Problem 379: Design Phone Directory

Problem Information

Difficulty: Medium
Acceptance Rate: 52.65%
Paid Only: Yes
Tags: Array, Hash Table, Linked List, Design, Queue

Problem Description

Design a phone directory that initially has `maxNumbers` empty slots that can store numbers. The directory should store numbers, check if a certain slot is empty or not, and empty a given slot.

Implement the `PhoneDirectory` class:

* `PhoneDirectory(int maxNumbers)` Initializes the phone directory with the number of available slots `maxNumbers`. * `int get()` Provides a number that is not assigned to anyone. Returns `-1` if no number is available. * `bool check(int number)` Returns `true` if the slot `number` is available and `false` otherwise. * `void release(int number)` Recycles or releases the slot `number`.

Example 1:

Input ["PhoneDirectory", "get", "get", "check", "get", "check", "release", "check"] [[3], [], [], [2], [], [2], [2], [2]] **Output** [null, 0, 1, true, 2, false, null, true] **Explanation** PhoneDirectory phoneDirectory = new PhoneDirectory(3); phoneDirectory.get(); // It can return any available phone number. Here we assume it returns 0. phoneDirectory.get(); // Assume it returns 1. phoneDirectory.check(2); // The number 2 is available, so return true. phoneDirectory.get(); // It returns 2, the only number that is left. phoneDirectory.check(2); // The number 2 is no longer available, so return false. phoneDirectory.release(2); // Release number 2 back to the pool. phoneDirectory.check(2); // Number 2 is available again, return true.

Constraints:

* `1 <= maxNumbers <= 104` * `0 <= number < maxNumbers` * At most `2 * 104` calls will be made to `get`, `check`, and `release`.

Code Snippets

C++:

```
class PhoneDirectory {
public:
PhoneDirectory(int maxNumbers) {

}

int get() {

}

bool check(int number) {

}

void release(int number) {

}
};

/**
* Your PhoneDirectory object will be instantiated and called as such:
* PhoneDirectory* obj = new PhoneDirectory(maxNumbers);
* int param_1 = obj->get();
* bool param_2 = obj->check(number);
* obj->release(number);
*/
```

Java:

```
class PhoneDirectory {

public PhoneDirectory(int maxNumbers) {

}
public int get() {

}

public boolean check(int number) {

}

public void release(int number) {

}
}

/**
* Your PhoneDirectory object will be instantiated and called as such:
* PhoneDirectory obj = new PhoneDirectory(maxNumbers);
* int param_1 = obj.get();
* boolean param_2 = obj.check(number);
* obj.release(number);
*/
```

Python3:

```
class PhoneDirectory:

def __init__(self, maxNumbers: int):


def get(self) -> int:


def check(self, number: int) -> bool:


def release(self, number: int) -> None:



# Your PhoneDirectory object will be instantiated and called as such:
# obj = PhoneDirectory(maxNumbers)
# param_1 = obj.get()
# param_2 = obj.check(number)
# obj.release(number)
```