|  |  |  |
| --- | --- | --- |
| | GUÍA 2.1.2:Actividad Colecciones POO | | |
| Sigla | Asignatura | Experiencia de Aprendizaje |
| DSY | Desarrollo Orientado a Objetos | EA2: Herencia y Colecciones |
| Tiempo | Modalidad de Trabajo | Indicadores de logro |
| 1 h | Individual | IL2.1 , IL2.2 y IL2.3 |

|  |
| --- |
| **Código QR con relleno sólido**  **Antecedentes generales** |

## Esta guía tiene como objetivo reconocer y aplicar almacenamientos temporales, específicamente colecciones.

## Esta actividad tiene carácter formativo, es decir: es para visualizar lo que aprendes, en la directa medida que tú docente de asignatura te va retroalimentando constantemente, tanto a nivel individual como colectivo.

## Deberá dejar registro del trabajo realizado por medio de la plataforma de Blackboad

|  |
| --- |
| **Lista con relleno sólido Requerimientos para esta actividad** |

## En esta actividad, los estudiantes deberán identificar las entidades obtenidas del análisis del problema planteado, para definir las clases, sus atributos, métodos y un medio de almacenamiento temporal. Además, se debe implementar el código utilizando una colección para la solución del problema planteado

1. Realizar diagramar de clases
2. Programar en el lenguaje JAVA utilizando el IDE Netbeans el diagrama de clases generado
3. Mostrar los datos de la colección, a través de un menú que permita seleccionar la lista.
4. Agregar métodos necesarios para el funcionamiento de las clases
5. Realizar invocaciones desde main para todas las clases

|  |  |
| --- | --- |
| **Inteligencia artificial con relleno sólidoActividad** |  |

**Caso de Estudio: Gestión de Reservas de Hoteles con Colecciones (ArrayList)**

**Vista de una ciudad

Descripción generada automáticamente con confianza mediaContexto:** Una agencia de viajes desea mejorar su sistema de reservas de hoteles mediante el uso adecuado de colecciones, específicamente ArrayList. Se busca una colaboración eficiente entre las clases de Hotel, Cliente, Reserva y Habitación, aprovechando las capacidades de las colecciones para simplificar la gestión de información.

**Requerimientos Iniciales:** El sistema debe permitir a los clientes buscar hoteles, realizar reservas de habitaciones y gestionar sus transacciones. La utilización de ArrayList será esencial para almacenar y gestionar eficientemente la información de hoteles, clientes, reservas y habitaciones.

Modelo propuesto:

1. Hotel:
   * Atributos:
     + idHotel (String): Identificador único del hotel.
     + nombre (String): Nombre del hotel.
     + habitacionesDisponibles (ArrayList): Lista que contiene la información de las habitaciones disponibles.
2. Cliente:
   * Atributos:
     + idCliente (String): Identificador único del cliente.
     + nombre (String): Nombre del cliente.
     + reservasRealizadas (ArrayList): Lista que contiene la información de las reservas realizadas por el cliente.
3. Reserva:
   * Atributos:
     + hotel (Hotel): Hotel asociado a la reserva.
     + habitacionReservada (Habitacion): Habitación reservada por el cliente.
4. Habitacion:
   * Atributos:
     + numeroHabitacion (String): Número único de la habitación.
     + precioNoche (Double): Precio por noche de la habitación.
     + disponible (Boolean): Indica si la habitación está disponible para reserva.

Colaboración con ArrayList:

* Cuando un cliente realiza una reserva, se actualiza la lista reservasRealizadas en la clase Cliente, utilizando la capacidad de ArrayList para almacenar múltiples reservas.
* La clase Hotel colabora con la clase Habitacion mediante la lista habitacionesDisponibles, facilitando la gestión y actualización de las habitaciones disponibles.
* La clase Reserva utiliza ArrayList para almacenar información detallada sobre las reservas, permitiendo un seguimiento eficiente de la actividad del cliente.
* Este enfoque de colaboración con ArrayList mejora la eficiencia y flexibilidad del sistema de reservas de hoteles, permitiendo un manejo más dinámico y escalable de la información.