|  |  |  |
| --- | --- | --- |
| | 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 de colecciones, específicamente ArrayList. Se busca una colaboración eficiente entre las clases **Hotel**, **Cliente**, **Reserva** y **Habitación**, de manera que se pueda almacenar y gestionar de forma sencilla la información.

**Requerimientos Iniciales:**

* Permitir registrar hoteles, clientes y habitaciones.
* Crear y gestionar reservas.
* Listar hoteles, habitaciones disponibles y reservas por cliente.
* Usar ArrayList para almacenar y administrar la información de forma dinámica.

Modelo propuesto:

1. Hotel:
   * idHotel (String)
   * nombre (String)
   * habitaciones (ArrayList<Habitacion>)
2. Cliente:
   * idCliente (String)
   * nombre (String)
   * reservas (ArrayList<Reserva>)
3. Habitacion:
   * numeroHabitacion (String)
   * precioNoche (double)
   * disponible (boolean)
4. Reserva:
   * hotel (Hotel)
   * habitacion (Habitacion)
   * cliente (Cliente)
   * fecha (LocalDate)
   * noches (int)

**Colaboración con ArrayList:**

* Cliente guarda todas sus reservas en una lista.
* Hotel gestiona su listado de habitaciones.
* Al crear una Reserva, se actualiza tanto la colección de reservas del cliente como la disponibilidad de la habitación.

**Menú sugerido**

* Registrar hoteles
* Registrar clientes
* Agregar habitación a un hotel
* Listar hoteles
* Listar habitaciones disponibles de un hotel
* Crear reserva (cliente, hotel, número de habitación, noches)
* Listar reservas de un cliente
* Salir

Regla de negocio: Un cliente no puede reservar la *misma habitación* del *mismo hotel* para el *mismo día*.