Problem 2526: Find Consecutive Integers from a Data Stream

Problem Information

Difficulty: Medium
Acceptance Rate: 50.37%
Paid Only: No
Tags: Hash Table, Design, Queue, Counting, Data Stream

Problem Description

For a stream of integers, implement a data structure that checks if the last `k` integers parsed in the stream are **equal** to `value`.

Implement the **DataStream** class:

* `DataStream(int value, int k)` Initializes the object with an empty integer stream and the two integers `value` and `k`. * `boolean consec(int num)` Adds `num` to the stream of integers. Returns `true` if the last `k` integers are equal to `value`, and `false` otherwise. If there are less than `k` integers, the condition does not hold true, so returns `false`.

Example 1:

Input ["DataStream", "consec", "consec", "consec", "consec"] [[4, 3], [4], [4], [4], [3]] **Output** [null, false, false, true, false] **Explanation** DataStream dataStream = new DataStream(4, 3); //value = 4, k = 3 dataStream.consec(4); // Only 1 integer is parsed, so returns False. dataStream.consec(4); // Only 2 integers are parsed. // Since 2 is less than k, returns False. dataStream.consec(4); // The 3 integers parsed are all equal to value, so returns True. dataStream.consec(3); // The last k integers parsed in the stream are [4,4,3]. // Since 3 is not equal to value, it returns False.

Constraints:

* `1 <= value, num <= 109` * `1 <= k <= 105` * At most `105` calls will be made to `consec`.

Code Snippets

C++:

```
class DataStream {
public:
DataStream(int value, int k) {

}

bool consec(int num) {

}
};

/**
* Your DataStream object will be instantiated and called as such:
* DataStream* obj = new DataStream(value, k);
* bool param_1 = obj->consec(num);
*/
```

Java:

```
class DataStream {

public DataStream(int value, int k) {

}

public boolean consec(int num) {

}
}

/**
* Your DataStream object will be instantiated and called as such:
* DataStream obj = new DataStream(value, k);
* boolean param_1 = obj.consec(num);
*/
```

Python3:

```
class DataStream:

def __init__(self, value: int, k: int):


def consec(self, num: int) -> bool:



# Your DataStream object will be instantiated and called as such:
# obj = DataStream(value, k)
# param_1 = obj.consec(num)
```