First task

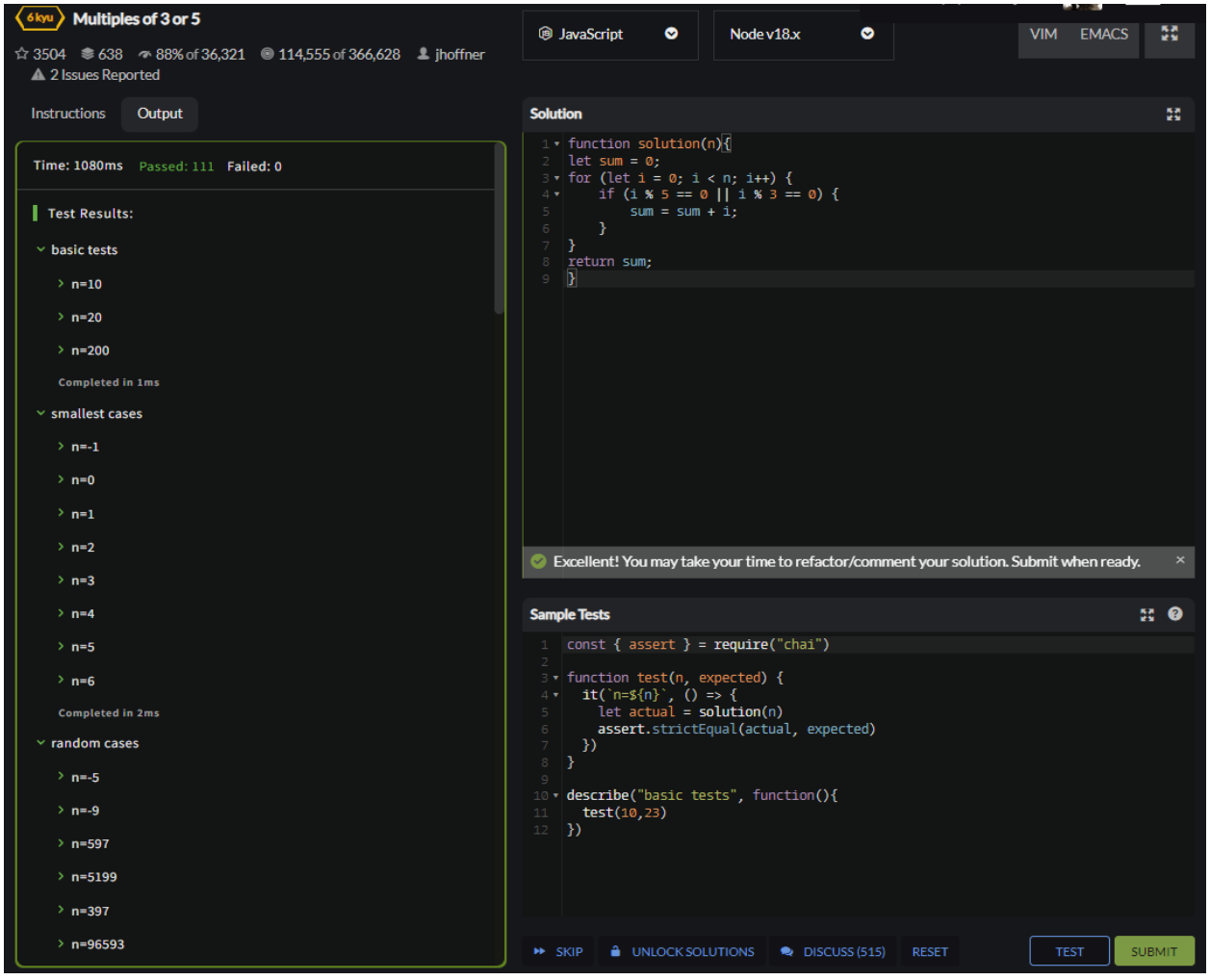
If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Finish the solution so that it returns the sum of all the multiples of 3 or 5 below the number passed in.

Additionally, if the number is negative, return 0.

Note: If the number is a multiple of both 3 and 5, only count it once.

```
function solution(n){
let sum = 0;
for (let i = 0; i < n; i++) {
    if (i % 5 == 0 || i % 3 == 0) {
        sum = sum + i;
    }
}
return sum;
}
```



Second task

Your task, is to calculate the minimal number of moves to win the game "Towers of Hanoi", with given number of disks.

Towers of Hanoi is a simple game consisting of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

Only one disk can be moved at a time.

Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.

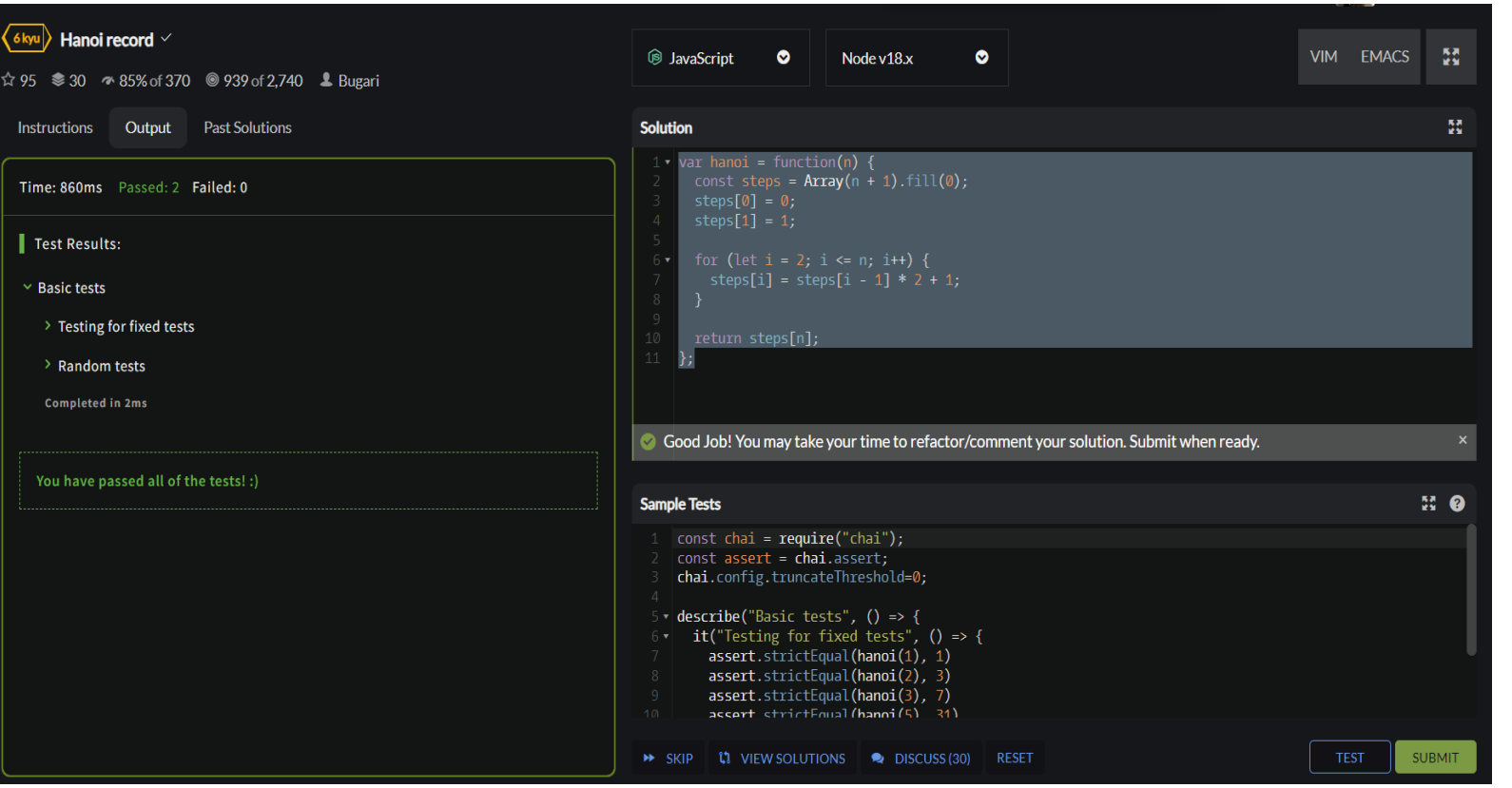
No disk may be placed on top of a smaller disk.

```
var hanoi = function(n) {
  const steps = Array(n + 1).fill(0);
  steps[0] = 0;
  steps[1] = 1;

  for (let i = 2; i <= n; i++) {
    steps[i] = steps[i - 1] * 2 + 1;
  }

  return steps[n];
};

};
```



Third task

Your task is to construct a building which will be a pile of n cubes. The cube at the bottom will have a volume of n^3, the cube above will have volume of (n-1)^3 and so on until the top which will have a volume of 1^3.

You are given the total volume m of the building. Being given m can you find the number n of cubes you will have to build?

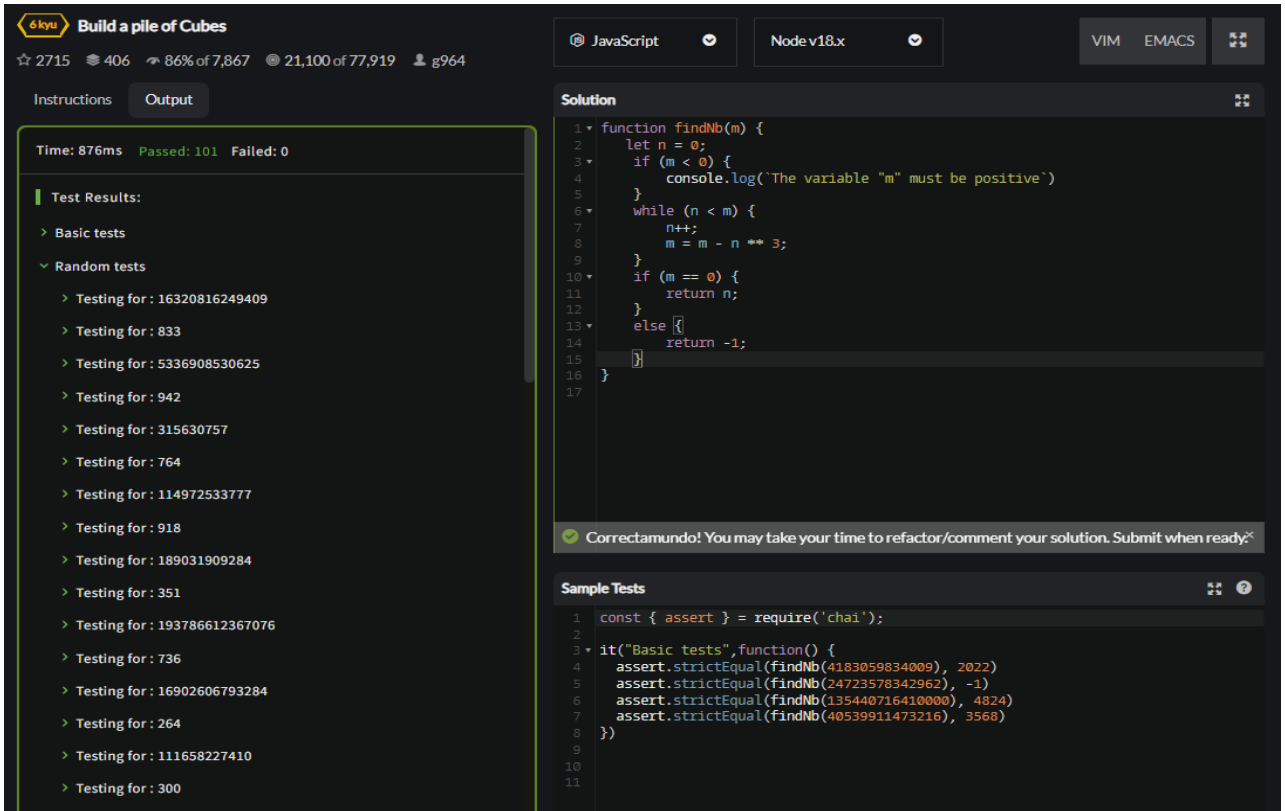
The parameter of the function findNb (find_nb, find-nb, findNb, ...) will be an integer m and you have to return the integer n such as n^3 + (n-1)^3 + (n-2)^3 + ... + 1^3 = m if such a n exists or -1 if there is no such n.

Examples:

findNb(1071225) --> 45

findNb(91716553919377) --> -1

```
function findNb(m) {
  let n = 0;
   if (m < 0) {
      console.log(`The variable "m" must be positive`)
   }
   while (n < m) {
      n++;
      m = m - n ** 3;
   }
   if (m == 0) {
      return n;
   }
   else {
      return -1;
   }
}
```



Fourth task

Define a function that takes an integer argument and returns a logical value true or false depending on if the integer is a prime.

Per Wikipedia, a prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself.

Requirements:

- You can assume you will be given an integer input.

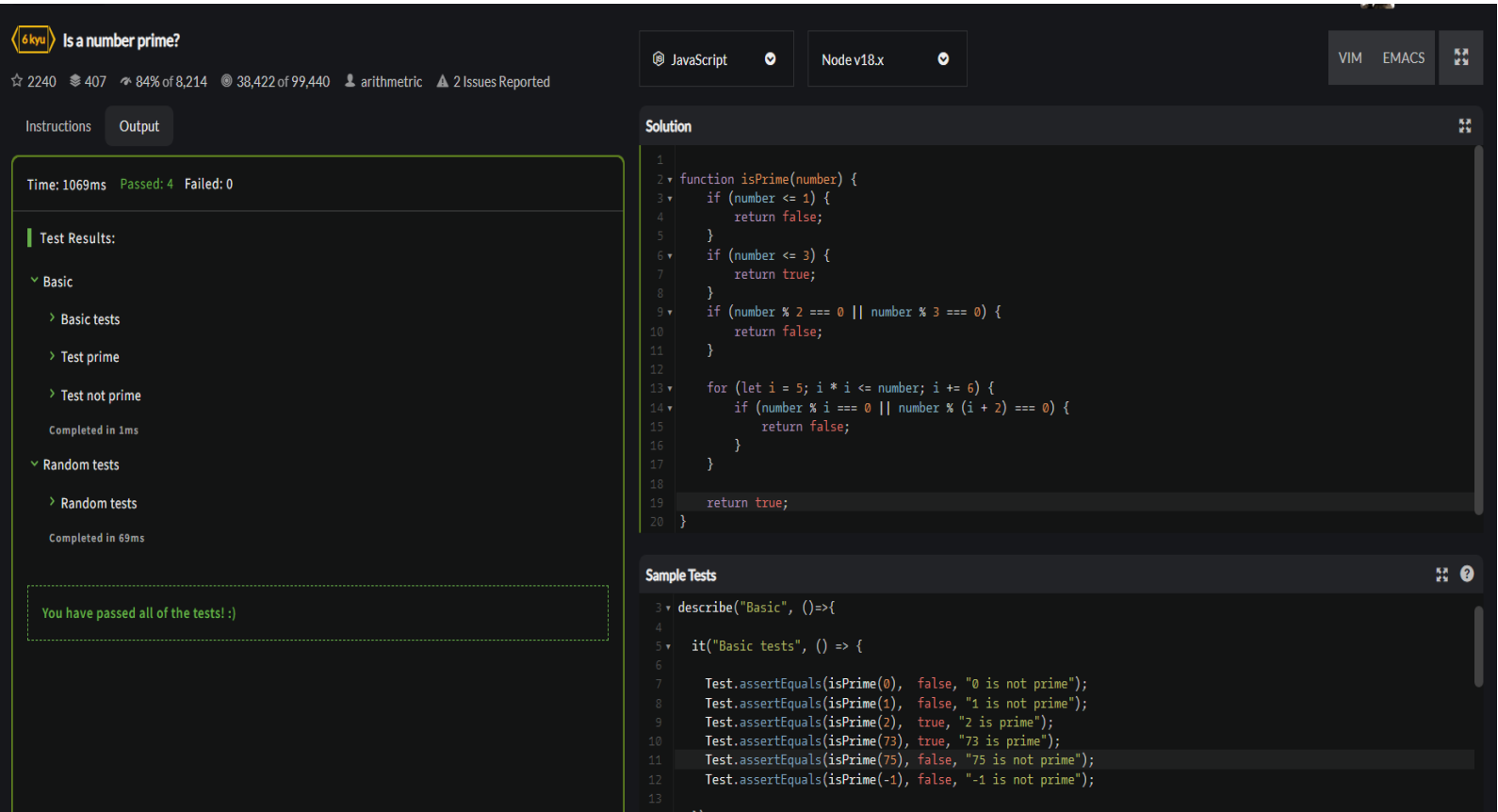
- You can not assume that the integer will be only positive. You may be given negative numbers as well (or 0).

- NOTE on performance: There are no fancy optimizations required, but still the most trivial solutions might time out. Numbers go up to 2^31 (or similar, depending on language). Looping all the way up to n, or n/2, will be too slow.

```
function isPrime(number) {
    if (number <= 1) {
        return false;
    }
    if (number <= 3) {
        return true;
    }
    if (number % 2 === 0 || number % 3 === 0) {
        return false;
    }

    for (let i = 5; i * i <= number; i += 6) {
        if (number % i === 0 || number % (i + 2) === 0) {
            return false;
        }
    }

    return true;
}
```



Fifth task

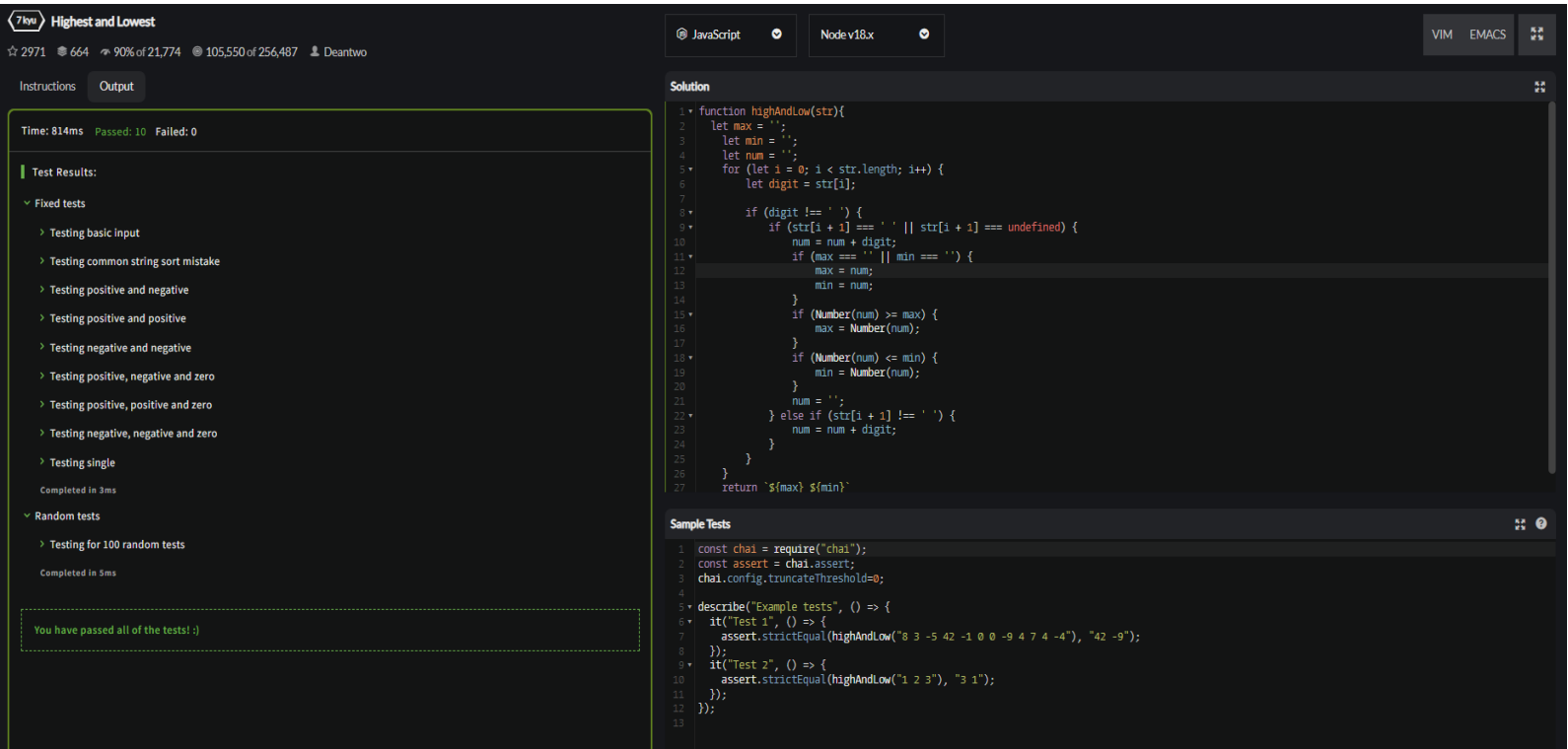
In this little assignment you are given a string of space separated numbers, and have to return the highest and lowest number.

Examples:

```
highAndLow("1 2 3 4 5");  // return "5 1"
highAndLow("1 2 -3 4 5"); // return "5 -3"
highAndLow("1 9 3 4 -5"); // return "9 -5"
```

```
function highAndLow(str){
  let max = '';
    let min = '';
    let num = '';
    for (let i = 0; i < str.length; i++) {
        let digit = str[i];

        if (digit !== ' ') {
            if (str[i + 1] === ' ' || str[i + 1] === undefined) {
                num = num + digit;
                if (max === '' || min === '') {
                    max = num;
                    min = num;
                }
                if (Number(num) >= max) {
                    max = Number(num);
                }
                if (Number(num) <= min) {
                    min = Number(num);
                }
                num = '';
            } else if (str[i + 1] !== ' ') {
                num = num + digit;
            }
        }
    }
    return `${max} ${min}`
}
```



Sixth task

Your task is to make a function that can take any non-negative integer as an argument and return it with its digits in descending order. Essentially, rearrange the digits to create the highest possible number.

Examples:

Input: 42145 Output: 54421

Input: 145263 Output: 654321

Input: 123456789 Output: 987654321

```
function descendingOrder(number){
  let n = String(number)
    if (n < 0 || n % 1 != 0) {
        return console.log(`Only positive integer numbers are  allowed `);
    }
    else {
        let sort_num = '';
        for (let i = 0; i < n.length; i++) {
            let digit = n[i];
            let j = 0;

            while (j < sort_num.length && digit < sort_num[j]) {
                j++;
            }
            sort_num = sort_num.slice(0, j) + digit + sort_num.slice(j);
        }
        return Number(sort_num);
    }
}
```