Problem 1117: Building H2O

Problem Information

Difficulty: Medium
Acceptance Rate: 58.22%
Paid Only: No
Tags: Concurrency

Problem Description

There are two kinds of threads: `oxygen` and `hydrogen`. Your goal is to group these threads to form water molecules.

There is a barrier where each thread has to wait until a complete molecule can be formed. Hydrogen and oxygen threads will be given `releaseHydrogen` and `releaseOxygen` methods respectively, which will allow them to pass the barrier. These threads should pass the barrier in groups of three, and they must immediately bond with each other to form a water molecule. You must guarantee that all the threads from one molecule bond before any other threads from the next molecule do.

In other words:

* If an oxygen thread arrives at the barrier when no hydrogen threads are present, it must wait for two hydrogen threads. * If a hydrogen thread arrives at the barrier when no other threads are present, it must wait for an oxygen thread and another hydrogen thread.

We do not have to worry about matching the threads up explicitly; the threads do not necessarily know which other threads they are paired up with. The key is that threads pass the barriers in complete sets; thus, if we examine the sequence of threads that bind and divide them into groups of three, each group should contain one oxygen and two hydrogen threads.

Write synchronization code for oxygen and hydrogen molecules that enforces these constraints.

Example 1:

Input: water = "HOH" **Output:** "HHO" **Explanation:** "HOH" and "OHH" are also valid answers.

Example 2:

Input: water = "OOHHHH" **Output:** "HHOHHO" **Explanation:** "HOHHHO", "OHHHHO", "HHOHOH", "HOHHOH", "OHHHOH", "HHOOHH", "HOHOHH" and "OHHOHH" are also valid answers.

Constraints:

* `3 * n == water.length` * `1 <= n <= 20` * `water[i]` is either `'H'` or `'O'`. * There will be exactly `2 * n` `'H'` in `water`. * There will be exactly `n` `'O'` in `water`.

Code Snippets

C++:

```
class H2O {
public:
H2O() {

}

void hydrogen(function<void()> releaseHydrogen) {

// releaseHydrogen() outputs "H". Do not change or remove this line.
releaseHydrogen();
}

void oxygen(function<void()> releaseOxygen) {

// releaseOxygen() outputs "O". Do not change or remove this line.
releaseOxygen();
}
};
```

Java:

```
class H2O {

public H2O() {

}

public void hydrogen(Runnable releaseHydrogen) throws InterruptedException {

// releaseHydrogen.run() outputs "H". Do not change or remove this line.
releaseHydrogen.run();
}

public void oxygen(Runnable releaseOxygen) throws InterruptedException {

// releaseOxygen.run() outputs "O". Do not change or remove this line.
releaseOxygen.run();
}
}
```

Python3:

```
class H2O:
def __init__(self):
pass



def hydrogen(self, releaseHydrogen: 'Callable[[], None]') -> None:

# releaseHydrogen() outputs "H". Do not change or remove this line.
releaseHydrogen()



def oxygen(self, releaseOxygen: 'Callable[[], None]') -> None:

# releaseOxygen() outputs "O". Do not change or remove this line.
releaseOxygen()
```