Universidad Rafael Landívar
Tradición Jesuita en Guatemala

Facultad de Ingeniería
Estructura de Datos II

Proyecto 1 – Simulación de atención en emergencias

Estructuras de datos a utilizar: Árbol B+, Montículo, Tabla Hash

El objetivo de este proyecto es aplicar estructuras de datos avanzadas para simular el funcionamiento básico del área de atención en emergencias de un hospital, utilizando estructuras eficientes para almacenamiento, búsqueda y ordenamiento de pacientes.

Descripción del proyecto

Se desarrollará un sistema que simule el funcionamiento de un hospital con enfoque en la atención de emergencias. Para ello tome en cuenta lo siguiente:

1. Los pacientes se almacenarán en un Árbol B+, indexados por su número de identificación (ID), los datos a almacenar son: primer nombre, segundo nombre, primer apellido, segundo apellido, fecha de nacimiento y correo electrónico.
2. Se mantendrá una tabla hash sincronizada con el árbol B+ que guardará únicamente primer nombre, primer apellido e ID de cada paciente, permitiendo búsqueda por nombre y apellido para obtener el ID.
3. Se simulará un día de atención en emergencias, donde los pacientes llegarán al hospital indicando su ID o su primer nombre y apellido, junto a un nivel de prioridad.
4. Al ser encontrados, su ID y prioridad se guardan en un montículo (heap máximo) para ser atendidos de mayor a menor prioridad.
5. El sistema procesará un archivo de entrada con múltiples secciones que pueden aparecer en cualquier orden y repetirse.

Formato del archivo de entrada (ejemplo)

El archivo puede contener secciones con los siguientes encabezados:

```
CREAR PACIENTES

1234567, Ana, Lucía, Gómez, Pérez, 2001-05-18, ana.gomez@mail.com
1234568, Juan, Carlos, Méndez, López, 1999-11-02, juan.mendez@mail.com

ATENCION EMERGENCIA

ID: 1234567, Prioridad: 9
NOMBRES: Juan, Méndez, Prioridad: 10

BORRAR PACIENTES

1234567
1234568
```

Comportamiento esperado del sistema

- Cada sección será procesada de inmediato al ser leída.
- Si se intenta insertar un paciente con un ID ya existente, debe mostrarse un mensaje de error.
- Al atender en emergencia:
    - Si el paciente indica su ID, se busca directamente en el árbol B+.
    - Si indica solo su primer nombre y apellido
        - Se busca en la tabla hash para obtener el ID.
        - Si existen múltiples coincidencias, se asume que el paciente correcto es el más joven (asuma que no existirán dos pacientes con los tres datos iguales)
    - Si el paciente no se encuentra, se rechaza con un mensaje claro (esto es una simulación; en un sistema real no se rechazaría).
    - Los pacientes encontrados se insertan en un montículo máximo con su nivel de prioridad.
    - Se atenderán en orden de mayor a menor prioridad, por cada extracción se debe mostrar a quién se atendió (todos los datos).
- Toda operación (creación, borrado, atención) debe escribirse en dos archivos de salida:

Archivo	Contenido
registro_operaciones.txt	Altas y bajas de pacientes, con éxito o fallo
registro_emergencias.txt	Atención por prioridad, con éxito o rechazo

- Al finalizar el procesamiento del archivo, se preguntará al usuario si desea leer otro archivo o salir.
- Al salir, el sistema debe guardar el estado del árbol B+ y de la tabla hash en archivos legibles por el usuario (por ejemplo: pacientes_final.txt y hash_nombres.txt).
- Al iniciar el programa, debe cargar estos archivos directamente en memoria (sin insertar elemento por elemento).

Ejemplo de salida

registro_operaciones.txt

Paciente creado: id 1234567, valor hash: 5
Paciente creado: id 1234568, valor hash: 10
Paciente eliminado: id 1234567
Paciente eliminado: id 1234568

registro_emergencias.txt

Llegada de paciente: ID: 1234567, Prioridad: 9 ***Encontrado
Llegada de paciente: NOMBRES: Juan, Méndez, Prioridad: 10 ***Encontrado ID 12345678
Llegada de paciente: NOMBRES: Pedro, Méndez, Prioridad: 15 ***Rechazado
Llegada de paciente: NOMBRES: Juan, Pérez, Prioridad: 5 *** 3 pacientes encontrados, se atiende a ID 1234599

Requisitos técnicos

- El árbol B+ debe implementarse desde cero y debe ser de grado configurable (queda a discreción del estudiante cómo se configura).
- La tabla hash debe manejar colisiones.
- El montículo debe ser un heap máximo y debe permitir inserción dinámica.
- Todas las estructuras deben estar sincronizadas al insertar o eliminar pacientes.
- El sistema debe ser modular, limpio, ejecutable por consola y **entregar un ejecutable**.
- El sistema debe manejar múltiples ejecuciones: cargar estructuras, procesar archivo, mostrar resultados, guardar estado.

Rúbrica

Criterio	Puntos
Árbol B+ funcional y correcto	12
Tabla hash funcional y sincronizada con el árbol	12
Montículo funcional y correcto	12
Lectura y procesamiento del archivo de entrada	12
Manejo de errores (ID duplicado, paciente inexistente)	12
Escritura de archivos de salida	15
Persistencia (cargar/guardar estructuras)	15
Estilo, orden, etc.	10
Total	100

Fecha de entrega

Entrega: miércoles 20 de agosto de 2025
Calificación: jueves 21 de agosto de 2025 en clase