Problem 1195: Fizz Buzz Multithreaded

Problem Information

Difficulty: Medium
Acceptance Rate: 74.42%
Paid Only: No
Tags: Concurrency

Problem Description

You have the four functions:

* `printFizz` that prints the word `"fizz"` to the console, * `printBuzz` that prints the word `"buzz"` to the console, * `printFizzBuzz` that prints the word `"fizzbuzz"` to the console, and * `printNumber` that prints a given integer to the console.

You are given an instance of the class `FizzBuzz` that has four functions: `fizz`, `buzz`, `fizzbuzz` and `number`. The same instance of `FizzBuzz` will be passed to four different threads:

* **Thread A:** calls `fizz()` that should output the word `"fizz"`. * **Thread B:** calls `buzz()` that should output the word `"buzz"`. * **Thread C:** calls `fizzbuzz()` that should output the word `"fizzbuzz"`. * **Thread D:** calls `number()` that should only output the integers.

Modify the given class to output the series `[1, 2, "fizz", 4, "buzz", ...]` where the `ith` token (**1-indexed**) of the series is:

* `"fizzbuzz"` if `i` is divisible by `3` and `5`, * `"fizz"` if `i` is divisible by `3` and not `5`, * `"buzz"` if `i` is divisible by `5` and not `3`, or * `i` if `i` is not divisible by `3` or `5`.

Implement the `FizzBuzz` class:

* `FizzBuzz(int n)` Initializes the object with the number `n` that represents the length of the sequence that should be printed. * `void fizz(printFizz)` Calls `printFizz` to output `"fizz"`. * `void buzz(printBuzz)` Calls `printBuzz` to output `"buzz"`. * `void fizzbuzz(printFizzBuzz)` Calls `printFizzBuzz` to output `"fizzbuzz"`. * `void number(printNumber)` Calls `printnumber`

to output the numbers.

Example 1:

Input: n = 15 **Output:** [1,2,"fizz",4,"buzz","fizz",7,8,"fizz","buzz",11,"fizz",13,14,"fizzbuzz"]

Example 2:

Input: n = 5 **Output:** [1,2,"fizz",4,"buzz"]

Constraints:

* `1 <= n <= 50`

Code Snippets

C++:

```
class FizzBuzz {
private:
int n;

public:
FizzBuzz(int n) {
this->n = n;
}

// printFizz() outputs "fizz".
void fizz(function<void()> printFizz) {

}

// printBuzz() outputs "buzz".
void buzz(function<void()> printBuzz) {

}

// printFizzBuzz() outputs "fizzbuzz".
void fizzbuzz(function<void()> printFizzBuzz) {
}

// printNumber(x) outputs "x", where x is an integer.
void number(function<void(int)> printNumber) {

}
};
```

Java:

```
class FizzBuzz {
private int n;

public FizzBuzz(int n) {
this.n = n;
}

// printFizz.run() outputs "fizz".
public void fizz(Runnable printFizz) throws InterruptedException {

}

// printBuzz.run() outputs "buzz".
public void buzz(Runnable printBuzz) throws InterruptedException {

}

// printFizzBuzz.run() outputs "fizzbuzz".
public void fizzbuzz(Runnable printFizzBuzz) throws InterruptedException {

}

// printNumber.accept(x) outputs "x", where x is an integer.
public void number(IntConsumer printNumber) throws InterruptedException {

}
}
```

Python3:

```
class FizzBuzz:
def __init__(self, n: int):
self.n = n

# printFizz() outputs "fizz"
def fizz(self, printFizz: 'Callable[[], None]') -> None:



# printBuzz() outputs "buzz"
def buzz(self, printBuzz: 'Callable[[], None]') -> None:



# printFizzBuzz() outputs "fizzbuzz"
def fizzbuzz(self, printFizzBuzz: 'Callable[[], None]') -> None:



# printNumber(x) outputs "x", where x is an integer.
def number(self, printNumber: 'Callable[[int], None]') -> None:
```