OBSERVACIONES DE LA PRÁCTICA

Bastien Quentin Clement 202525085
Jonathan David Galeano Sosa 202226332

a) ¿Qué estructura de datos subyacente se utiliza comúnmente para implementar una cola de prioridad en Python?

RTA: La estructura más común para implementar una cola de prioridad en Python es el heap, que organiza los datos según su prioridad.

b) ¿Cuál es la diferencia entre una cola FIFO tradicional y una cola de prioridad?

RTA: En una cola FIFO los elementos salen en el mismo orden en que entran. En cambio, en la cola de prioridad los elementos salen según su nivel de prioridad en vez de su orden de llegada.

c) ¿Qué módulo proporciona Python para trabajar fácilmente con colas de prioridad?

RTA: Python tiene el módulo `heapq`, que permite crear y manejar colas de prioridad fácilmente.

d) ¿Qué ventajas tiene el uso de una cola de prioridad sobre una lista ordenada manualmente?

RTA: Usar una cola de prioridad es mejor que ordenar una lista manualmente porque el programa se ejecuta más rápido y no hay que ordenar todo cada vez que se agrega un nuevo dato.

e) Si dos elementos tienen la misma prioridad, ¿cómo decide la cola cuál atender primero?

RTA: Si dos elementos tienen la misma prioridad, la cola puede atender cualquiera de los dos primero, ya que ambos tienen el mismo nivel de importancia.

f) ¿Qué situaciones del mundo real se pueden modelar con colas de prioridad? Mencione al menos dos.

RTA: Pueden ser la cola de prioridad de un centro medico donde se atienden a los mas graves antes, o tambien un gestor de tareas que priorice las de mayor demanda.

g) En un sistema de atención médica, ¿cómo se puede usar una cola de prioridad para organizar a los pacientes?

RTA: Se puede usar una cola de prioridad para que los pacientes más graves sean atendidos primero, sin importar el orden en que llegaron.

h) ¿Cómo afectaría al rendimiento usar una lista simple en lugar de una estructura especializada como *heap_pq* para manejar prioridades?

RTA: Si se usa una lista común en lugar de una estructura como *heap*, el programa sería más lento porque tendría que buscar y ordenar manualmente las prioridades en cada iteración.

i) ¿Qué complejidad temporal tiene la inserción y eliminación/atención en una cola de prioridad basada en *heap*?

RTA: Insertar o eliminar un elemento tiene una complejidad de aproximadamente O(log n) lo que la hace eficiente incluso con muchos datos.