**Informe De Análisis Y Diseño Del Desafió I**

Carlos Andrés Buelvas y Juan José Medina Mejía

Facultad de ingeniería, Universidad de Antioquia, Medellín

Informática II

Aníbal Guerra Soler y Augusto Salazar

17 de abril de 2025

Imagen que contiene Patrón de fondo

Descripción generada automáticamente

**1. Análisis del Problema**

El sistema UdeAStay modela un mercado de estadías hogareñas con cuatro entidades principales: **Anfitriones, Huéspedes, Alojamientos y Reservaciones**. Estas entidades se relacionan de la siguiente forma:

* Un **Anfitrión** puede tener múltiples **Alojamientos**.
* Un **Huésped** puede tener múltiples **Reservaciones**, siempre que no se solapen en fechas.
* Un **Alojamiento** está asociado a un único **Anfitrión** y puede estar reservado durante fechas específicas.
* Una **Reservación** vincula un **Huésped**, un **Alojamiento**, y un rango de fechas.

**2. Base de Datos en Memoria (Diseño y Estructura)**

Para minimizar la **duplicación de datos** y optimizar el **uso de memoria y las búsquedas**, se ha definido una base de datos en memoria con las siguientes características:

* Se usarán **cuatro arreglos dinámicos**:
  1. Alojamiento\* alojamientos
  2. Anfitrion\* anfitriones
  3. Huesped\* huespedes
  4. Reserva\* reservasActivas (y un quinto para el histórico)
* Cada objeto mantiene **punteros a sus relaciones**:
  1. Alojamiento tiene un Anfitrion\*
  2. Reserva tiene un Huesped\* y un Alojamiento\*

**Ventajas de este diseño:**

* Elimina redundancia: los datos del anfitrión o del huésped no se replican en cada alojamiento o reserva.
* Mejora eficiencia de acceso: al centralizar las relaciones mediante punteros, se facilita la validación de datos y el cálculo de consumo de memoria.
* Simplifica el manejo del histórico de reservas: el arreglo reservasActivas se puede recorrer de forma directa sin acceder a cada alojamiento.

**3. Estructura de Archivos TXT (Almacenamiento Persistente)**

Se definieron **cinco archivos** de almacenamiento externo, con formato delimitado por ; para facilitar la lectura secuencial:

1. **anfitriones.txt**

documento;antiguedad;puntuacion

12345678;24;4.7

1. **huespedes.txt**

documento;antiguedad;puntuacion

10987654;6;3.9

1. **alojamientos.txt**

codigo;nombre;docAnfitrion;departamento;municipio;tipo;direccion;precio;amenidades

ALOJ001;CasaPaisa;12345678;Antioquia;Medellín;casa;Cra 45 # 10-23;120000;1,0,1,1,1,0

1. **reservas.txt**

codigoReserva;codAlojamiento;docHuesped;fechaEntrada;duracion;metodoPago;fechaPago;monto;anotacion

RSV001;ALOJ001;10987654;14/05/2025;3;PSE;10/05/2025;360000;¿Tiene aire acondicionado?

1. **historico\_reservas.txt**
   * Igual estructura que reservas.txt, pero contiene solo las reservas vencidas.

**4. Relaciones entre Clases**

| **Entidad** | **Asociación principal** |
| --- | --- |
| Anfitrion | Tiene 1 a N Alojamiento |
| Alojamiento | Tiene 1 Anfitrion y 0 a N Reserva |
| Huesped | Tiene 0 a N Reserva |
| Reserva | Apunta a 1 Alojamiento y 1 Huesped |

**5. Decisiones Clave de Eficiencia**

* La clase Reserva **no estará contenida en Alojamiento**, sino en un **arreglo global de reservas**.
* Se utilizarán funciones auxiliares para medir:
  + **Iteraciones** realizadas durante procesos clave (búsquedas, validaciones).
  + **Uso de memoria** total, sumando los sizeof de los objetos dinámicos vivos.

**6. Propuesta Inicial de Clases (resumen)**

* class Anfitrion
  + string documento; int antiguedad; float puntuacion;
  + VectorDinamico<Alojamiento\*> alojamientos;
* class Huesped
  + string documento; int antiguedad; float puntuacion;
* class Alojamiento
  + string codigo; string nombre; Anfitrion\* propietario;
  + bool reservasPorDia[365];
* class Reserva
  + string codigo; Alojamiento\* alojamiento; Huesped\* huesped; Fecha fechaEntrada;
* class Fecha
  + int dia, mes, anio;

**7. Siguientes pasos**

* Implementación de las clases base (Fecha, Reserva, Alojamiento, etc.).
* Construcción del cargador de datos desde archivos.
* Desarrollo progresivo del menú de usuario y funcionalidades específicas.
* Medición de recursos e iteraciones como parte del motor central del sistema.