Least Recently Used (LRU) Cache

A Least Recently Used (LRU) Cache organizes items in order of use, allowing you to quickly identify which item hasn't been used for the longest amount of time.
We are given a cache size, when cache is to insert new element (NOT exist in cache), it removes the least recently used element if cache is full.

Design and implement a data structure for Least Recently Used (LRU) cache. It should support the following operations: **get**, **put** and **remove** methods. **Average time complexity for all 3 methods should be O(1).**

1. get(key) - Get the value of the key if the key exists in the cache, otherwise return null.
2. put(key, value) - Set or insert the value if the key is not already present. When the cache reached its capacity, it should invalidate the least recently used item before inserting a new item.
3. remove(key) – remove the key if the key exists in the cache.

The cache is initialized with a positive capacity.

Hint:

Use doubly linked list and hash map.