Problem 1 – LRU Cache

As I have started with Udacity Data Structures and Algorithms course I have also taken subscription at algoexpert.io – to elevate the gains from the Udacity course. For this problem I have implemented the LRU Cache procedure taught by algoexpert.io. This way I would like to avoid any plagiarism.

We would like to implement insertion (`set()`) and retrieval (`get()`) functions in constant time, therefore we need to use hash table. For this purpose, our cache will be a Python dictionary. For the structure we will use Doubly Linked List.

This way, insertion to cache takes constant time O(1) and also, O(1) space. Getting a value given the key takes O(1) time complexity and O(1) space complexity. As these tasks takes constant amount of elementary operations. Hence, these tasks takes O(1) time complexity, because ours cache is a dictionary.

These tasks takes O(1) space complexity, because linked list are spread throughout whole memory, and the value can be accessed easily given the key. This way, there is no need to loop through whole linked list.