Juan David Forero Huerfano, jd.foreroh1@uniandes.edu.co, 202411707.
David Felipe Mendoza Carrillo, d.mendozacarrillo@uniandes.edu.co, 202421689.
Juan José Hernández Vera, jj.hernandezve1@uniandes.edu.co, 202423465.

Preguntas laboratorio 10

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

La estructura de datos en la cual se fundamenta el funcionamiento de una cola de prioridad en python es el heap, que es un árbol binario que cumple las propiedades del min heap (el valor del padre es menor al de sus hijos y el elemento de mínimo valor está en la raíz) o maxheap (el valor del padre es mayor al valor de sus hijos y el elemento de máximo valor está en la raíz).

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

La principal diferencia es que al no existir una prioridad en los elementos de la cola FIFO, el primer elemento que llega a la cola es el primer elemento que sale o que es entregado a un usuario, tal como funciona una cola en la vida real. Por otro lado, como los elementos de una cola de prioridad tienen predominancia sobre otros elementos de la misma cola, son estos elementos que tienen predominancia los que son entregados primero al usuario. Además, la organización de la cola en las colas de prioridad está dada por su predominancia, no por el orden de llegada.

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

El módulo que proporciona python para el trabajo fácil con colas de prioridad es el módulo queue, específicamente el módulo de heapq que se encarga de la verificación de heaps para la construcción de colas de prioridad.

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

Las principales ventajas que existen están dadas por la complejidad temporal y espacial de cada algoritmo. Además, al ser un arbol binario que sigue las propiedades del minheap o el max heap, la actualización de los elementos es automática y su prioridad está dada por su organización en el árbol (todo realizado de forma recursiva). Por otro lado, siempre que se accede a cualquier elemento y se elimina, es necesario una reorganización de prioridades que genera una pérdida de eficiencia y un aumento en tiempos de ejecución.

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

Si dos elementos tienen la misma prioridad, el funcionamiento de la entrega al usuario está dado por su orden de llegada al heap. Este comportamiento es propio de las colas FIFO, y esto se debe principalmente a que en una misma prioridad, se da primero el que está más arriba del árbol (elemento que primero llegó a la cola de prioridad).

f) ¿Qué se debe hacer para que los elementos personalizados puedan ser almacenados en una cola de prioridad en Python?

Es necesario que los elementos tengan algún atributo que permita que sean ordenados basados en este. Este atributo interno puede ser numérico o alfabético, pero debe ser compatible con el sort criteria usado en el Heap.

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

La gestión de emergencias sanitarias en una clínica puede ser manejada como una cola de prioridad, donde ciertos pacientes tienen mayor prioridad que otros independientemente del orden en el que hayan llegado o entrado al sistema. Estos son organizados basado en un criterio de gravedad para ser atendidos, y salir de la línea de espera en ese orden. Otro ejemplo podría ser la logística para el control de tráfico aéreo, donde las autorizaciones para el despegue salen de acuerdo a la prioridad del vuelo, y no necesariamente en el orden en el que llegaron a la terminal aérea.

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

Se pueden organizar a los pacientes de acuerdo al tipo de persona que necesita atención (niño/adulto/adulto mayor) y asimismo se puede priorizar de acuerdo al motivo por el que solicita la atención. Si un adulto mayor o un niño solicita un servicio de por una urgencia de alto nivel, será atendido con mayor prioridad que un adulto que viene por citas de control anuales.

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

Como fue explicado en el literal d), las complejidades para la inserción (llegada a la cola) y la extracción (salida de la cola), son mucho mejores en un heap que en una lista.En la inserción y extracción, mientras que en una cola de prioridad es logn (al ser un árbol binario), en una lista tiende a O (n), además de la reorganización necesaria (por la prioridad) de todos los elementos de la lista. Además, en caso de inserción, no es necesario reordenar los elementos del heap, a diferencia de la lista. Esto, por las funciones recursivas de un árbol, permite el ordenamiento automático al alterar el árbol.

j) ¿Qué complejidad tiene la inserción y extracción en una cola de prioridad basada en heap?

-Inserción = O (logn)
-Extracción O (logn)

Al ser un árbol binario, el acceso al elemento que va a ser eliminado se da por el valor de la llave (si es menor o mayor a su padre dependiendo el criterio de comparación) o que convierte a la mitad las posibilidades del elemento. Al acceder a él, se elimina el elemento y recursivamente se ocupa su lugar y se reordena el árbol dependiendo las prioridades y su acceso. Similar a esto, la inserción de estos elementos es logarítmica al buscar recursivamente la posición donde debe ser insertado.