AlgoExpert Quad Layout Python 12px Sublime Monokai 00:00:00

Prompt Scratchpad Our Solution(s) Video Explanation Run Code

```
Solution 1
 {\tt 1}{\tt} # Copyright @ 2020 AlgoExpert, LLC. All rights reserved.
    class LRUCache:
        def __init__(self, maxSize):
            self.cache = {}
            self.maxSize = maxSize or 1
            self.currentSize = 0
             self.listOfMostRecent = DoublyLinkedList()
10
        \# O(1) time \mid O(1) space
        def insertKeyValuePair(self, key, value):
11
12
            if key not in self.cache:
13
                if self.currentSize == self.maxSize:
14
                    self.evictLeastRecent()
15
16
                     self.currentSize += 1
                self.cache[key] = DoublyLinkedListNode(key, value)
17
18
19
                self.replaceKey(key, value)
20
            self.updateMostRecent(self.cache[key])
21
        # 0(1) time | 0(1) space
22
23
        def getValueFromKey(self, key):
24
             if key not in self.cache:
25
                return None
26
             self.updateMostRecent(self.cache[key])
27
             \textbf{return self.cache} [\texttt{key}]. \textbf{value}
28
29
        \# O(1) time | O(1) space
30
        def getMostRecentKey(self):
31
             return self.listOfMostRecent.head.key
32
        def evictLeastRecent(self):
33
            keyToRemove = self.listOfMostRecent.tail.key
34
             self.listOfMostRecent.removeTail()
35
36
             del self.cache[keyToRemove]
37
38
        def updateMostRecent(self, node):
39
            self.listOfMostRecent.setHeadTo(node)
40
41
        def replaceKey(self, key, value):
42
            if key not in self.cache:
43
                raise Exception("The provided key isn't in the cache!")
44
             self.cache[key].value = value
45
46
47
    class DoublyLinkedList:
48
        def __init__(self):
49
            self.head = None
50
             self.tail = None
51
52
        def setHeadTo(self, node):
53
            if self.head == node:
54
                return
             elif self.head is None:
55
56
                self.head = node
57
                 self.tail = node
58
            elif self.head == self.tail:
59
                self.tail.prev = node
60
                self.head = node
                self.head.next = self.tail
61
62
63
                if self.tail == node:
64
                    self.removeTail()
                node.removeBindings()
65
                self.head.prev = node
66
                node.next = self.head
67
68
                self.head = node
69
70
        def removeTail(self):
            if self.tail is None:
71
72
                return
73
            if self.tail == self.head:
                self.head = None
74
75
                self.tail = None
76
                return
77
             self.tail = self.tail.prev
78
            self.tail.next = None
79
80
    class DoublyLinkedListNode:
81
82
        def __init__(self, key, value):
83
             self.key = key
             self.value = value
85
            self.prev = None
86
            self.next = None
87
        def removeBindings(self):
88
89
            if self.prev is not None:
                self.prev.next = self.next
90
            if self.next is not None:
92
                self.next.prev = self.prev
93
            self.prev = None
            self.next = None
94
```

95