Task 6: Executors, Concurrent Collections, CompletableFuture

Use an ExecutorService to parallelize a task that calculates prime numbers up to a given number and then use CompletableFuture to write the results to a file asynchronously. package com.wipro.model;

```
import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.List;

import java.util.concurrent.*;


public class PrimeNumberParallelFileWriter {


    public static void main(String[] args) throws InterruptedException, ExecutionException {
        int max = 100;
        String fn = "prime_numbers.txt";
        ExecutorService executor = Executors.newFixedThreadPool(4);
        List<Future<List<Integer>>> futures = new ArrayList<>();
        int size = 4;
        for (int i=0;i<size;i++) {
            int start=i*(max/size)+1;
            int end=(i+1)*(max/size);
            if (i==size-1) {
                end=max;
            }
            Future<List<Integer>> future=executor.submit(new PrimeNumberCalculator(start,end));
            futures.add(future);
        }
        List<Integer> allPrimes=new ArrayList<>();
        for (Future<List<Integer>>future:futures) {
            allPrimes.addAll(future.get());
        }
        executor.shutdown();
```

```
    CompletableFuture<Void> writeToFileFuture=CompletableFuture.runAsync(() -> {
        try (BufferedWriter writer=new BufferedWriter(new FileWriter(fn))) {
            for (Integer prime:allPrimes) {
                writer.write(prime+"\n");
            }
            System.out.println("Prime numbers written to "+fn);
        } catch (IOException e) {
            e.printStackTrace();
        }
    });
    writeToFileFuture.get();
    System.out.println("Program completed successfully.");
}
static class PrimeNumberCalculator implements Callable<List<Integer>> {
    private int start;
    private int end;
    public PrimeNumberCalculator(int start, int end) {
        this.start=start;
        this.end=end;
    }
    @Override
    public List<Integer> call() {
        List<Integer> primes = new ArrayList<>();
        for (int num=start;num<=end;num++) {
            if (isPrime(num)) {
                primes.add(num);
            }
        }
        return primes;
    }
    private boolean isPrime(int num) {
```

```
        if (num<=1){
            return false;
        }
        for (int i=2;i<=Math.sqrt(num);i++) {
            if(num%i==0)
            {
                return false;
            }
        }
        return true;
      }
   }
}
```