SKILL-08

Threads

2100030059

B. Revathi

S-12

1. Fizz Buzz Multithreaded:-

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 i^{th} 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

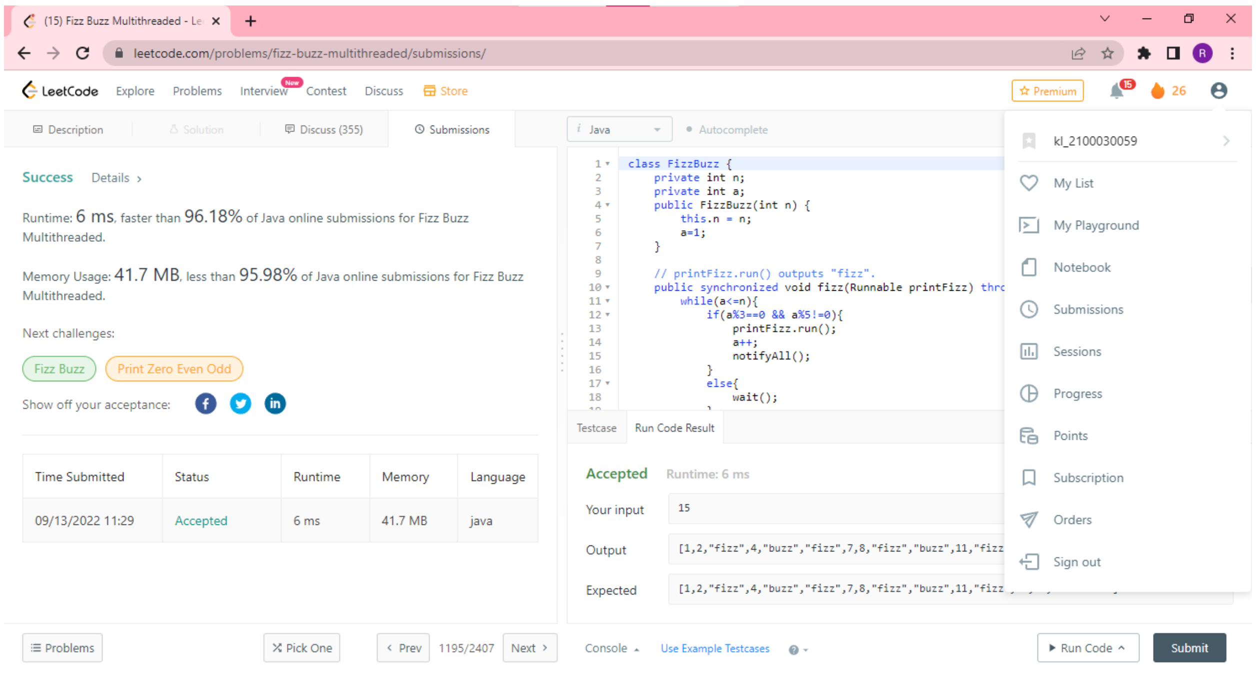
Output: [1,2,"fizz",4,"buzz"]

Constraints:

- 1 <= n <= 50

ANSWER:-

```
lass FizzBuzz {

    private int n;

    private int a;

    public FizzBuzz(int n) {

        this.n = n;

        a=1;
    }


    // printFizz.run() outputs "fizz".
    public synchronized void fizz(Runnable printFizz) throws InterruptedException {
        while(a<=n){
            if(a%3==0 && a%5!=0){
                printFizz.run();
                a++;
                notifyAll();
            }
            else{
                wait();
            }
        }
    }


    // printBuzz.run() outputs "buzz".
    public synchronized void buzz(Runnable printBuzz) throws InterruptedException {
        while(a<=n){
            if(a%3!=0 && a%5==0){
                printBuzz.run();
                a++;
                notifyAll();
            }
            else{
                wait();
            }
        }

    }



    // printFizzBuzz.run() outputs "fizzbuzz".

    public synchronized void fizzbuzz(Runnable printFizzBuzz) throws
InterruptedException {

        while(a<=n){

            if(a%3==0 && a%5==0){

                printFizzBuzz.run();

                a++;

                notifyAll();

            }

            else{

                wait();

            }

        }

    }



    // printNumber.accept(x) outputs "x", where x is an integer.

    public synchronized void number(IntConsumer printNumber) throws
InterruptedException {

        while(a<=n){

            if(a%3!=0 && a%5!=0){

                printNumber.accept(a);

                a++;

                notifyAll();

            }

            else{
                wait();

            }

        }

    }

}
```



(15) Fizz Buzz Multithreaded - Le

leetcode.com/problems/fizz-buzz-multithreaded/submissions/

Success Details

Runtime: 6 ms, faster than 96.18% of Java online submissions for Fizz Buzz Multithreaded.

Memory Usage: 41.7 MB, less than 95.98% of Java online submissions for Fizz Buzz Multithreaded.

Next challenges: Fizz Buzz, Print Zero Even Odd

Time Submitted	Status	Runtime	Memory	Language
09/13/2022 11:29	Accepted	6 ms	41.7 MB	java

```
class FizzBuzz {
    private int n;
    private int a;
    public FizzBuzz(int n) {
        this.n = n;
        a=1;
    }

    // printFizz.run() outputs "fizz".
    public synchronized void fizz(Runnable printFizz) thro
        while(a<=n){
            if(a%3==0 && a%5!=0){
                printFizz.run();
                a++;
                notifyAll();
            }
            else{
                wait();
```

Accepted Runtime: 6 ms

Your input: 15

Output: [1,2,"fizz",4,"buzz","fizz",7,8,"fizz","buzz",11,"fizz

Expected: [1,2,"fizz",4,"buzz","fizz",7,8,"fizz","buzz",11,"fizz

2. <u>Building H2O:-</u>

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.

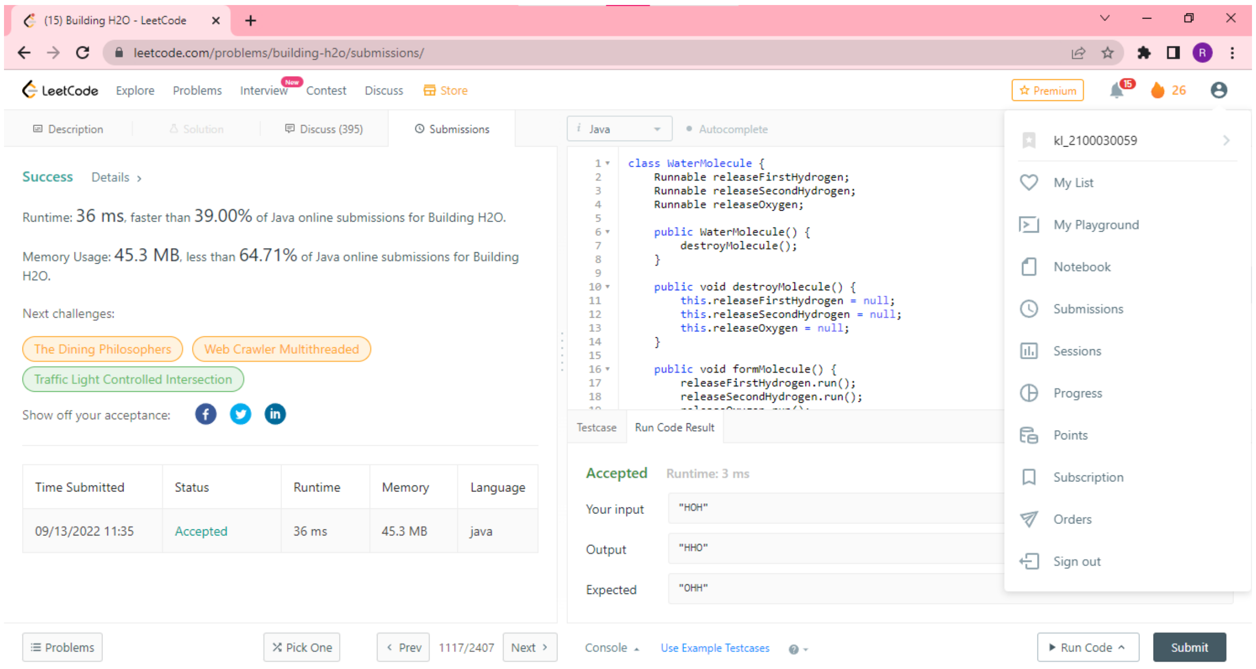
Constraints:

- 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.

ANSWER:-

```
class WaterMolecule {

    Runnable releaseFirstHydrogen;

    Runnable releaseSecondHydrogen;

    Runnable releaseOxygen;



    public WaterMolecule() {

        destroyMolecule();
    }


    public void destroyMolecule() {

        this.releaseFirstHydrogen = null;

        this.releaseSecondHydrogen = null;

        this.releaseOxygen = null;

    }


    public void formMolecule() {

        releaseFirstHydrogen.run();

        releaseSecondHydrogen.run();

        releaseOxygen.run();

    }


    public boolean canMoleculeBeFormed() {

        return ((releaseFirstHydrogen != null) && (releaseSecondHydrogen != null) &&
(releaseOxygen != null));

    }

}


class H2O {

    private static final WaterMolecule waterMolecule = new WaterMolecule();


    public H2O() {}


    public void hydrogen(Runnable releaseHydrogen) throws InterruptedException {

                synchronized(waterMolecule) {
```

```
            while((waterMolecule.releaseFirstHydrogen != null) &&
(waterMolecule.releaseSecondHydrogen != null)) {

                waterMolecule.wait();

            }

            if(waterMolecule.releaseFirstHydrogen == null) {

                waterMolecule.releaseFirstHydrogen = releaseHydrogen;

            } else {

                waterMolecule.releaseSecondHydrogen = releaseHydrogen;

            }

            if(waterMolecule.canMoleculeBeFormed()) {

                waterMolecule.formMolecule();

                waterMolecule.destroyMolecule();

            }

            waterMolecule.notifyAll();

        }

    }


    public void oxygen(Runnable releaseOxygen) throws InterruptedException {

        synchronized(waterMolecule) {

            while(waterMolecule.releaseOxygen != null) {

                waterMolecule.wait();

            }

            waterMolecule.releaseOxygen = releaseOxygen;

            if(waterMolecule.canMoleculeBeFormed()) {

                waterMolecule.formMolecule();

                waterMolecule.destroyMolecule();

            }

            waterMolecule.notifyAll();
        }
    }
}
```



Success Details

Runtime: 36 ms, faster than 39.00% of Java online submissions for Building H2O.

Memory Usage: 45.3 MB, less than 64.71% of Java online submissions for Building H2O.

Next challenges: The Dining Philosophers, Web Crawler Multithreaded, Traffic Light Controlled Intersection

Time Submitted	Status	Runtime	Memory	Language
09/13/2022 11:35	Accepted	36 ms	45.3 MB	java

```java
class WaterMolecule {
    Runnable releaseFirstHydrogen;
    Runnable releaseSecondHydrogen;
    Runnable releaseOxygen;

    public WaterMolecule() {
        destroyMolecule();
    }

    public void destroyMolecule() {
        this.releaseFirstHydrogen = null;
        this.releaseSecondHydrogen = null;
        this.releaseOxygen = null;
    }

    public void formMolecule() {
        releaseFirstHydrogen.run();
        releaseSecondHydrogen.run();
```

Accepted Runtime: 3 ms

Your input "HOH"

Output "HHO"

Expected "OHH"

3. <u>Print in Order:-</u>

Suppose we have a class:

```
public class Foo {
  public void first() { print("first"); }
  public void second() { print("second"); }
  public void third() { print("third"); }
}
```

The same instance of Foo will be passed to three different threads. Thread A will call first(), thread B will call second(), and thread C will call third(). Design a mechanism and modify the program to ensure that second() is executed after first(), and third() is executed after second().

Note:

We do not know how the threads will be scheduled in the operating system, even though the numbers in the input seem to imply the ordering. The input format you see is mainly to ensure our tests' comprehensiveness.

Example 1:

Input: nums = [1,2,3]

Output: "firstsecondthird"

Explanation: There are three threads being fired asynchronously. The input [1,2,3] means thread A calls first(), thread B calls second(), and thread C calls third(). "firstsecondthird" is the correct output.

Example 2:

Input: nums = [1,3,2]

Output: "firstsecondthird"

Explanation: The input [1,3,2] means thread A calls first(), thread B calls third(), and thread C calls second(). "firstsecondthird" is the correct output.

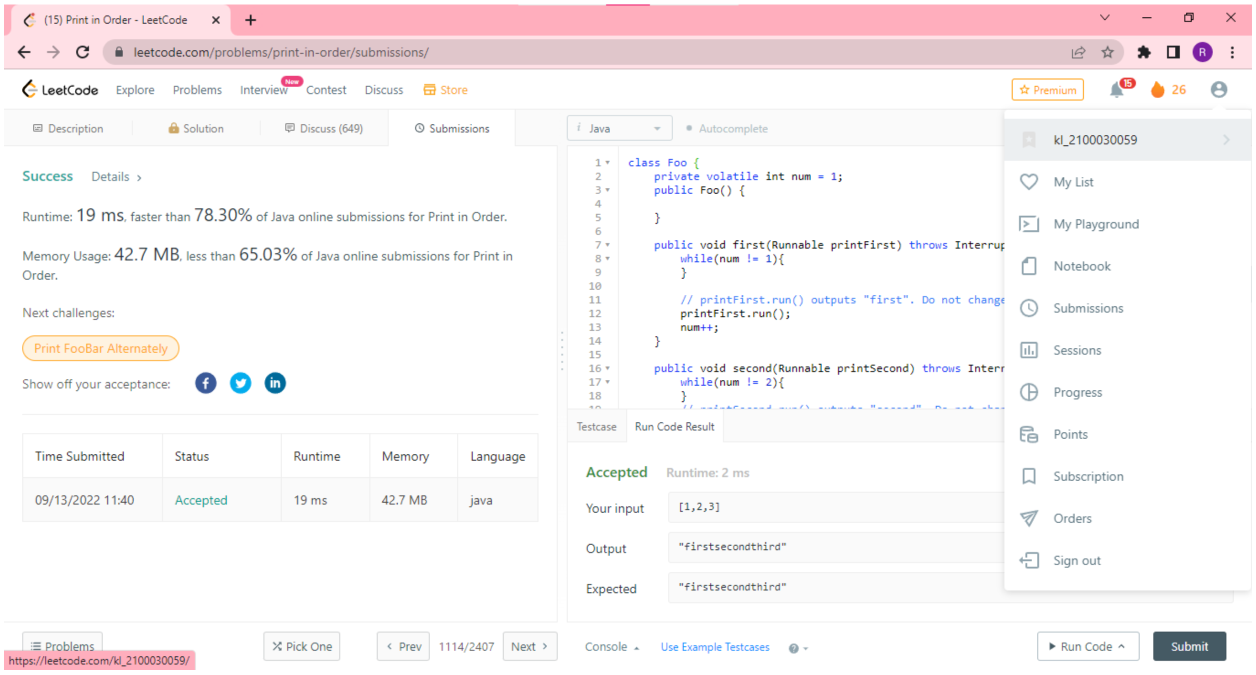
Constraints:

- nums is a permutation of [1, 2, 3].

ANSWER:-

```
class Foo {

    private volatile int num = 1;

    public Foo() {


    }


    public void first(Runnable printFirst) throws InterruptedException {

        while(num != 1){

        }


        // printFirst.run() outputs "first". Do not change or remove this line.

        printFirst.run();

        num++;

    }
```

```java
    public void second(Runnable printSecond) throws InterruptedException {
        while(num != 2){
        }
        // printSecond.run() outputs "second". Do not change or remove this line.
        printSecond.run();
        num++;
    }

    public void third(Runnable printThird) throws InterruptedException {
        while(num != 3){
        }
        // printThird.run() outputs "third". Do not change or remove this line.
        printThird.run();
        num++;
    }
}
```



(15) Print in Order - LeetCode

leetcode.com/problems/print-in-order/submissions/

Success Details

Runtime: 19 ms, faster than 78.30% of Java online submissions for Print in Order.

Memory Usage: 42.7 MB, less than 65.03% of Java online submissions for Print in Order.

Next challenges:

Print FooBar Alternately

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
09/13/2022 11:40	Accepted	19 ms	42.7 MB	java

```java
class Foo {
    private volatile int num = 1;
    public Foo() {

    }

    public void first(Runnable printFirst) throws Interrup
        while(num != 1){
        }

        // printFirst.run() outputs "first". Do not change
        printFirst.run();
        num++;
    }

    public void second(Runnable printSecond) throws Interr
        while(num != 2){
        }
```

Accepted Runtime: 2 ms

Your input `[1,2,3]`

Output `"firstsecondthird"`

Expected `"firstsecondthird"`

kl_2100030059

https://leetcode.com/kl_2100030059/