Ruby

def is\_prime(num)

return false if num <= 1

return true if num <= 3

return false if (num % 2 == 0) || (num % 3 == 0)

i = 5

while i \* i <= num

return false if (num % i == 0) || (num % (i + 2) == 0)

i += 6

end

return true

end

print "Enter no: "

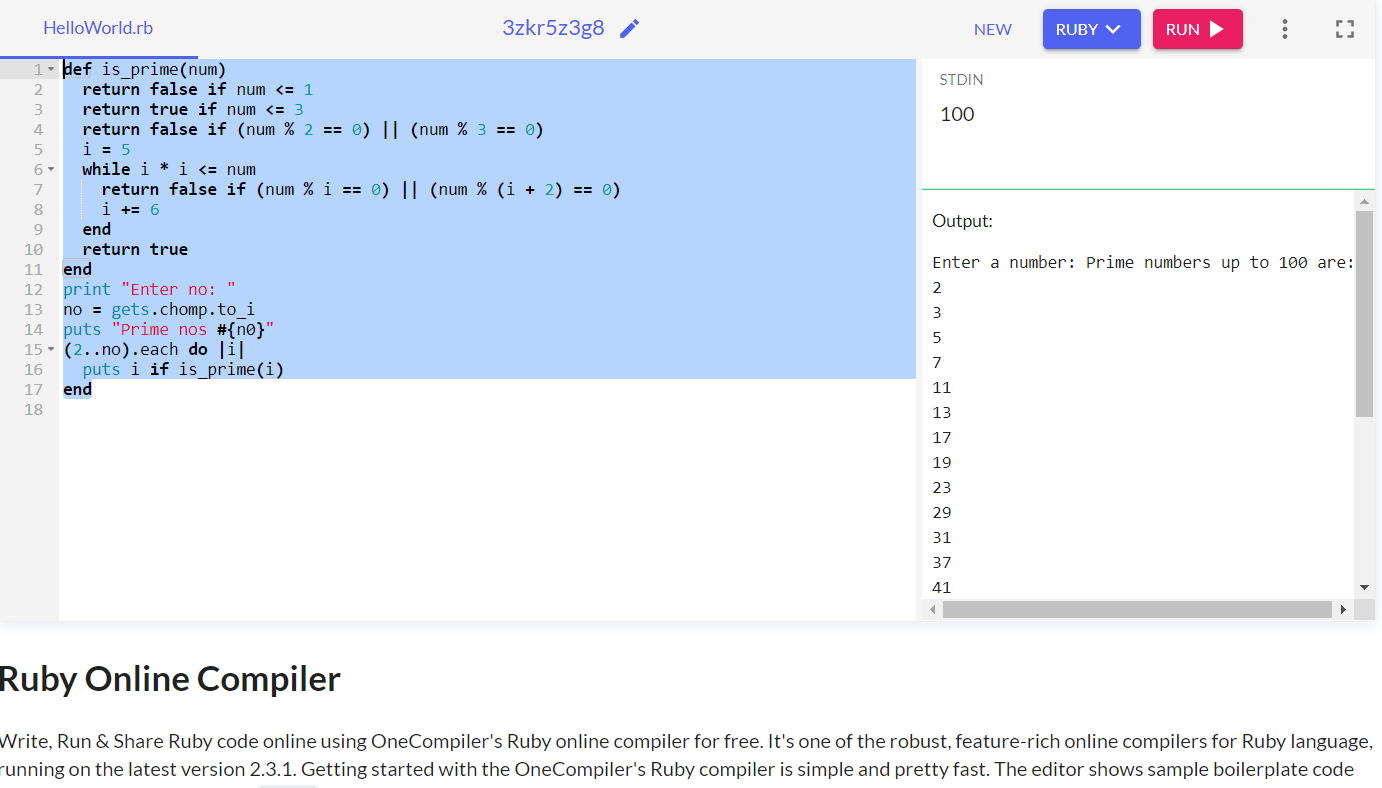
no = gets.chomp.to\_i

puts "Prime nos #{n0}"

(2..no).each do |i|

puts i if is\_prime(i)

end



Python

def is\_prime(num):

if num <= 1:

return False

if num <= 3:

return True

if num % 2 == 0 or num % 3 == 0:

return False

i = 5

while i \* i <= num:

if num % i == 0 or num % (i + 2) == 0:

return False

i += 6

return True

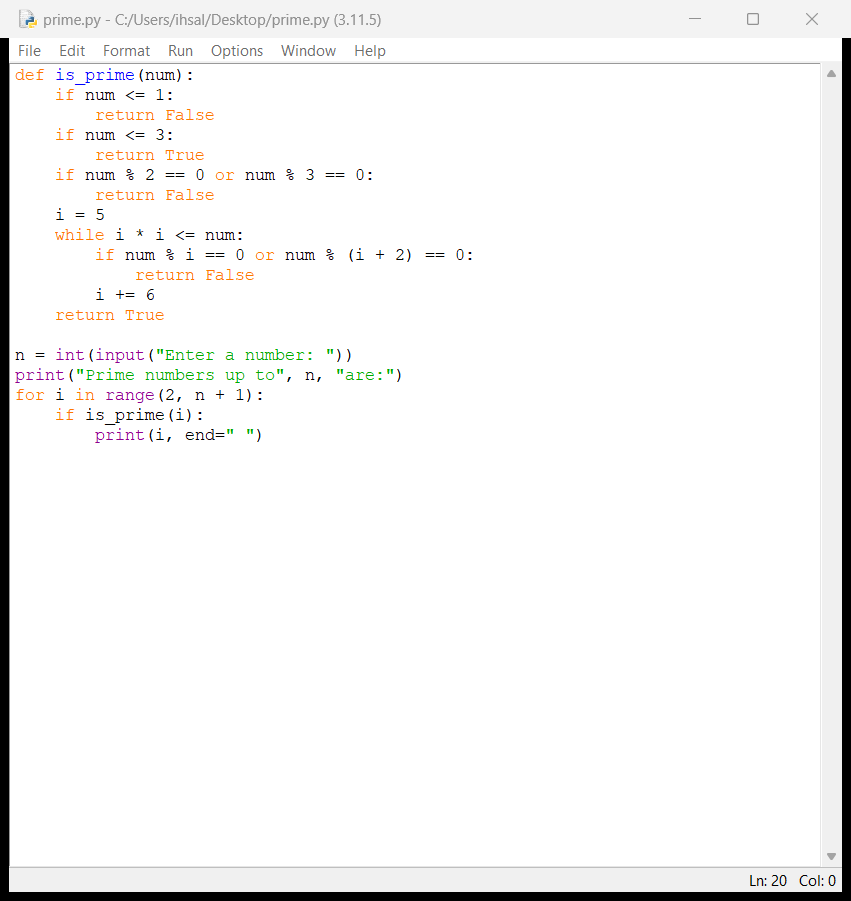
n = int(input("Enter a number: "))

print("Prime numbers up to", n, "are:")

for i in range(2, n + 1):

if is\_prime(i):

print(i, end=" ")



Javascript

const lNo = parseInt(prompt('Enter low(crush rejected him) number: '));

const hNo = parseInt(prompt('Enter high(it smoked too much weed unfortunately) number: '));

console.log(`The prime numbers between ${lNo} and ${hNo} are:`);

for (let i = lNo; i <= hNo; i++) {

let flag = 0;

for (let j = 2; j < i; j++) {

if (i % j == 0) {

flag = 1;

break;

}

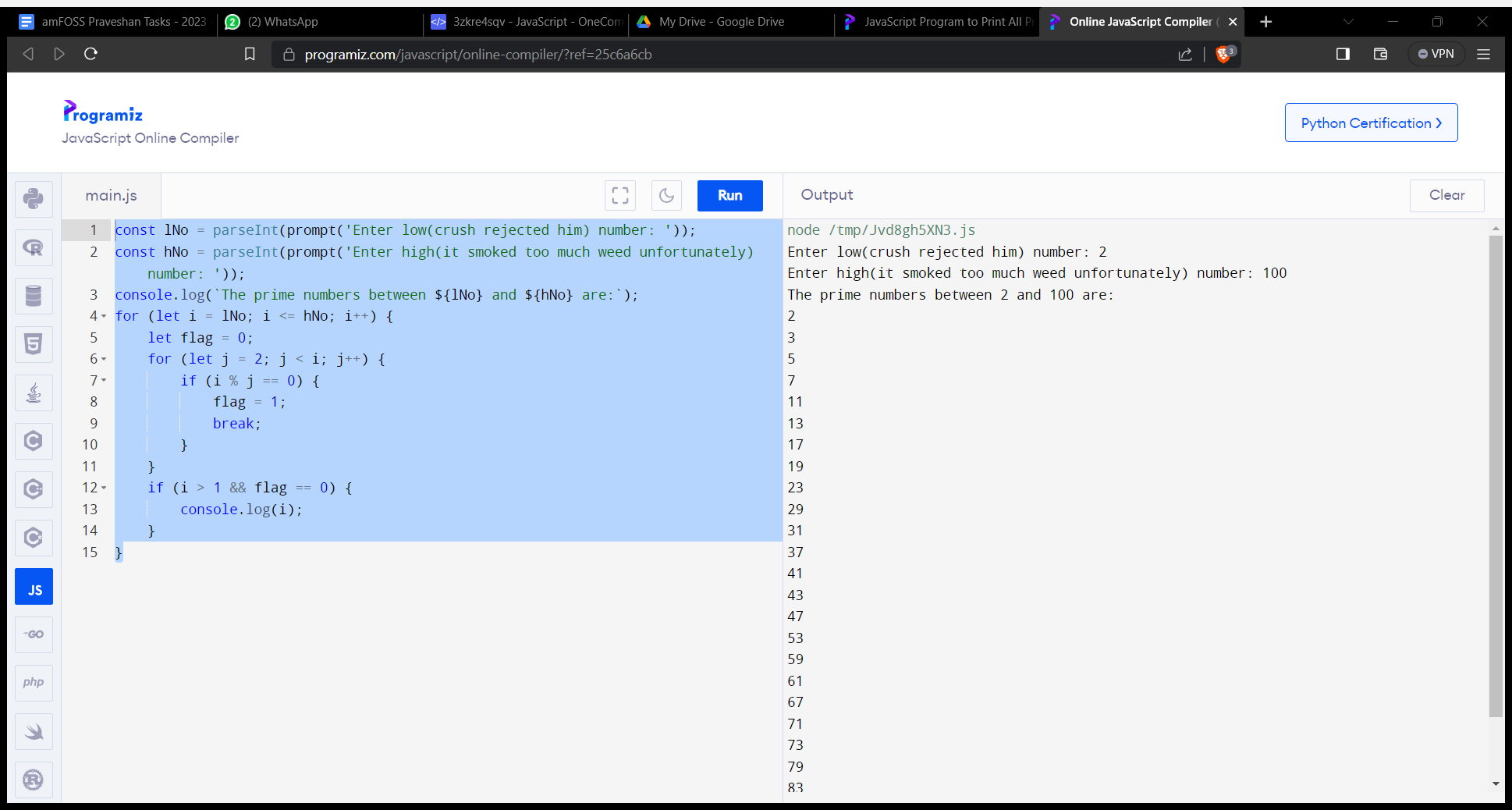
}

if (i > 1 && flag == 0) {

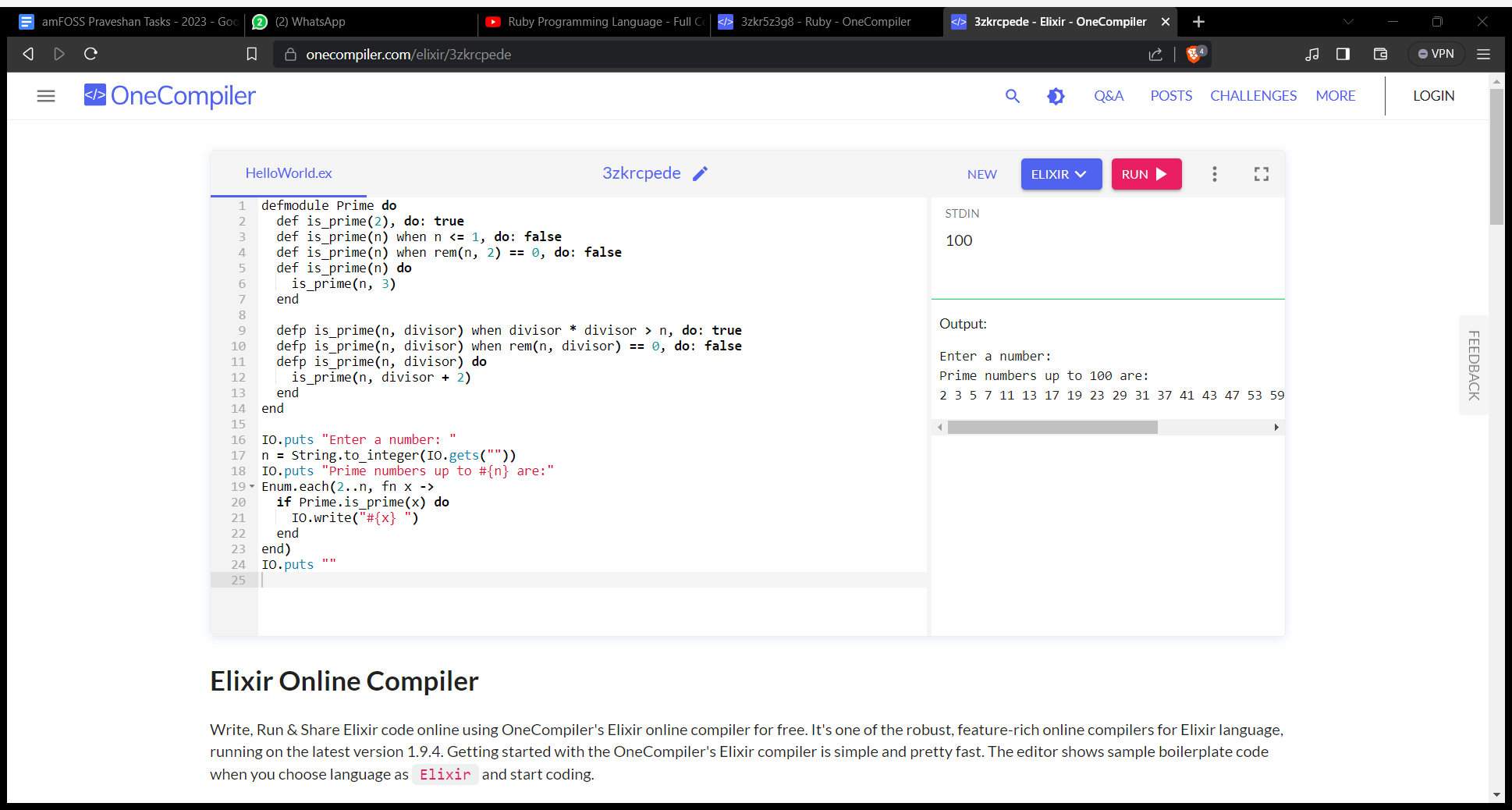
console.log(i);

}

}



Elixir



Java

import java.util.Scanner;

public class PrimeNumbers {

public static boolean isPrime(int num) {

if (num <= 1) return false;

if (num <= 3) return true;

if (num % 2 == 0 || num % 3 == 0) return false;

int i = 5;

while (i \* i <= num) {

if (num % i == 0 || num % (i + 2) == 0) return false;

i += 6;

}

return true;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a no(non zero and not one, obviously): ");

int n = scanner.nextInt();

System.out.print("Prime numbers(Optimus, where are you, when the world truly needs you? " + n + " are: ");

for (int i = 2; i <= n; i++) {

if (isPrime(i)) {

System.out.print(i + " ");

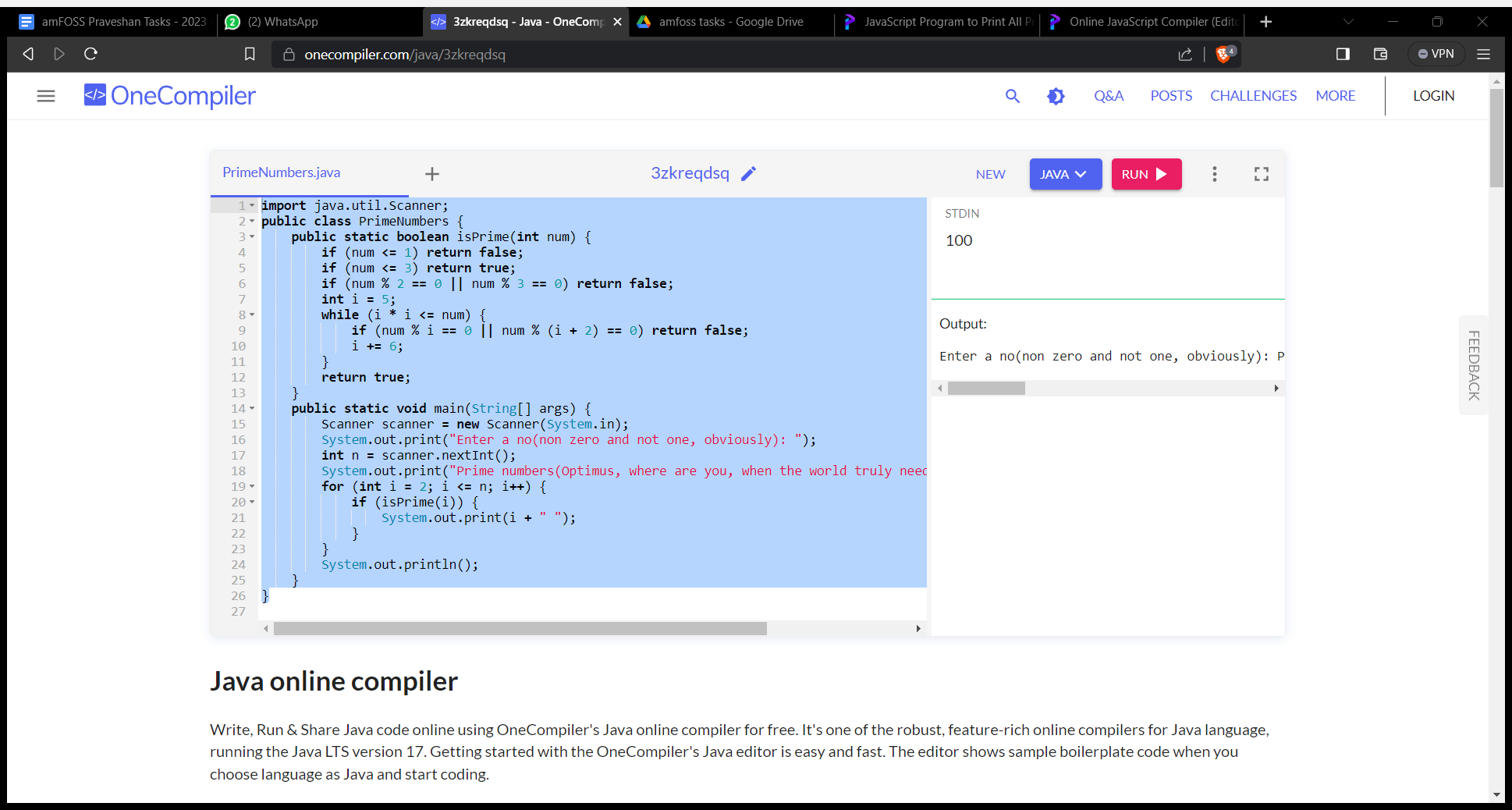
}

}

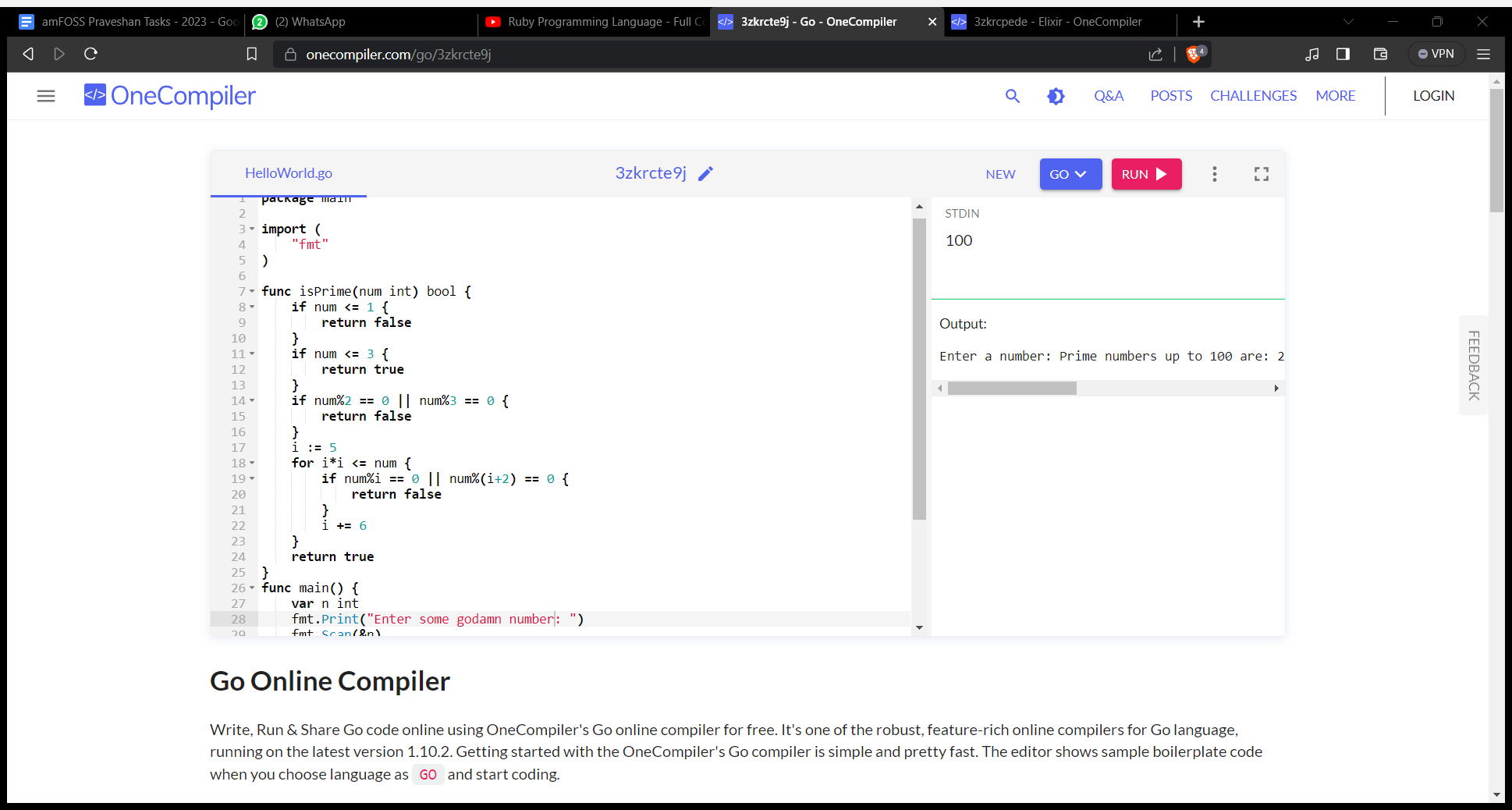
System.out.println();

}

}



Go



C

#include <stdio.h>

#include <stdbool.h>

bool is\_prime(int num) {

if (num <= 1) return false;

if (num <= 3) return true;

if (num % 2 == 0 || num % 3 == 0) return false;

int i = 5;

while (i \* i <= num) {

if (num % i == 0 || num % (i + 2) == 0) return false;

i += 6;

}

return true;

}

int main() {

int n;

printf("Enter a godammit number(I wonder if anyone will notice this?): ");

scanf("%d", &n);

printf("Prime numbers(sidenote: where is OptimusPrime tho?) %d are: ", n);

for (int i = 2; i <= n; i++) {

if (is\_prime(i)) {

printf("%d ", i);

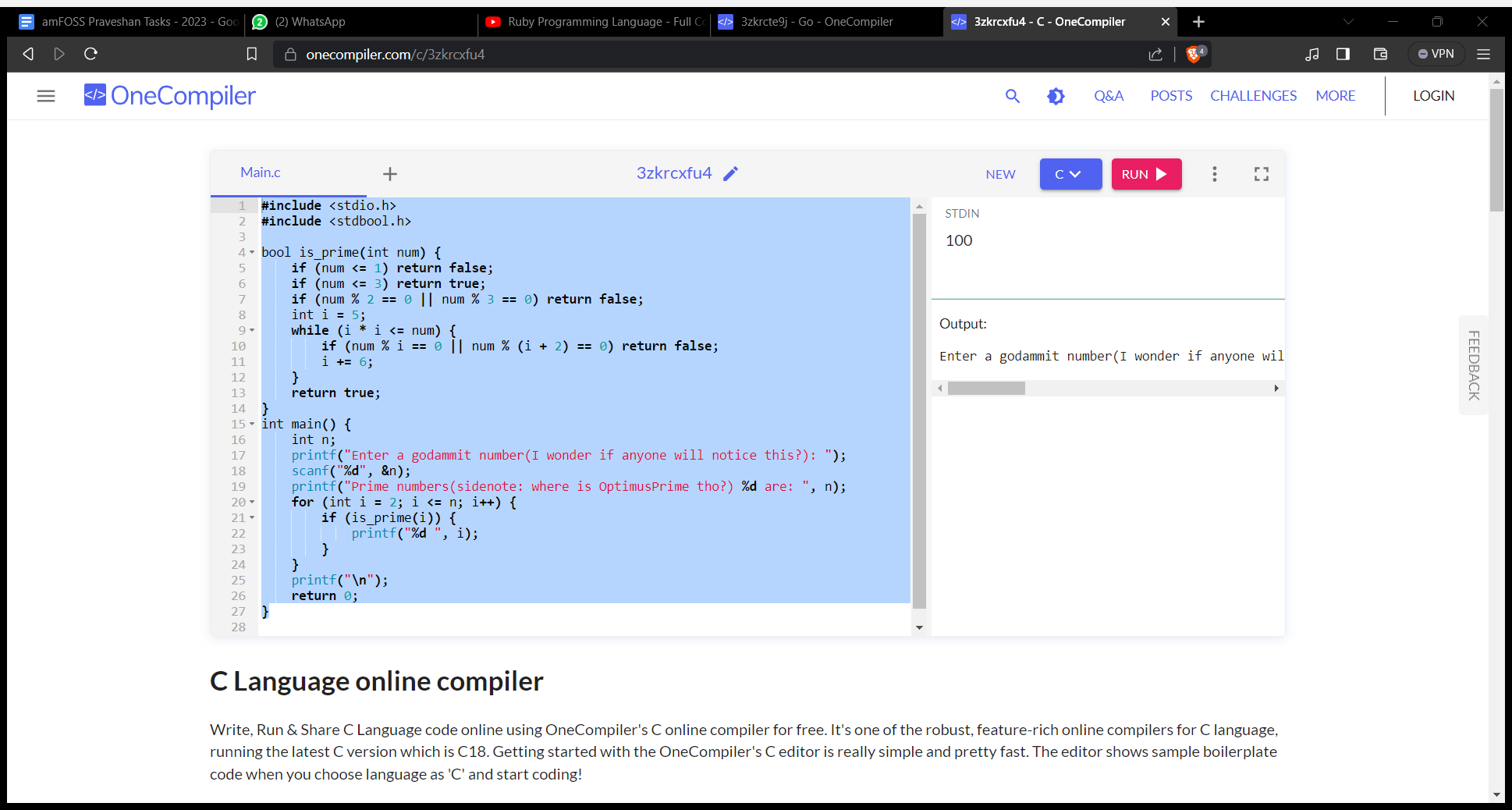
}

}

printf("\n");

return 0;

}



C++

#include <iostream>

bool is\_prime(int num) {

if (num <= 1) return false;

if (num <= 3) return true;

if (num % 2 == 0 || num % 3 == 0) return false;

int i = 5;

while (i \* i <= num) {

if (num % i == 0 || num % (i + 2) == 0) return false;

i += 6;

}

return true;

}

int main() {

int n;

std::cout << "Enter satannumber(get it? since the last one was goddamit?) number: ";

std::cin >> n;

std::cout << "Prime no are: ";

for (int i = 2; i <= n; i++) {

if (is\_prime(i)) {

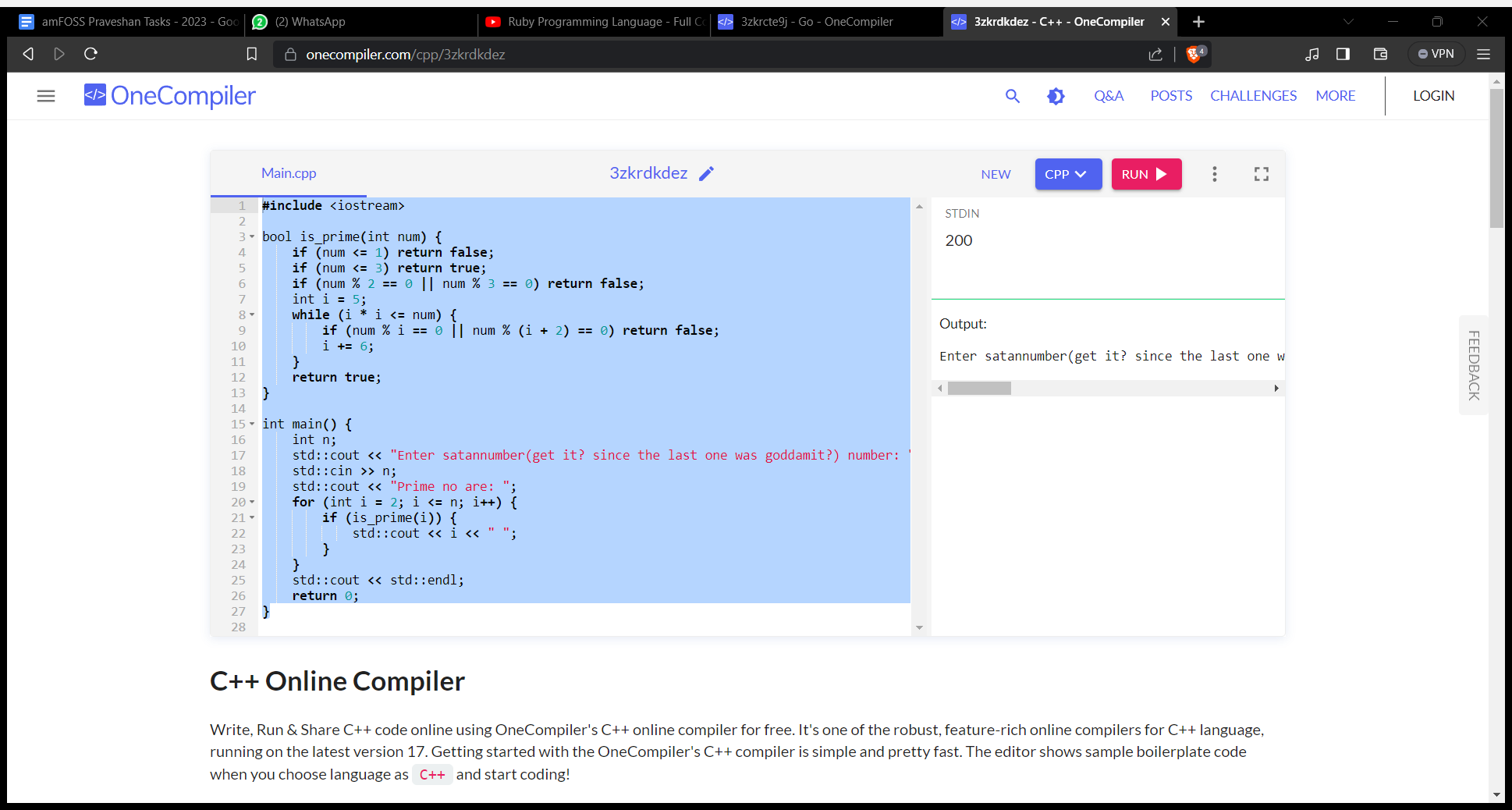
std::cout << i << " ";

}

}

std::cout << std::endl;

return 0;

}

Rusk

use std::io;

fn is\_prime(num: u32) -> bool {

if num <= 1 {

return false;

}

if num <= 3 {

return true;

}

if num % 2 == 0 || num % 3 == 0 {

return false;

}

let mut i = 5;

while i \* i <= num {

if num % i == 0 || num % (i + 2) == 0 {

return false;

}

i += 6;

}

true

}

fn main() {

let mut input = String::new();

println!("you know, this task is a lot more annoying than i thought it was: anyways, enter a number");

io::stdin().read\_line(&mut input).expect("Failed to read line");

let n: u32 = input.trim().parse().expect("Invalid input");

println!("Prime stuff's {} are:", n);

for i in 2..=n {

if is\_prime(i) {

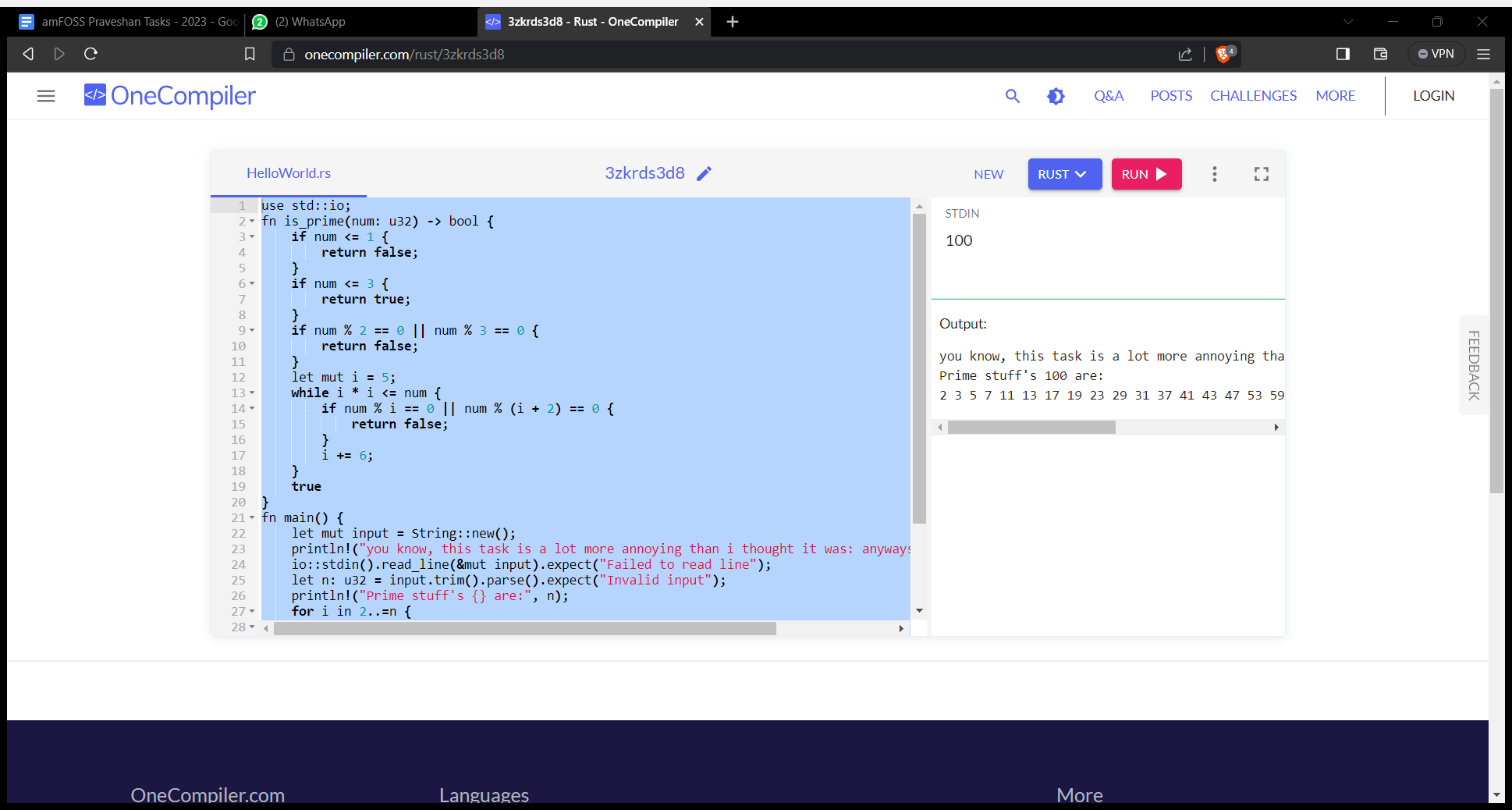
print!("{} ", i);

}

}

println!();

}



isPrime :: Int -> Bool

isPrime n

| n <= 1 = False

| n <= 3 = True

| n `mod` 2 == 0 || n `mod` 3 == 0 = False

| otherwise = isPrime' n 5

where

isPrime' n i

| i \* i > n = True

| n `mod` i == 0 || n `mod` (i + 2) == 0 = False

| otherwise = isPrime' n (i + 6)

main :: IO ()

main = do

putStrLn "Enter a number(this is the last one, thank god): "

n <- readLn :: IO Int

putStr "Prime numbers up to "

putStr (show n)

putStr " are: "

let primes = filter isPrime [2..n]

putStrLn (unwords (map show primes))

