Problem 1286: Iterator for Combination

Problem Information

Difficulty: Medium
Acceptance Rate: 72.58%
Paid Only: No
Tags: String, Backtracking, Design, Iterator

Problem Description

Design the `CombinationIterator` class:

* `CombinationIterator(string characters, int combinationLength)` Initializes the object with a string `characters` of **sorted distinct** lowercase English letters and a number `combinationLength` as arguments. * `next()` Returns the next combination of length `combinationLength` in **lexicographical order**. * `hasNext()` Returns `true` if and only if there exists a next combination.

Example 1:

Input ["CombinationIterator", "next", "hasNext", "next", "hasNext", "next", "hasNext"] [["abc", 2], [], [], [], [], [], []] **Output** [null, "ab", true, "ac", true, "bc", false] **Explanation** CombinationIterator itr = new CombinationIterator("abc", 2); itr.next(); // return "ab" itr.hasNext(); // return True itr.next(); // return "ac" itr.hasNext(); // return True itr.next(); // return "bc" itr.hasNext(); // return False

Constraints:

* `1 <= combinationLength <= characters.length <= 15` * All the characters of `characters` are **unique**. * At most `104` calls will be made to `next` and `hasNext`. * It is guaranteed that all calls of the function `next` are valid.

Code Snippets

C++:

```
class CombinationIterator {
public:
CombinationIterator(string characters, int combinationLength) {

}

string next() {

}

bool hasNext() {

}
};

/**
* Your CombinationIterator object will be instantiated and called as such:
* CombinationIterator* obj = new CombinationIterator(characters,
combinationLength);
* string param_1 = obj->next();
* bool param_2 = obj->hasNext();
*/
```

Java:

```
class CombinationIterator {

public CombinationIterator(String characters, int combinationLength) {

}

public String next() {

}

public boolean hasNext() {

}
}

/**
* Your CombinationIterator object will be instantiated and called as such:
* CombinationIterator obj = new CombinationIterator(characters,
```

```
combinationLength);
* String param_1 = obj.next();
* boolean param_2 = obj.hasNext();
*/
```

Python3:

```
class CombinationIterator:

def __init__(self, characters: str, combinationLength: int):


def next(self) -> str:


def hasNext(self) -> bool:



# Your CombinationIterator object will be instantiated and called as such:
# obj = CombinationIterator(characters, combinationLength)
# param_1 = obj.next()
# param_2 = obj.hasNext()
```