Test cases

**Stack**

La implementación fue basada en: <https://www.geeksforgeeks.org/python/stack-in-python/>

Implemente un stack desde cero usando nodos y linked lists como la base, me base en un código de geeks for geeks para implementarlo. Mis casos de uso fueron insertar 10,20 y 30, luego hacer pop, luego peek, imprimir el tamaño del stack y revisar si está vacío. Luego fue vaciar el stack con pop, intentar hacer pop otra vez cuando este vacío, luego hacer peek, size y finalmente revisar si este vacío.

**Queue**

La implementación fue basada en: <https://www.w3schools.com/python/python_dsa_queues.asp>

Hice una implementación de cero con linked lists, basándome en el código de w3schools, usando como casos de prueba llenarlo con strings, luego hacer un dequeu, luego peek, size, isempty y finalmente vaciarlo y hacer isempty, peek y size con el queue vacio para revisar que el código funciona correctamente.

**Hash**

La implementación fue basada en: <https://manishankarjaiswal.medium.com/understanding-hashing-and-hash-tables-in-python-c6336447073d>

Genere una implementación con listas de Python como la base, usando la función de hasheo de Python y un manejo de colisiones con chaining para que crezca la lista dentro del mismo índice. Para probarlo genere un hash table con tamaño de 5, inserte 6 elementos para forzar una colisión al menos, luego un display para ver cómo se generó y un search para buscar un elemento por su llave. Posteriormente un delete y finalmente una actualización del valor cuando tenga exactamente la misma llave y un print para revisar que todo funciono correctamente.