lru_cache_explanation

February 21, 2022

0.0.1 LRU-Cache data structure

Design Considerations This problem's objective was to design a least recently used cache, where the least recently used entry is removed in order to create room when it's capacity is maxed out. Specifically I was to design the get method when given a key and set method that inserts a key-value data pair to the cache. I decided to use a dictionary since insertion and accessing takes constant time 0(1).

Time and Space analysis For the `get(key)` method the worst case time notation is constant i.e $O(1)$. While, for the `set(key,value)` insertion for a new key and without exceeding the capacity is constant too therefore $O(1)$. However, insertion when memory is exceeded a for loop for the whole dictionar is utilised leading to a worst case of $O(n)$.

As for the space analysis two dictionaries are used one for the actual key-value pair storage of `n` items while the second is for storing number of times a key is accesed hence also contains `n` items. Therefore the space complexity can be represented as $O(n) + O(n)$ which is $O(n)$.