OBJETIVO
Desarrollar un sistema para gestionar reservas de hotel

ESPECIFICACIONES

1. Enum EstadoReserva
Crear un enum con los siguientes valores:

PENDIENTE - Reserva creada pero no confirmada
CONFIRMADA - Reserva confirmada por el cliente
CHECK_IN - Huésped ingresó al hotel
CHECK_OUT - Huésped salió del hotel
CANCELADA - Reserva cancelada

2. Clase Reserva
Atributos privados:

codigoReserva (String)
nombreHuesped (String)
numeroHabitacion (int)
cantidadNoches (int)
estado (EstadoReserva)
fechaIngreso (LocalDate)

Constructor:

Debe recibir: código, nombre del huésped, número de habitación y cantidad de noches
Estado inicial: PENDIENTE
Fecha de ingreso: fecha actual del sistema

Métodos:

Todos los getters necesarios
Setter solo para estado
toString() que retorne: Código: [código] | Huésped: [nombre] | Habitación: [número] | Noches: [cantidad] | Estado: [estado]

3. Excepción ReservaNoEncontradaException

Debe ser una excepción verificada (checked exception)
Constructor que reciba un mensaje (String)

4. Clase SistemaHotel
Atributos privados:

Un HashMap<String, Reserva> para guardar reservas (clave: código de reserva)
Un HashMap<Integer, String> para habitaciones ocupadas (clave: número de habitación, valor: código de reserva)

Constante:

TOTAL_HABITACIONES = 50

Constructor:

Inicializar los HashMaps vacíos

Métodos a implementar:
4.1 agregarReserva(Reserva reserva)

Verificar que la habitación NO esté ocupada
Si está ocupada: lanzar IllegalStateException con mensaje: "La habitación [número] ya está ocupada"
Si está disponible:

Agregar la reserva al HashMap de reservas
Marcar la habitación como ocupada en el HashMap de habitaciones

4.2 confirmarReserva(String codigoReserva)

Buscar la reserva por código
Si no existe: lanzar ReservaNoEncontradaException
Cambiar su estado a CONFIRMADA

4.3 hacerCheckIn(String codigoReserva)

Buscar la reserva por código
Si no existe: lanzar ReservaNoEncontradaException
Cambiar su estado a CHECK_IN
La habitación permanece ocupada

4.4 hacerCheckOut(String codigoReserva)

Buscar la reserva por código
Si no existe: lanzar ReservaNoEncontradaException
Cambiar su estado a CHECK_OUT
LIBERAR la habitación del HashMap de habitaciones ocupadas

4.5 cancelarReserva(String codigoReserva)

Buscar la reserva por código
Si no existe: lanzar ReservaNoEncontradaException
Cambiar su estado a CANCELADA
LIBERAR la habitación del HashMap de habitaciones ocupadas

4.6 buscarReservaPorCodigo(String codigo)

Buscar y retornar la reserva
Si no existe: lanzar ReservaNoEncontradaException con mensaje: "Reserva [código] no encontrada"
**Deben crear al excepción

4.7 obtenerReservasPorEstado(EstadoReserva estado)

Retornar un ArrayList<Reserva> con todas las reservas que tengan ese estado
Recorrer el HashMap de reservas con un loop

4.8 generarReporteOcupacion()

Retornar un String con el siguiente formato:

Total reservas: [cantidad]
Habitaciones ocupadas: [cantidad]
Ocupación: [porcentaje]%

5. Clase Main - Programa de Prueba
****El main debe producir EXACTAMENTE este output:

=== SISTEMA DE RESERVAS HOTEL ===

Agregando reservas...
✓ Reserva R001 agregada (Habitación 101 ocupada)
✓ Reserva R002 agregada (Habitación 102 ocupada)
✗ Error: La habitación 101 ya está ocupada

Confirmando reserva R001...
✓ Reserva R001 confirmada

Haciendo check-in de R001...
✓ Check-in realizado para R001

Buscando reserva R001:
Código: R001 | Huésped: Juan Pérez | Habitación: 101 | Noches: 3 | Estado: CHECK_IN

Haciendo check-out de R001...
✓ Check-out realizado (Habitación 101 liberada)

Cancelando reserva R002...
✓ Reserva R002 cancelada (Habitación 102 liberada)

=== REPORTE FINAL ===
Total reservas: 2
Habitaciones ocupadas: 0
Ocupación: 0.0%

Reservas en estado CHECK_OUT:
- Código: R001 | Huésped: Juan Pérez | Habitación: 101 | Noches: 3 | Estado: CHECK_OUT