Sistema de Reservas de Habitaciones de un Hotel

2249338 - Jheffrey Urbano Garces  
 Alexander Mosquera Fajardo  
 Laura Vanessa Torres

Contenido

[Análisis del problema: 1](#_Toc194778788)

[Estructura Hotel: 1](#_Toc194778789)

[Funcionalidades: 1](#_Toc194778790)

[Requisitos: 1](#_Toc194778791)

[Algoritmo en Pseudocódigo: 2](#_Toc194778792)

[Evidencias de ejecución del programa: 6](#_Toc194778793)

# Análisis del problema:

## Estructura Hotel:

* 5 pisos, 6 habitaciones por piso (Total: 30 habitaciones).
* Tipos de habitación: 2, 3 y 4 personas.

## Funcionalidades:

* Hacer la reserva
* Mostrar todas las reservas
* Cancelar reserva
* Mostrar las habitaciones disponibles.

## Requisitos:

* Abstracción, herencia, encapsulamiento, polimorfismo (overloading y overwritting).
* Código documentado (PEP8), clases separadas en archivos, así como el programa principal y desarrollada en Visual Studio Code.

|  |  |
| --- | --- |
| Sistema de Gestión de un Hotel | |
| Entrada:  **Datos huésped:** Nombre y apellido de la reserva: str documento: int  **Reserva:** tipo de habitación: str cantidad de personas: int cantidad de días: int | Salida:  **Confirmación de reserva:** id\_reserva: int costo\_total: int detalles\_habitación: f ”str” fecha\_inicio\_fin: date(DD/MM /YY)  **Listados:** mostrar\_reservas\_disponibles(): dict mostrar\_habitaciones\_disponibles(): dict |
| Restricciones:   * La cantidad de personas no puede exceder la capacidad de la habitación. * No se permite reservar una habitación ya ocupada. * El documento del huésped debe ser numérico. | |

# Algoritmo en Pseudocódigo:

Inicio

// Inicialización de estructuras

Definir habitaciones[1..30] como arreglo de registros con campos:

- estado: "libre" o "ocupada"

- tipo: entero (2, 3 o 4) // capacidad asignada

- piso: entero

- número: entero

- id\_habitacion: entero // Será calculado como (piso\*100 + número)

Inicializar cada habitación:

Para piso desde 1 hasta 5 Hacer

Para número desde 1 hasta 6 Hacer

índice ← (piso - 1) \* 6 + número

habitaciones[índice].piso ← piso

habitaciones[índice].número ← número

habitaciones[índice].id\_habitacion ← (piso \* 100) + número

habitaciones[índice].estado ← "libre"

// Asignar el tipo de habitación según la lógica definida (2, 3 o 4)

FinPara

FinPara

Definir lista\_reservas como lista vacía

contador\_reservas ← 0

// Bucle principal

Repetir

Escribir "Menú Principal:"

Escribir "1. Reservar habitación"

Escribir "2. Ver reservas activas"

Escribir "3. Cancelar reserva"

Escribir "4. Mostrar disponibilidad"

Escribir "5. Salir"

Leer opcion

Segun opcion Hacer

Caso 1: // Reservar habitación

Escribir "Ingrese el tipo de habitación (2, 3 o 4 personas):"

Leer tipo\_deseado

Si tipo\_deseado ≠ 2 Y tipo\_deseado ≠ 3 Y tipo\_deseado ≠ 4 Entonces

Escribir "Tipo de habitación inválido."

Sino

habitacion\_encontrada ← FALSO

Para i desde 1 hasta 30 Hacer

Si habitaciones[i].estado = "libre" Y habitaciones[i].tipo = tipo\_deseado Entonces

habitacion\_encontrada ← VERDADERO

Escribir "Ingrese nombre y apellido:"

Leer nombre\_cliente

Escribir "Ingrese documento (número):"

Leer documento

Escribir "Ingrese cantidad de días de la reserva:"

Leer dias\_reserva

contador\_reservas ← contador\_reservas + 1

id\_reserva ← contador\_reservas

costo\_habitacion ← (Determinar costo según tipo\_deseado) // función o tabla de costos

costo\_total ← costo\_habitacion \* dias\_reserva

Agregar a lista\_reservas el registro:

{ id: id\_reserva,

id\_habitacion: habitaciones[i].id\_habitacion,

piso: habitaciones[i].piso,

numero: habitaciones[i].número,

cliente: nombre\_cliente,

documento: documento,

dias: dias\_reserva,

costo\_total: costo\_total }

habitaciones[i].estado ← "ocupada"

Escribir "Reserva confirmada:"

Escribir "ID Reserva: ", id\_reserva

Escribir "Habitación: ", habitaciones[i].id\_habitacion

Escribir "Costo Total: ", costo\_total

Salir del ciclo Para

FinSi

FinPara

Si habitacion\_encontrada = FALSO Entonces

Escribir "No hay habitaciones disponibles para el tipo solicitado."

FinSi

FinSi

Caso 2: // Ver reservas activas

Si lista\_reservas está vacía Entonces

Escribir "No existen reservas activas."

Sino

Escribir "Listado de Reservas Activas:"

Para cada reserva en lista\_reservas Hacer

Escribir "ID: ", reserva.id,

" | Habitación: ", reserva.id\_habitacion,

" | Cliente: ", reserva.cliente,

" | Costo: ", reserva.costo\_total

FinPara

FinSi

Caso 3: // Cancelar reserva

Escribir "Ingrese el ID de la reserva a cancelar:"

Leer id\_cancelar

reserva\_encontrada ← FALSO

Para cada reserva en lista\_reservas con índice j Hacer

Si reserva.id = id\_cancelar Entonces

reserva\_encontrada ← VERDADERO

Para i desde 1 hasta 30 Hacer

Si habitaciones[i].id\_habitacion = reserva.id\_habitacion Entonces

habitaciones[i].estado ← "libre"

Salir del ciclo Para

FinSi

FinPara

Eliminar reserva en posición j de lista\_reservas

Escribir "Reserva cancelada exitosamente."

Salir del ciclo Para

FinSi

FinPara

Si reserva\_encontrada = FALSO Entonces

Escribir "ID de reserva no encontrado."

FinSi

Caso 4: // Mostrar disponibilidad

Escribir "Disponibilidad de Habitaciones:"

Para piso desde 1 hasta 5 Hacer

Escribir "Piso ", piso, ":"

Para cada habitación en habitaciones donde habitación.piso = piso Hacer

Si habitación.estado = "libre" Entonces

Escribir "Habitación ", habitación.id\_habitacion, " (Tipo ", habitación.tipo, ") ✔"

Sino

Escribir "Habitación ", habitación.id\_habitacion, " (Tipo ", habitación.tipo, ") ✖"

FinSi

FinPara

FinPara

Caso 5: // Salir

Escribir "Gracias por usar el sistema de reservas."

Salir del bucle principal

De Otro Modo:

Escribir "Opción inválida. Intente nuevamente."

FinSegun

Hasta que opcion = 5

Fin

# Evidencias de ejecución del programa:













