6

LFU Cache

Try to solve the LFU Cache problem.

We'll cover the following

- Statement
- Examples
- · Understand the problem
- Figure it out!
- Try it yourself

Statement

Design and implement a data structure for a Least Frequently Used (LFU) cache.

Implement the **LFUCache** class. Here is how it should be implemented:

- LFUCache(capacity): This function initializes the object with the capacity of the data structure.
- **get(key):** This function gets the value of the **key** if it exists in the cache. Otherwise, it returns -1.
- **put(key, value):** This function updates the value of the **key** if present, or inserts the **key** if it's not present. When the cache reaches its **capacity**, it should invalidate and remove the least frequently used key before inserting a new item. For this problem, when there's a tie, that is, two or more keys have the same frequency, the least recently used **key** is invalidated.

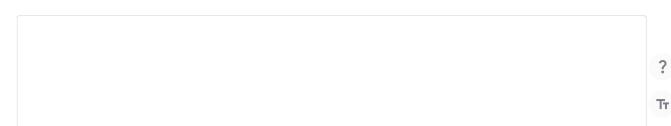
To determine the least frequently used key, a use counter is maintained for each key in the cache. The key with the smallest use counter is the least frequently used key. When a key is first inserted into the cache, its use counter is set to 1 (due to the **put()** operation). The use counter for a key in the cache is incremented and either a **get()** or **put()** operation is called on it.

The **get()** and **put()** functions should both run with an average time complexity of O(1).

Constraints:

- $0 \le \text{capacity} \le 10^4$
- $0 \le \text{key} \le 10^5$
- $0 \le \text{value} \le 10^9$
- At most 2×10^5 calls will be made to **get()** and **put()**.

Examples



1 of 8
2 of 8
3 of 8

C

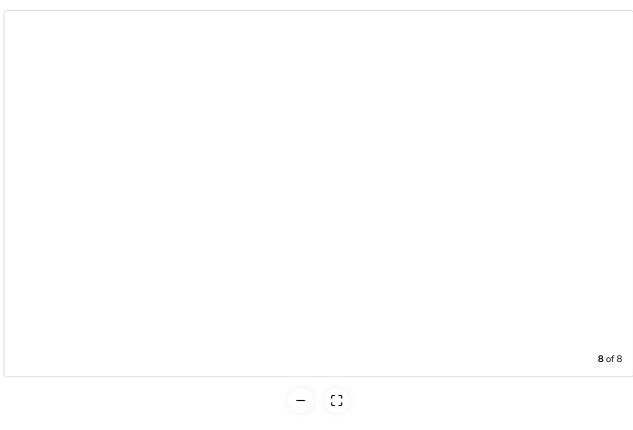
4 of 8	
	4 - 4 0
5 of 8	4 01 8
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
5 of 8	
	5 of 8

0.10
6 of 8
7 of 8

?

Тт

6



Understand the problem

Let's take a moment to make sure you've correctly understood the problem. The quiz below helps you check if you're solving the correct problem:

```
LFU Cache

What is the output if the cache size is 2 and the following keys and values are given as input?

put →[1, 1]

put →[2, 2]

get → [2]

put →[3, 3]

get → [1]

get → [3]

put →[4, 4]

put →[5, 5]

get → [1]

get → [3]

get → [5]

A) [NULL, NULL, 2, NULL, 1, 3, NULL, NULL, -1, 3, 5]
```

?

 T_T

6

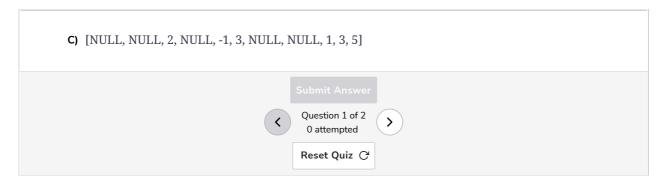
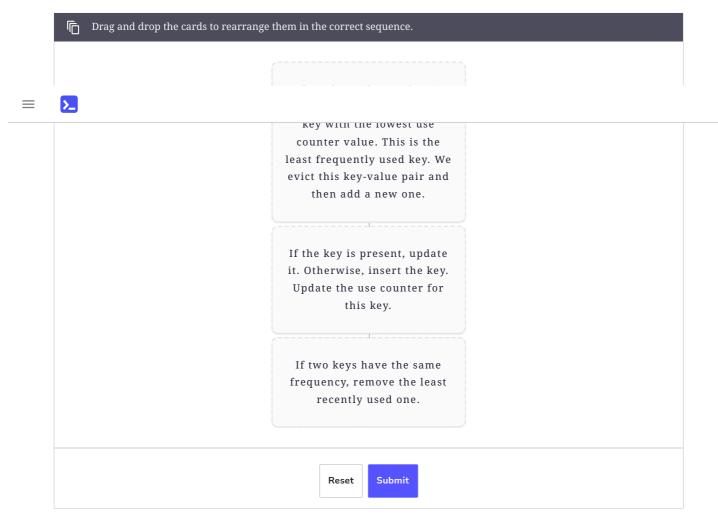


Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Note: As an additional challenge, we have intentionally hidden the solution to this puzzle.



Try it yourself

Implement your solution in LFUCache. java in the following coding playground.

Note: We have left the solution to this challenge as an exercise for you. You may try to translate the logic of the solved puzzle into a coded solution.

To Solved Puzzle into a coded solution.

LFUCache.java

LinkedListNode.java

LinkedList.java

```
1 import java.util.*;
2 import java.util.*;
3
4 class LFUCache {
5     // Constructor that sets the size of the cache
6     public LFUCache(int size) {
```

.

Ττ

C