Design and implement a data structure for a compressed string iterator. The given compressed string will be in the form of each letter followed by a positive integer representing the number of this letter existing in the original uncompressed string.

Implement the StringIterator class:

* next() Returns **the next character** if the original string still has uncompressed characters, otherwise returns a **white space**.
* hasNext() Returns true if there is any letter needs to be uncompressed in the original string, otherwise returns false.

**Example 1:**

**Input**

["StringIterator", "next", "next", "next", "next", "next", "next", "hasNext", "next", "hasNext"]

[["L1e2t1C1o1d1e1"], [], [], [], [], [], [], [], [], []]

**Output**

[null, "L", "e", "e", "t", "C", "o", true, "d", true]

**Explanation**

StringIterator stringIterator = new StringIterator("L1e2t1C1o1d1e1");

stringIterator.next(); // return "L"

stringIterator.next(); // return "e"

stringIterator.next(); // return "e"

stringIterator.next(); // return "t"

stringIterator.next(); // return "C"

stringIterator.next(); // return "o"

stringIterator.hasNext(); // return True

stringIterator.next(); // return "d"

stringIterator.hasNext(); // return True

**Constraints:**

* 1 <= compressedString.length <= 1000
* compressedString consists of lower-case an upper-case English letters and digits.
* The number of a single character repetitions in compressedString is in the range [1, 10^9]
* At most 100 calls will be made to next and hasNext.