotel"	Hotel object	Hotel object	P
	2	Hotel not found	hotelName="Nonexistent Hotel"	null	null	P
	3	Empty hotelName	hotelName=""	null	null	P
getHotels	1	Returns list of hotels		ArrayList of Hotel objects	ArrayList of Hotel objects	P
	2	Empty list of hotels (if applicable)		Empty ArrayList	Empty ArrayList	P
		Class: Hotel				
Method	**#**	**Test Description**	**Sample input Data**	**Expected Output**	**Actual Output**	**P/F**
Hotel	1	Initialize a hotel with valid name and number of rooms.	"Hilton", 5	Hotel "Hilton" with 5 rooms initialized with base price 1299.0.	Hotel "Hilton" with 5 rooms initialized with base price 1299.0.	P

	2	Initialize a hotel with invalid number of rooms (<1).	"Marriott", -5	Number of rooms set to 1 with base price 1299.0.	Number of rooms set to 1 with base price 1299.0.	P
	3	Initialize a hotel with invalid number of rooms (>50).	"Hyatt", 100	Number of rooms set to 1 with base price 1299.0.	Number of rooms set to 1 with base price 1299.0.	P
getName	1	Retrieve the name of the hotel.	Hotel "Sheraton"	"Sheraton"	"Sheraton"	P
	2	Retrieve the name of the hotel after name change.	Hotel "Sheraton Grand"	"Sheraton Grand"	"Sheraton Grand"	P
setName	3	Change the name of the hotel to a new unique name.	"Sheraton" -> "Sheraton Grand"	Name changed to "Sheraton Grand".	Name changed to "Sheraton Grand".	P
	1	Change the name of the hotel to an already existing name.	"Sheraton Grand" -> "Hilton"	"Name already exists."	"Name already exists."	P
getNumOfRooms	1	Retrieve the current number of rooms in the hotel.	Hotel "Hilton"	5 (after adding/removing rooms as applicable).	5 (after adding/removing rooms as applicable).	P
	2	Retrieve the current number of rooms after changes.	Hotel "Hilton"	Number adjusted to changes made in the test cases.	Number adjusted to changes made in the test cases.	P
setNumOfRooms	3	Set the number of rooms ensuring within valid range.	"Hilton", 10	Number of rooms set to 10.	Number of rooms set to 10.	P
	1	Set the number of rooms to an invalid number (>50).	"Hilton", 60	Number of rooms set to 1 with base price 1299.0.	Number of rooms set to 1 with base price 1299.0.	P
	2	Set the number of rooms to an invalid number (<1).	"Hilton", 0	Number of rooms set to 1 with base price 1299.0.	Number of rooms set to 1 with base price 1299.0.	P
getMaxRooms	3	Retrieve the maximum number of rooms allowed.	Hotel "Hilton"	50	50	P
getAvailableRoomsOnDate	1	Retrieve the list of available rooms for a date range.	"Hilton", 5, 10	List of available rooms (Room objects)	List of available rooms (Room objects)	P
	2	Retrieve the list of available rooms when none are free.	"Hilton", 5, 10	Empty list	Empty list	P
	3	Retrieve the list of available rooms with reservations.	"Hilton", 5, 10	List of available rooms considering existing reservations.	List of available rooms considering existing reservations.	P
areEmptyReservations	1	Check if all rooms have no reservations.	Hotel "Hilton"	true (if all rooms have no reservations)	true (if all rooms have no reservations)	P
	2	Check if some rooms have reservations.	Hotel "Hilton"	false (if some rooms have reservations)	false (if some rooms have reservations)	P
getBasePrice	3	Retrieve the base price for rooms in the hotel.	Hotel "Hilton"	1299	1299	P
	1	Retrieve the base price after changing it.	Hotel "Hilton"	Updated base price	Updated base price	P
setBasePrice	2	Set the base price for rooms assuming no reservations.	"Hilton", 1499.0	Base price set to 1499.0 for all rooms.	Base price set to 1499.0 for all rooms.	P
	3	Set the base price with existing reservations.	"Hilton", 1599.0	"Cannot update price due to existing reservations."	"Cannot update price due to existing reservations."	P
getEstimatedEarnings	1	Retrieve the estimated total earnings from all rooms.	Hotel "Hilton"	Total earnings from all rooms based on their reservations.	Total earnings from all rooms based on their reservations.	P
	2	Retrieve the estimated earnings when there are no reservations.	Hotel "Hilton"	0 (if no reservations exist).	0 (if no reservations exist).	P
getUniqueNamesList	1	Get Initial Unique Names List (Empty Used Names List)		Non-empty list containing all unique room names (e.g., "Azure", "Moonlight", etc.)	Non-empty list containing all unique room names (e.g., "Azure", "Moonlight", etc.)	P
getUsedNamesList	1	Get Unique Names List After Adding Rooms (Populated Used Names List)	(assuming rooms are added)	Non-empty list with some unique names removed (depending on added rooms)	Non-empty list with some unique names removed (depending on added rooms)	P
getUniqueName	1	Get Unique Name (Available Names List)		A unique name from the uniqueNamesList (e.g., "Azure")	A unique name from the uniqueNamesList (e.g., "Azure")	P
	2	Get Unique Name (No Available Names)	(assuming all unique names are used)	Fallback name "Roomu" + (number of rooms + 1) (e.g., "Roomu51")	Fallback name "Roomu" + (number of rooms + 1) (e.g., "Roomu51")	P
getRooms	1	Get Rooms After Initialization	(assuming rooms are initialized)	Non-empty list containing initialized Room objects	Non-empty list containing initialized Room objects	P
	2	Get Rooms After Adding Rooms	(assuming rooms are added)	Updated list containing additional Room objects	Updated list containing additional Room objects	P
initializeRooms	1	Initialize Rooms with Valid Number	numOfRooms = 10	Creates 10 Rooms with unique names and base price	Creates 10 Rooms with unique names and base price	P
	2	Initialize Rooms with Invalid Number (Too High)	numOfRooms = 100	No rooms created (or error message)	No rooms created (or error message)	P
	3	Initialize Rooms with Invalid Number (Too Low)	numOfRooms = 0	No rooms created (or sets numOfRooms to default 1)	No rooms created (or sets numOfRooms to default 1)	P
getRoom	1	Get Existing Room by Name	roomName = "Azure" (assuming a room with this name exists)	Room object with name "Azure"	Room object with name "Azure"	P
	2	Get Non-Existing Room by Name	roomName = "Nonexistent Room"	null	null	P
addRoom	1	Add Room with Valid Room Type	roomType = "Deluxe"	Room is added with unique name and room type	Room is added with unique name and room type	P
	2	Add Room When Hotel is Full	(assuming max rooms are reached)	Message indicating hotel is full and room cannot be added	Message indicating hotel is full and room cannot be added	P
	3	Add Room with Invalid Room Type	roomType = "Invalid Type"	Room is not added (or error message)	Room is not added (or error message)	P
removeRoom	1	Remove Existing Room by Name	roomName = "Azure" (assuming a room with this name exists)	Room is removed from the Hotel's list	Room is removed from the Hotel's list	P
	2	Remove Non-Existing Room by Name	roomName = "Nonexistent Room"	No change in Hotel's rooms list (or message indicating room not found)	No change in Hotel's rooms list (or message indicating room not found)	P
getAvailableRoomsOnDate(with roomType)	1	Get Available Rooms by Type and Date with Available Rooms	checkInDate = 1, checkOutDate = 3, roomType = "Standard"	List of available suite rooms for the specified dates	List of available suite rooms for the specified dates	P
	2	Get Available Rooms by Type and Date with No Available Rooms	checkInDate = 1, checkOutDate = 3, roomType = "Deluxe" (assuming no suite rooms available)	Empty list	Empty list	P
getRoomsByType	1	Get Rooms by Type with Existing Rooms	roomType = "Standard"	List of rooms with type "Standard"	List of rooms with type "Standard"	P
	2	Get Rooms by Type with No Matching Rooms	roomType = "InvalidType"	Empty list	Empty list	P
		Class: Reservation				
Method	**#**	**Test Description**	**Sample input Data**	**Expected Output**	**Actual Output**	**P/F**
Reservation	1	Create a reservation with valid details.	guestName: "John Doe", room: Room("Room-1", 1299.0), checkInDate: 1, checkOutDate: 5	Reservation object with given details	Reservation object with given details	P
	2	Create a reservation with a single-day stay.	guestName: "Jane Doe", room: Room("Room-2", 1299.0), checkInDate: 1, checkOutDate: 1	Reservation object with single-day stay details	Reservation object with single-day stay details	P
	3	Create a reservation with same check-in and check-out.	guestName: "Alice Smith", room: Room("Room-3", 1299.0), checkInDate: 2, checkOutDate: 2	Reservation object with check-in and check-out on the same date	Reservation object with check-in and check-out on the same date	P
getGuestName	1	Retrieve the guest name from the reservation.	guestName: "John Doe"	"John Doe"	"John Doe"	P
	2	Retrieve the guest name from a different reservation.	guestName: "Jane Doe"	"Jane Doe"	"Jane Doe"	P
getCheckInDate	1	Retrieve the check-in date from the reservation.	checkInDate: 1	1	1	P
	2	Retrieve the check-in date from another reservation.	checkInDate: 2	2	2	P
getCheckOutDate	1	Retrieve the check-out date from the reservation.	checkOutDate: 5	5	5	P
	2	Retrieve the check-out date from another reservation.	checkOutDate: 2	2	2	P
getRoom	1	Retrieve the room details from the reservation.	room: Room("Room-1", 1299.0)	Room object with name "Room-1" and price 1299.0	Room object with name "Room-1" and price 1299.0	P
	2	Retrieve the room details from another reservation.	room: Room("Room-2", 1299.0)	Room object with name "Room-2" and price 1299.0	Room object with name "Room-2" and price 1299.0	P
getTotalPrice	1	Calculate total price for multi-day reservation.	checkInDate: 1, checkOutDate: 5, costPerNight: 1299.0	6495.0 (5 * 1299.0)	6495.0 (5 * 1299.0)	P
	2	Calculate total price for single-day reservation.	checkInDate: 1, checkOutDate: 1, costPerNight: 1299.0	2598.0 (1 * 1299.0 + 1299.0)	2598.0 (1 * 1299.0 + 1299.0)	P
	3	Calculate total price with same check-in and check-out.	checkInDate: 2, checkOutDate: 2, costPerNight: 1299.0	2598.0 (0 * 1299.0 + 1299.0)	2598.0 (0 * 1299.0 + 1299.0)	P
getCostPerNight	1	Retrieve the cost per night for the reserved room.	room: Room("Room-1", 1299.0)	1299	1299	P
	2	Retrieve the cost per night from another reservation.	room: Room("Room-2", 1499.0)	1499	1499	P
getId	1	Retrieve the ID of the reservation				P
getDiscountCode	1	Get the discount code		Returns the Discount Code set by the user	Returns the Discount Code set by the user	P
	2	Get the default discount code ("N/A")		Returns the default Discount Code "N/A"	Returns the default Discount Code "N/A"	P
getTotalPrice	1	Get Total Price (No Discount)	checkInDate = 10, checkOutDate = 11, room price = 1000.00	1000.00 (one night)	1000.00 (one night)	P
	2	Get Total Price (Multiple Nights)	checkInDate = 5, checkOutDate = 8, room price = 1000.00	3000.00 (three nights)	3000.00 (three nights)	P
	3	Get Total Price with Discount (I_WORK_HERE)	checkInDate = 5, checkOutDate = 8, room price = 1000.00, discountCode = "I_WORK_HERE"	900.00 (10% discount)	900.00 (10% discount)	P
	4	Get Total Price with Discount (STAY4_GET1)	checkInDate = 5, checkOutDate = 9, room price = 1000.00, discountCode = "STAY4_GET1"	4000.00 (first night free for 5-night stay)	4000.00 (first night free for 5-night stay)	P
	5	Get Total Price with Discount (PAYDAY)	checkInDate = 14, checkOutDate = 16, room price = 1000.00, discountCode = "PAYDAY"	1850.00 (7% discount on 15th)	1850.00 (7% discount on 15th)	P
calculateTotalPrice	1	Calculate Raw Total Price (Single Night)	checkInDate = 10, checkOutDate = 11	Room price for checkInDate (consider using room.getPriceOnDate())	Room price for checkInDate (consider using room.getPriceOnDate())	P
	2	Calculate Raw Total Price (Multiple Nights)	checkInDate = 5, checkOutDate = 8	Sum of room prices for days 5, 6, and 7 (using room.getPriceOnDate())	Sum of room prices for days 5, 6, and 7 (using room.getPriceOnDate())	P
	3	Calculate Total Price with Discount (I_WORK_HERE)	checkInDate = 5, checkOutDate = 8, room price = 500.00, discountCode = "I_WORK_HERE"	450.00 (10% discount)	450.00 (10% discount)	P
	4	Calculate Total Price with Discount (STAY4_GET1)	checkInDate = 5, checkOutDate = 9, room price = 500.00, discountCode = "STAY4_GET1"	2000.00 (first night free for 5-night stay)	2000.00 (first night free for 5-night stay)	P
	5	Calculate Total Price with Discount (PAYDAY)	checkInDate = 14, checkOutDate = 16, room price = 500.00, discountCode = "PAYDAY"	925.00 (7% discount on 15th)	925.00 (7% discount on 15th)	P
	6	Calculate Total Price with Modified Prices	checkInDate = 5, checkOutDate = 8, room prices vary (using room.getPriceOnDate())	Sum of modified room prices	Sum of modified room prices	P
		Class: Room				
Method	**#**	**Test Description**	**Sample input Data**	**Expected Output**	**Actual Output**	**P/F**
Room	1	Create Room with Valid Data	name = "Ocean View", pricePerNight = 100.00, roomType = "Deluxe"	CREATED Room object with name "Ocean View", price 120.00 (Deluxe modifier), and roomType "Deluxe"	CREATED Room object with name "Ocean View", price 120.00 (Deluxe modifier), and roomType "Deluxe"	P
	2	Create Room with Empty Name	name = "", pricePerNight = 100.00, roomType = "Deluxe"	Impossible Case	Impossible Case	P
	3	Create Room with Invalid Price	name = "Ocean View", pricePerNight = -50.00, roomType = "Deluxe"	Impossible Case	Impossible Case	P
	4	Create Room with Invalid Room Type	name = "Ocean View", pricePerNight = 100.00, roomType = "Invalid"	Room object with name "Ocean View", price 100.00 (Standard price), and roomType "Standard"	Room object with name "Ocean View", price 100.00 (Standard price), and roomType "Standard"	P
Room (with Room parameter)	1	Create Room Copy	room (existing Room object)	New Room object with same properties as the original room	New Room object with same properties as the original room	
getName	1	Retrieve the name of the room.	name: "Room-1"	"Room-1"	"Room-1"	P
	2	Retrieve the name of a different room.	name: "Room-2"	"Room-2"	"Room-2"	P
getPricePerNight	1	Retrieve the price per night for the room.	pricePerNight: 1299.0	1299	1299	P
	2	Retrieve the price per night for a different room.	pricePerNight: 1499.0	1499	1499	P
setPricePerNight	1	Set a new price per night for the room.	price: 1599.0	pricePerNight updated to 1599.0	pricePerNight updated to 1599.0	P
	2	Set the price per night to zero.	price: 0.0	pricePerNight updated to 0.0	pricePerNight updated to 0.0	P
isAvailable	1	Check availability for an available date range.	checkIn: 1, checkOut: 5	TRUE	TRUE	P
	2	Check availability for a date range with reservations.	checkIn: 1, checkOut: 5 (after adding a reservation)	FALSE	FALSE	P
	3	Check availability for the same check-in and check-out	checkIn: 2, checkOut: 2	TRUE	TRUE	P
getReservations	1	Retrieve the list of reservations.	reservations: [Reservation("John Doe", Room, 1, 5)]	List with 1 reservation	List with 1 reservation	P

	2	Retrieve the list of reservations when empty.	reservations: []	Empty list	Empty list	P
getReservation	1	Retrieve a reservation by guest name.	guestName: "John Doe"	Reservation object with guest name "John Doe"	Reservation object with guest name "John Doe"	P
	2	Retrieve a reservation for a different guest.	guestName: "Jane Doe"	null	null	P
addReservation	1	Add Reservation for Available Dates	guestName = "John Doe", checkInDate = 12, checkOutDate = 14 (assuming days 12-14 are avai	Reservation is added and dates are marked unavailable	Reservation is added and dates are marked unavailable	P
	2	Add Reservation for Unavailable Dates	guestName = "Jane Doe", checkInDate = 12, checkOutDate = 14 (assuming days 12-14 are alre	Reservation is not added, error message or return false	Reservation is not added, error message or return false	P
	3	Add Reservation for Single Day	guestName = "Alice", checkInDate = 20, checkOutDate = 20	Reservation is added for one day	Reservation is added for one day	P
removeReservation	1	Remove Existing Reservation	Id = 3 (assuming reservation with Id 3 exists)	Reservation is removed and corresponding dates are marked available	Reservation is removed and corresponding dates are marked available	P
	2	Remove Non-Existing Reservation	Id = 10 (assuming no reservation with Id 10)	No changes made	No changes made	P
reserveDates	1	Reserve dates for a room.	checkInDate: 1, checkOutDate: 5	Dates marked unavailable	Dates marked unavailable	P
	2	Reserve already booked dates.	checkInDate: 1, checkOutDate: 5 (after reserving for John Doe)	FALSE	FALSE	P
	3	Reserve dates with same check-in and check-out.	checkInDate: 2, checkOutDate: 2	Date marked unavailable	Date marked unavailable	P
cancelReserveDates	1	Cancel reserved dates for a room.	checkInDate: 1, checkOutDate: 5	Dates marked available	Dates marked available	P
	2	Cancel dates that were not reserved.	checkInDate: 6, checkOutDate: 10	No change (dates remain available)	No change (dates remain available)	P
getTotalEarnings	1	Calculate total earnings from reservations.	reservations: [Reservation("John Doe", Room, 1, 5)], pricePerNight: 1299.0	Total earnings: 6495.0	Total earnings: 6495.0	P
	2	Calculate total earnings with no reservations.	reservations: [], pricePerNight: 1299.0	Total earnings: 0.0	Total earnings: 0.0	P
getAvailabilityDates	1	Get all Availability Dates		List of all availability dates	List of all availability dates	P
getAvailabilityDate	1	Get a Availability Date	dateNumber = 15	AvailabilityDate object for day 15	AvailabilityDate object for day 15	P
getReservedDates	1	Get Reserved Dates		List of dates that are not available	List of dates that are not available	P
getPriceOnDate	1	Get Price for Available Date	selectedDate = 10 (assuming day 10 is available)	Price for day 10 (base price)	Price for day 10 (base price)	P
	2	Get Price for Modified Date	selectedDate = 15 (assuming day 15 has a modified price due to a reservation)	Modified price for day 15	Modified price for day 15	P
		Class: AvailabilityDate				
Method	**#**	**Test Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
AvailabilityDate	1	Create AvailabilityDate with Valid Data	dateNumber = 10, basePrice = 1000.00	AvailabilityDate object with dateNumber 10, basePrice 1000.00, available, modifiedPrice 1000.00, isModified fals	AvailabilityDate object with dateNumber 10, basePrice 1000.00, available, modifiedPrice 1000.00, isModified fals	P
	2	Create AvailabilityDate with Base Price Below Minimum	dateNumber = 10, basePrice = 499.00	AvailabilityDate object with dateNumber 10, basePrice 500.00 (default minimum), available, modifiedPrice 500.0	AvailabilityDate object with dateNumber 10, basePrice 500.00 (default minimum), available, modifiedPrice 500.0	P
	3	Create AvailabilityDate with Zero Base Price	dateNumber = 10, basePrice = 0.00	AvailabilityDate object with dateNumber 10, basePrice 500.00 (default minimum), available, modifiedPrice 500.0	AvailabilityDate object with dateNumber 10, basePrice 500.00 (default minimum), available, modifiedPrice 500.0	P
AvailabilityDate (AvailabilityDate as parameter)	1	Create AvailabilityDate Copy	date (existing AvailabilityDate object)	New AvailabilityDate object with same properties as the original	New AvailabilityDate object with same properties as the original	P
	2	Create AvailabilityDate Copy with Modified Values	date (existing AvailabilityDate object with modified values)	New AvailabilityDate object with copied modified values	New AvailabilityDate object with copied modified values	P
	3	Create AvailabilityDate Copy with Null Input	date = null	Exception or handle null input gracefully	Exception or handle null input gracefully	P
getDateNumber	1	Get date number of the date	5 (assume initialized)	5	5	P
isAvailable	1	Get the availability (boolean)	isAvailable = true	TRUE	TRUE	P
setAvailable	1	Changes the availability state (boolean)	FALSE	FALSE	FALSE	P
getBasePrice	1	Get the base price	basePrice = 1000	1000	1000	P
setBasePrice	1	Set a new base price	basePrice to 2000	base price will now be 2000	base price will now be 2000	P
getModifiedPrice	1	Get the modified price that is modified by the date price modifier	modifiedPrice (after being modified by date price modifier) = 3500	3500	3500	P
getIsModified	1	Get the boolean state if its modified	isModified = false	FALSE	FALSE	P
setIsModified	1	Set the boolean state if modified	isModified to true	TRUE	TRUE	P
getModifier	1	Get the modifier value to be multiplied	modifier = 2.0	2	2	P
setModifier	1	Set the modifier value to be multiplied	modifier to 3.5	3.5	3.5	P
setModifiedPrice	1	Set modified price to 1.0	newModifier = 1.0	ModifiedPrice is set to basePrice, isModified is set to false, modifier is set to 1.0	ModifiedPrice is set to basePrice, isModified is set to false, modifier is set to 1.0	P
	2	Set modified price to valid value within range (0.5 - 1.5)	newModifier = 1.2	ModifiedPrice is calculated, isModified is set to true, modifier is set to 1.2	ModifiedPrice is calculated, isModified is set to true, modifier is set to 1.2	P
	3	Set modified price to invalid value (less than 0.5)	newModifier = 0.4	ModifiedPrice is not changed, isModified is set to false, modifier is not changed	ModifiedPrice is not changed, isModified is set to false, modifier is not changed	P
	4	Set modified price to invalid value (greater than 1.5)	newModifier = 1.6	ModifiedPrice is not changed, isModified is set to false, modifier is not changed	ModifiedPrice is not changed, isModified is set to false, modifier is not changed	P