Informe de temperaturas- Simulación de Sala de Emergencias

UNIVERSIDAD NACIONAL DE ENTRE RÍOS

Facultad de Ingeniería

Carrera: T.U.P.E.D

Materia: Fundamentos de Algoritmos y Estructuras de Datos

Estudiante: Franco Tomiozzo

Profesor: Dr. Javier E. Diaz Zamboni

Bioing. Jordán F. Insfrán

Esp. Bioing. Francisco Rizzato

Comisión: 1

Fecha: Octubre 2025

Introducción

En el presente trabajo desarrollé una simulación de una **sala de emergencias** cuyo propósito fue modelar la atención de pacientes según su **nivel de riesgo clínico**. El objetivo principal consistió en implementar una estructura de datos que garantice que **el paciente más urgente sea siempre atendido primero**, gestionando adecuadamente los flujos de ingreso y atención dentro del sistema.

Desarrollo y estructura seleccionada

Para resolver el problema seleccioné una **cola de prioridad implementada mediante un heap (montículo binario)**, utilizando el módulo estándar heapq de Python. La estructura creada —contenida en el archivo ColaPrioridad— es **genérica**, es decir, puede almacenar cualquier tipo de objeto, no solo pacientes. Esto se logró implementando las inserciones como tuplas de la forma (prioridad, contador, dato).

- **Prioridad:** representa el nivel de riesgo clínico del paciente (1 crítico, 2 moderado, 3 bajo).
- **Contador:** se utiliza como segundo criterio de orden (FIFO) para desempatar entre pacientes con igual nivel de riesgo.
- **Dato:** almacena el objeto completo del paciente.

De esta manera, garantizar que el heap mantenga siempre en el tope al paciente con **menor valor de prioridad**, es decir, **el caso más urgente**. El uso del contador adicional asegura la estabilidad de la cola, respetando el orden de llegada ante empates.

Complejidad y justificación técnica

Elegí un **min-heap** porque permite un manejo eficiente de las operaciones principales:

- **Inserción (heappush)**: O(logn)O(\log n)O(logn)
- **Extracción del más prioritario (heappop)**: O(logn)O(\log n)O(logn)
- **Consulta del siguiente sin extraer (peek)**: O(1)O(1)O(1)

Estas características hacen que el heap sea ideal para un sistema dinámico donde ingresan y se atienden pacientes de forma continua. Además, al ser una estructura basada en lista, no requiere espacio adicional más allá del almacenamiento de los elementos.

Resultados de la simulación

Durante la ejecución del archivo **accion_de_pacientes.py** se observó que **los pacientes eran atendidos según su prioridad médica**, cumpliendo con el propósito del sistema. Por ejemplo, cuando ingresó un paciente con riesgo crítico-1 (ej: Antonio Belgrano), fue atendido

inmediatamente en el siguiente ciclo, sin importar cuántos pacientes de riesgo moderado o bajo se encontraban en espera. Esto evidencia que la cola administra correctamente las prioridades y respeta las reglas de triaje.

Conclusión

Considero que el uso del **heap como cola de prioridad genérica** fue la elección más adecuada para este escenario. Permite manejar el flujo de pacientes de manera eficiente, estable y justa, priorizando siempre los casos más urgentes.

En futuras mejoras podría añadirse una interfaz gráfica o un registro histórico de atención para ampliar el análisis de desempeño del sistema.