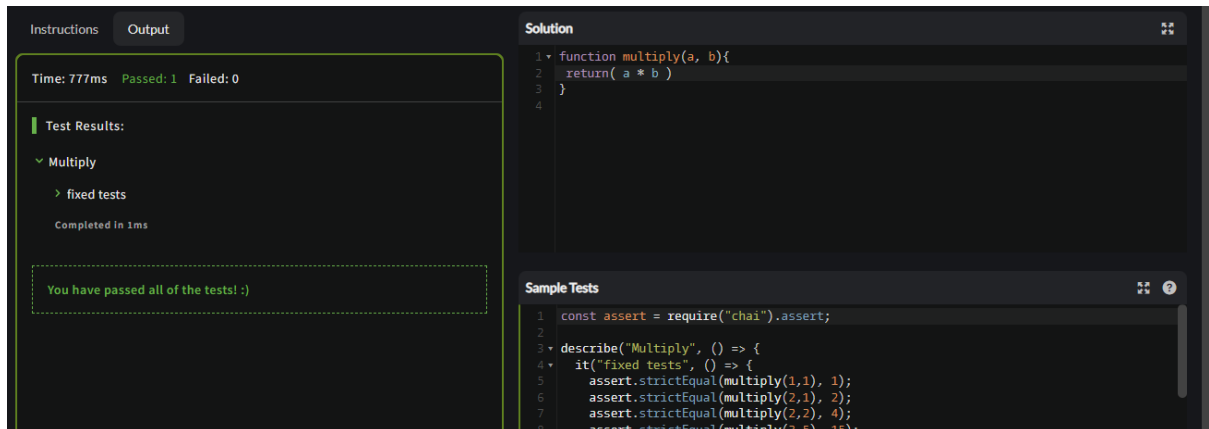


25 program challenge

1. Multiply:

```
function multiply(a, b){  
  return( a * b )  
}
```



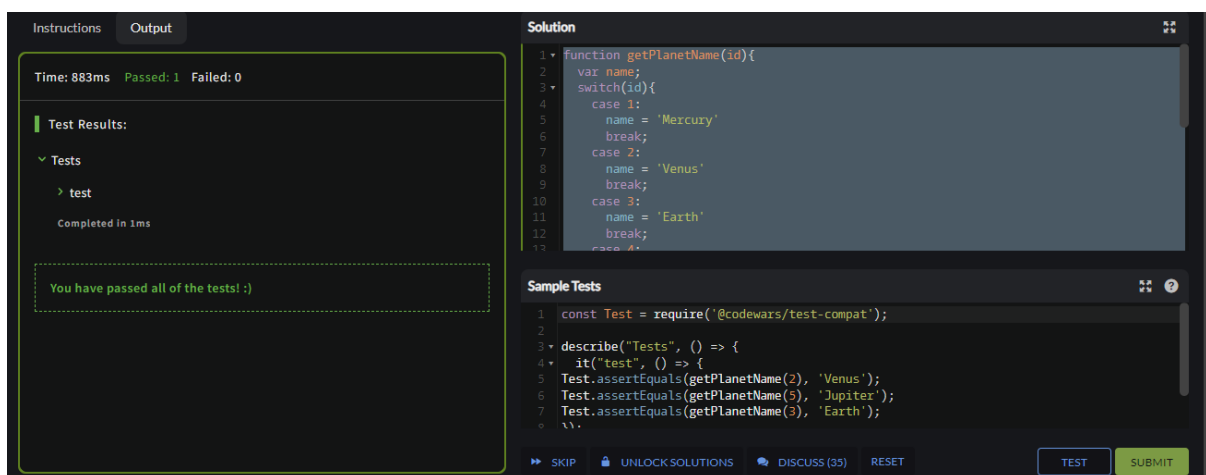
2. Get Planet Name By ID:

```
function getPlanetName(id){  
  var name;  
  switch(id){  
    case 1:  
      name = 'Mercury'  
      break;  
    case 2:  
      name = 'Venus'  
      break;  
    case 3:  
      name = 'Earth'  
      break;  
    case 4:  
      name = 'Mars'
```

```

    break;
case 5:
    name = 'Jupiter'
    break;
case 6:
    name = 'Saturn'
    break;
case 7:
    name = 'Uranus'
    break;
case 8:
    name = 'Neptune'
    break;
}
return name;
}

```



The screenshot shows a coding interface with three main panels. The left panel, titled 'Instructions', displays test results: 'Time: 883ms Passed: 1 Failed: 0' and a message 'You have passed all of the tests! :)'. The middle panel, titled 'Solution', shows the following JavaScript code:

```

1 function getPlanetName(id){
2   var name;
3   switch(id){
4     case 1:
5       name = 'Mercury'
6       break;
7     case 2:
8       name = 'Venus'
9       break;
10    case 3:
11      name = 'Earth'
12      break;
13    case 4:
14      name = 'Mars'
15      break;
16    case 5:
17      name = 'Jupiter'
18      break;
19    case 6:
20      name = 'Saturn'
21      break;
22    case 7:
23      name = 'Uranus'
24      break;
25    case 8:
26      name = 'Neptune'
27      break;
28  }
29  return name;
30 }

```

The right panel, titled 'Sample Tests', shows the following test code:

```

1 const Test = require('@codewars/test-compat');
2
3 describe("Tests", () => {
4   it("test", () => {
5     Test.assertEquals(getPlanetName(2), 'Venus');
6     Test.assertEquals(getPlanetName(5), 'Jupiter');
7     Test.assertEquals(getPlanetName(3), 'Earth');
8   })
9 })

```

At the bottom of the interface are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (35)', 'RESET', 'TEST', and 'SUBMIT'.

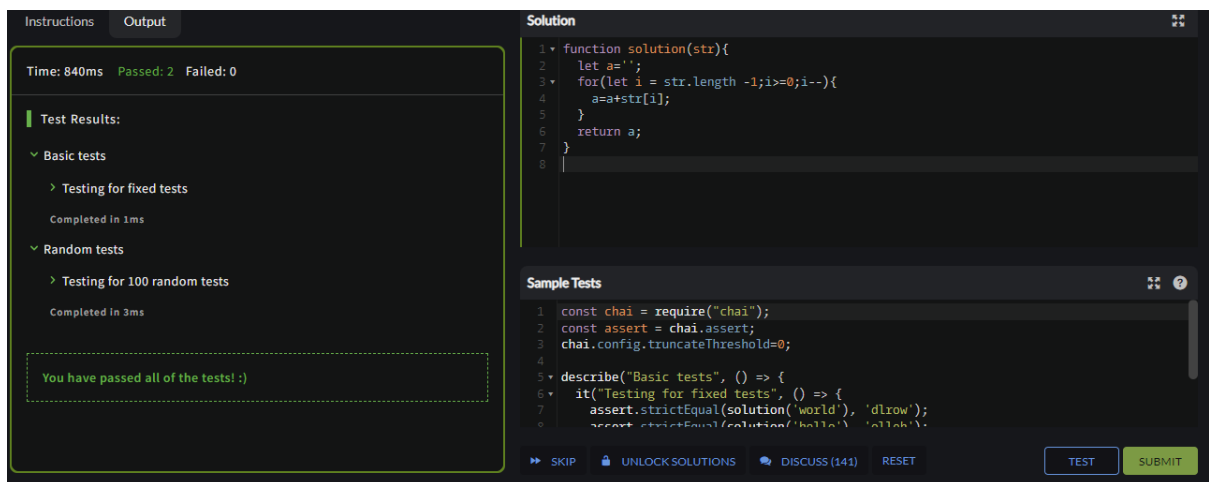
3. Reversed Strings

```
function solution(str){
```

```

let a="";
for(let i = str.length -1;i>=0;i--){
    a=a+str[i];
}
return a;
}

```



4. Even or Odd

```

function evenOrOdd(number) {
    if(number % 2 ==0){
        return "Even";
    }
    else{
        return "Odd"
    }
}

```

The screenshot shows a coding challenge interface. On the left, the 'Instructions' tab is active, displaying 'Time: 787ms', 'Passed: 45', and 'Failed: 0'. Below this, 'Test Results' are shown, including sample tests (2 is even, 7 is odd, -42 is even, -7 is odd, 0 is even) and random tests (Should return 'Even' for 822, Should return 'Odd' for -997). The 'Output' tab is also visible. On the right, the 'Solution' tab shows a JavaScript function:

```
function evenOrOdd(number) {
  if(number % 2 ==0){
    return "Even";
  }
  else{
    return "Odd"
  }
}
```

 A green message box says 'Great! You may take your time to refactor/comment your solution. Submit when ready.' Below the solution, 'Sample Tests' are shown with a Jest test:

```
const chai = require('chai');
const assert = chai.assert;

describe('Sample tests', () => {
  it('2 is even', () => {
    assert.strictEqual(evenOrOdd(2), "Even");
  });
});
```

 At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (225)', 'RESET', 'TEST', and 'SUBMIT'.

5. Counting sheep...

```
function countSheeps(sheep) {  
  let count=0;  
  for (let i=0;i<=sheep.length;i++){  
    if (sheep[i]==true){  
      count+=1  
    }  
  }  
  return count;  
}
```

The screenshot shows a coding challenge interface for 'countSheeps'. On the left, the 'Instructions' tab is active, displaying 'Time: 837ms', 'Passed: 2', and 'Failed: 0'. Below this, 'Test Results' are shown, including 'Submission Tests' (Fixed Tests, Random Tests) and a message 'You have passed all of the tests! :)'. The 'Output' tab is also visible. On the right, the 'Solution' tab shows a JavaScript function:

```
function countSheeps(sheep) {  
  // TODO  
  let count=0;  
  for (let i=0;i<=sheep.length;i++){  
    if (sheep[i]==true){  
      count+=1  
    }  
  }  
  return count;  
}
```

 A green message box says 'Outstanding! You may take your time to refactor/comment your solution. Submit when ready.' Below the solution, 'Sample Tests' are shown with a Jest test:

```
const {assert} = require('chai');  
  
describe('Sample Tests', function() {  
  const tests = [  
    [[], 0],  
    [[undefined], 0],  
    [[null], 0],  
  ];  
});
```

 At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (259)', 'RESET', 'TEST', and 'SUBMIT'.

6. Vowel Count

```
function getCount(str) {  
  let count=0;  
  for (let i=0;i<str.length;i++){  
    if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u'){  
      count+=1  
    }  
  }  
  return count;  
}
```

The screenshot displays a coding challenge interface. On the left, the 'Instructions' tab is active, showing test results for the 'Vowels Count Tests'. The results indicate that all tests passed, with a time of 740ms and 6 passed tests. The tests include: 'should return 5 for 'abracadabra'', 'should return 4 for 'pear tree'', 'should return 13 for 'o a kak ushakov lil vo kashu kakao'', 'should return 0 for 'my pyx'', 'should return 168 for a long input', and 'should return correct results for random tests'. A green message at the bottom states 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing the JavaScript code for the 'getCount' function. The code is identical to the one provided in the first block. Below the solution, there is a 'Sample Tests' section with a single test case: 'should return 5 for 'abracadabra'', which is passed. At the bottom of the interface, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (206)', 'RESET', 'TEST', and 'SUBMIT'.

Instructions Output

Time: 740ms Passed: 6 Failed: 0

Test Results:

Vowels Count Tests

- should return 5 for 'abracadabra'
- should return 4 for 'pear tree'
- should return 13 for 'o a kak ushakov lil vo kashu kakao'
- should return 0 for 'my pyx'
- should return 168 for a long input
- should return correct results for random tests

Completed in 3ms

You have passed all of the tests! :)

Solution

```
1 function getCount(str) {  
2   let count=0;  
3   for (let i=0;i<str.length;i++){  
4     if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u'){  
5       count+=1  
6     }  
7   }  
8   return count;  
9 }
```

Outstanding! You may take your time to refactor/comment your solution. Submit when ready.

Sample Tests

```
1 const {assert} = require("chai");  
2  
3 describe("Vowels Count Tests",function(){  
4   it("should return 5 for 'abracadabra'",function(){  
5     assert.strictEqual(getCount("abracadabra"), 5);  
6   });  
7 });
```

SKIP UNLOCK SOLUTIONS DISCUSS (206) RESET TEST SUBMIT

7. Jenny's secret message

```
function greet(name){  
  if(name === "Johnny"){  
    return "Hello, my love!";  
  }  
  else{  
    return "Hello, " + name + "!";  
  }  
}
```

Instructions

Output

Time: 834ms Passed: 2 Failed: 0

Test Results:

Jenny's greeting function

should greet some people normally

should greet Johnny a little bit more special

Completed in 2ms

You have passed all of the tests! :)

Solution

```

1 function greet(name){
2   if(name === "Johnny"){
3     return "Hello, my love!";
4   }
5   else{
6     return "Hello, " + name + "!";
7   }
8 }

```

Impressive! You may take your time to refactor/comment your solution. Submit when ready.

Sample Tests

```

1 const Test = require('@codewars/test-compat');
2
3 describe("Jenny's greeting function", function(){
4   it("should greet some people normally",function(){
5     Test.assertEquals(greet("Jim"), "Hello, Jim!");
6     Test.assertEquals(greet("Jane"), "Hello, Jane!");
7     Test.assertEquals(greet("Simon"), "Hello, Simon!");
8   })
9 })

```

SKIP

UNLOCK SOLUTIONS

DISCUSS (49)

RESET

TEST

SUBMIT

8. Is n divisible by x and y?

```

function isDivisible(n, x, y) {
  if(n%x==0 && n%y==0){
    return true;
  }
  else{
    return false;
  }
}

```

Instructions

Output

Time: 759ms Passed: 2 Failed: 0

Test Results:

Basic tests

Fixed Tests

Completed in 1ms

Random tests

Testing for 40 random tests

Completed in 1ms

You have passed all of the tests! :)

Solution

```

1 function isDivisible(n, x, y) {
2   if(n%x==0 && n%y==0){
3     return true;
4   }
5   else{
6     return false;
7   }
8 }

```

Correctamundo! You may take your time to refactor/comment your solution. Submit when ready.

Sample Tests

```

1 const chai = require("chai");
2 const assert = chai.assert;
3 chai.config.truncateThreshold=0;
4
5 describe("Basic tests", () => {
6   it("Fixed Tests", () => {
7     assert.strictEqual(isDivisible(3,3,4),false);
8     assert.strictEqual(isDivisible(12,3,4),true);
9   })
10 })

```

SKIP

UNLOCK SOLUTIONS

DISCUSS (121)

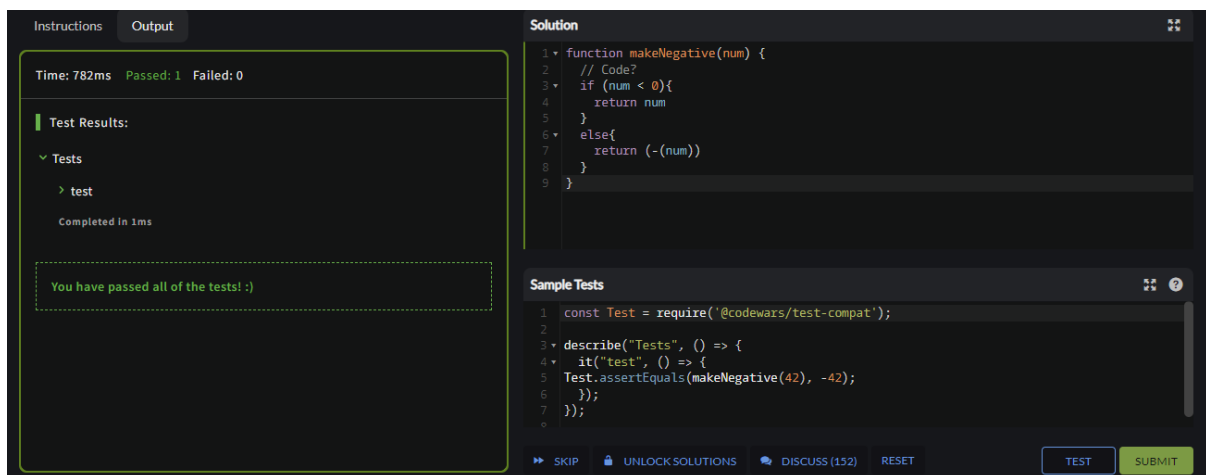
RESET

TEST

SUBMIT

9. Return Negative

```
function makeNegative(num) {  
  // Code?  
  if (num < 0){  
    return num  
  }  
  else{  
    return -(num)  
  }  
}
```



10. Find the smallest integer in the array

```
class SmallestIntegerFinder {  
  findSmallestInt(args) {  
    let small=999999999999999;  
    for (let i=0;i<args.length;i++){  
      if(args[i]<small){  
        small=args[i]  
      }  
    }  
  }  
}
```

```

    }

    return small;

}
}

```

The screenshot shows a coding challenge interface with three main sections: Instructions, Output, and Solution.

Instructions: Time: 728ms Passed: 2 Failed: 0. Test Results: Smallest Integer Tests. Fixed Tests. Random Tests. Completed in 17ms. A message states: "You have passed all of the tests! :)"

Solution: A JavaScript class `SmallestIntegerFinder` with a method `findSmallestInt(args)`. The method initializes `small` to `999999999999999`, then iterates through the `args` array, updating `small` to the minimum value found.

```

1 class SmallestIntegerFinder {
2   findSmallestInt(args) {
3     let small=999999999999999;
4     for (let i=0;i<args.length;i++){
5       if(args[i]<small){
6         small=args[i]
7       }
8     }
9     return small;
10  }
11 }

```

Sample Tests: A series of assertions testing the `findSmallestInt` method with various input arrays.

```

5 let sif = new SmallestIntegerFinder();
6
7 it("Fixed Tests", () => {
8   assert.strictEqual(sif.findSmallestInt([78,56,232,12,8]),8,'Should return the smallest integer');
9   assert.strictEqual(sif.findSmallestInt([78,56,232,12,18]),12,'Should return the smallest integer');
10  assert.strictEqual(sif.findSmallestInt([78,56,232,412,228]),56,'Should return the smallest integer');
11  assert.strictEqual(sif.findSmallestInt([78,56,232,12,0]),0,'Should return the smallest integer');
12  assert.strictEqual(sif.findSmallestInt([1,56,232,12,8]),1,'Should return the smallest integer');
13 })

```

Buttons at the bottom: SKIP, UNLOCK SOLUTIONS, DISCUSS (156), RESET, TEST, SUBMIT.

11. Grasshopper – Summation

```

var summation = function (num) {

  // Code here

  let sum=0;

  for (let i=1;i<=num;i++){

    sum+=i;

  }

  return sum;

}

```


The screenshot shows a Codecademy challenge interface. On the left, the 'Instructions' tab is active, displaying 'Time: 782ms Passed: 2 Failed: 0'. Below this, the 'Test Results' section shows a 'summation' test with two sub-tests: 'should return the correct total' and 'random numbers test'. A green message box states 'You have passed all of the tests! :)'. On the right, the 'Solution' tab shows the following JavaScript code:

```
1 var summation = function (num) {  
2   // Code here  
3   let sum=0;  
4   for (let i=1;i<=num;i++){  
5     sum+=i;  
6   }  
7   return sum;  
8 }
```

Below the solution, the 'Sample Tests' section shows the following code:

```
1 const assert = require('chai').assert;  
2  
3 describe('summation', function () {  
4   it('should return the correct total', function () {  
5     assert.strictEqual(summation(1), 1);  
6     assert.strictEqual(summation(2), 3);  
7     assert.strictEqual(summation(8), 36);  
8   })  
9 })
```

At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (107)', 'RESET', 'TEST', and 'SUBMIT'.

12. Get the mean of an array

```
function getAverage(marks){
```

```
  //TODO : calculate the downward rounded average of the marks array
```

```
  let sum=0;
```

```
  let len=marks.length;
```

```
  for(let i=0;i<len;i++){
```

```
    sum=sum+marks[i];
```

```
  }
```

```
  let avg=sum/len;
```

```
  return Math.floor(avg);
```

```
}
```

The screenshot shows a Codecademy challenge interface. On the left, the 'Instructions' tab is active, displaying 'Time: 756ms Passed: 2 Failed: 0'. Below this, the 'Test Results' section shows a 'Test with pre-create arrays' test with two sub-tests: 'should return the good average' and 'Should work with randomly created arrays'. A green message box states 'You have passed all of the tests! :)'. On the right, the 'Solution' tab shows the following JavaScript code:

```
1 function getAverage(marks){  
2   //TODO : calculate the downward rounded average of the marks array  
3   let sum=0;  
4   let len=marks.length;  
5   for(let i=0;i<len;i++){  
6     sum=sum+marks[i];  
7   }  
8   let avg=sum/len;  
9   return Math.floor(avg);  
10 }
```

Below the solution, the 'Sample Tests' section shows the following code:

```
1 const Test = require('@codewars/test-compat');  
2  
3 describe('Tests', () => {  
4   it('test', () => {  
5     Test.assertEquals(getAverage([2,2,2]),2);  
6     Test.assertEquals(getAverage([1,2,3,4,5]),3);  
7     Test.assertEquals(getAverage([1,1,1,1,1,1,2]),1);  
8   })  
9 })
```

At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (51)', 'RESET', 'TEST', and 'SUBMIT'.

13. Rock Paper Scissors!

```
const rps = (p1, p2) => {  
  if(p1=='scissors' && p2=='paper' || p1=='rock' && p2=='scissors' ||  
  p1=='paper' && p2=='rock'){  
    return "Player 1 won!";  
  }  
  
  else if (p1=='scissors' && p2=='rock' || p1=='paper' && p2=='scissors' ||  
  p1=='rock' && p2=='paper'){  
    return "Player 2 won!";  
  }  
  
  else{  
    return "Draw!"  
  }  
};
```

The screenshot shows a coding challenge interface. On the left, the 'Instructions' tab is active, displaying 'Time: 974ms Passed: 4 Failed: 0'. Below this, the 'Test Results' section shows a tree view for 'rock paper scissors' with sub-items 'player 1 win', 'player 2 win', 'draw', and 'random'. A message at the bottom states 'You have passed all of the tests! :)'.

On the right, the 'Solution' tab is active, showing the following code:

```
1 • const rps = (p1, p2) => {  
2 •   if(p1=='scissors' && p2=='paper' || p1=='rock' && p2=='scissors' || p1=='paper' && p2=  
3 •     return "Player 1 won!";  
4 •   }  
5 •   else if (p1=='scissors' && p2=='rock' || p1=='paper' && p2=='scissors' || p1=='rock' &  
6 •     return "Player 2 won!";  
7 •   }  
8 •   else{  
9 •     return "Draw!"  
10 •   }  
11 • };
```

Below the solution code, the 'Sample Tests' section shows two test cases:

```
12 • it('player 2 win', function() {  
13 •   Test.assertEquals(rps('scissors', 'rock'), getMsg(2));  
14 •   Test.assertEquals(rps('paper', 'scissors'), getMsg(2));  
15 •   Test.assertEquals(rps('rock', 'paper'), getMsg(2));  
16 • });  
17 •  
18 • it('draw', function() {  
19 •   Test.assertEquals(rps('rock', 'rock'), 'Draw!');
```

At the bottom of the interface, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (77)', 'RESET', 'TEST', and 'SUBMIT'.

14. Remove First and Last Character

```
function removeChar(str){  
  str = str.slice(1, -1);  
  
  return str;  
};
```

Instructions

Output

Time: 782ms Passed: 2 Failed: 0

Test Results:

Tests

Fixed Tests

Random Tests

Testing for 100 random tests

Completed in 2ms

You have passed all of the tests! :)

Solution

```

1 function removeChar(str){
2   //You got this!
3   str = str.slice(1, -1);
4   return str;
5 }
6
7
8
9

```

Sample Tests

```

1 const chai = require("chai");
2 const assert = chai.assert;
3 chai.config.truncateThreshold=0;
4
5 describe("Tests", () => {
6   it("Fixed Tests", () => {
7     assert.strictEqual(removeChar('eloquent'), 'loquen');
8     assert.strictEqual(removeChar('country'), 'ountry');
9   });
10 })

```

SKIP

UNLOCK SOLUTIONS

DISCUSS (164)

RESET

TEST

SUBMIT

15. Sum of positive

```

function positiveSum(arr) {
  let sum=0;
  for(let i=0;i<arr.length;i++){
    if (arr[i]>0){
      sum+=arr[i]
    }
  }
  return sum;
}

```

Instructions

Output

Time: 789ms Passed: 101 Failed: 0

Test Results:

Basic tests

Testing for fixed tests

Completed in 1ms

Random tests

positiveSum([14,89,29,-86,-99,21,64,-45,-99,-81,46,96,57,-53,-11,85,-60,-50,-69,94,85,-41,16,36,95,21,25,-18,42,-93,62,-27,-53,21,-19,-82,-6,-75,-3,-18,58,-19,-62,29,44,72,54,-51,-54,-62,3,5,-30,-40,30,-50,36,-8,-35,80,-37,8,-30,-31,-47,-48,-78,81,-7

Solution

```

1 function positiveSum(arr) {
2   let sum=0;
3   for(let i=0;i<arr.length;i++){
4     if (arr[i]>0){
5       sum+=arr[i]
6     }
7   }
8   return sum;
9 }

```

Sample Tests

```

1 const { assert } = require("chai")
2
3 describe("Basic tests", () => {
4   it("Testing for fixed tests", () => {
5     assert.strictEqual(positiveSum([1,2,3,4,5]),15);
6     assert.strictEqual(positiveSum([1,-2,3,4,5]),13);
7     assert.strictEqual(positiveSum([1]),1);
8     assert.strictEqual(positiveSum([1, 2, 3, 4, 5]), 15);
9   });
10 })

```

SKIP

UNLOCK SOLUTIONS

DISCUSS (128)

RESET

TEST

SUBMIT

16. Basic Mathematical Operations

```
function basicOp(operation, value1, value2)
```

```
{
```

```
  if (operation=='+'){
```

```
    return (value1+value2);
```

```
  }
```

```
  if (operation=='-'){
```

```
    return (value1-value2);
```

```
  }
```

```
  if (operation=='*'){
```

```
    return (value1*value2);
```

```
  }
```

```
  if (operation=='/'){
```

```
    return (value1/value2);
```

```
  }
```

```
}
```

The screenshot displays a coding challenge interface with a dark theme. On the left, the 'Instructions' tab is active, showing test results: 'Time: 760ms', 'Passed: 1', 'Failed: 0'. A 'Test Results' section shows a single test 'test' completed in 4ms. A message states 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing the following code:

```
1 function basicOp(operation, value1, value2)
2 {
3   // Code
4   if (operation=='+'){
5     return (value1+value2);
6   }
7
8   if (operation=='-'){
9     return (value1-value2);
10  }
11
12  if (operation=='*'){
13    return (value1*value2);
14  }
15
16  if (operation=='/'){
17    return (value1/value2);
18  }
19 }
```

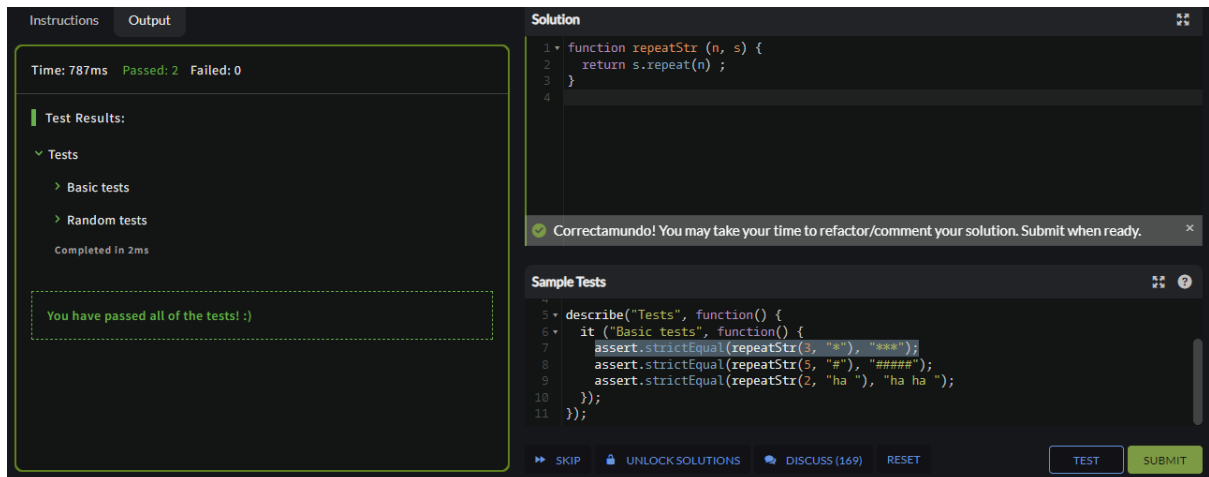
Below the solution, the 'Sample Tests' tab is active, showing test cases:

```
1 const Test = require("@codewars/test-compat");
2
3 describe("Tests", () => {
4   it("test", () => {
5     console.log("Basic tests\n");
6     Test.assertSimilar(basicOp('+', 4, 7), 11);
7     Test.assertSimilar(basicOp('-', 15, 10), -3);
8     Test.assertSimilar(basicOp('*', 5, 6), 30);
9     Test.assertSimilar(basicOp('/', 15, 3), 5);
10  });
11 })
```

At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (81)', 'RESET', 'TEST', and 'SUBMIT'.

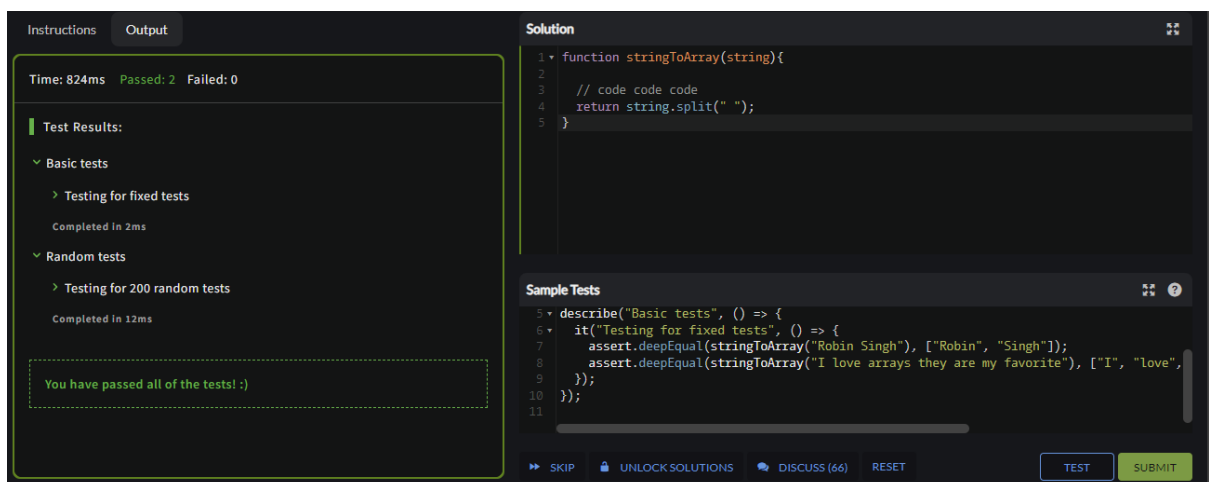
17. String repeat

```
function repeatStr (n, s) {  
  return s.repeat(n);  
}
```



18. Convert a string to an array

```
function stringToArray(string){  
  return string.split(" ");  
}
```



19. Remove String Spaces

```
function noSpace(x){
```

```

return x.replace(/ /g, "");
}

```

The screenshot shows a coding challenge interface with a dark theme. On the left, the 'Instructions' tab is active, displaying 'Time: 774ms', 'Passed: 2', and 'Failed: 0'. Below this, 'Test Results' are shown, including 'Basic tests' (Fixed Tests and Random Tests) and a message: 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing a JavaScript function: `function noSpace(x){ return x.replace(/ /g, ""); }`. Below the solution, 'Sample Tests' are listed with various inputs and expected outputs. At the bottom right, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (102)', 'RESET', 'TEST', and 'SUBMIT'.

20. Beginner - Lost Without a Map

```

function maps(x){
  let a= []
  for(let i=0;i<x.length;i++){
    a.push(x[i]*2);
  }
  return a;
}

```

The screenshot shows a coding challenge interface with a dark theme. On the left, the 'Instructions' tab is active, displaying 'Time: 762ms', 'Passed: 3', and 'Failed: 0'. Below this, 'Test Results' are shown, including 'Basic tests' (Fixed tests, Sanity test, and Random tests) and a message: 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing a JavaScript function: `function maps(x){ let a= []; for(let i=0;i<x.length;i++){ a.push(x[i]*2); } return a; }`. Below the solution, 'Sample Tests' are listed with inputs like `[1, 2, 3]` and expected outputs like `[2, 4, 6]`. At the bottom right, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (92)', 'RESET', 'TEST', and 'SUBMIT'.

21. Is he gonna survive?

```
function hero(bullets, dragons){  
  //Get Coding!  
  if((dragons*2)<=bullets){  
    return true;  
  }  
  else{  
    return false;  
  }  
}
```

The screenshot shows a coding challenge interface. On the left, the 'Output' tab is active, displaying test results: 'Time: 829ms', 'Passed: 2', 'Failed: 0'. Below this, 'Test Results' are shown, including 'Fixed tests' and 'Random tests', both completed. A green dashed box contains the message 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing the JavaScript code for the 'hero' function. Below the solution, 'Sample Tests' are listed, including a 'chai.config.truncateThreshold=0;' line and a 'describe' block for 'Fixed tests' with three 'it' blocks for testing. At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (77)', 'RESET', 'TEST', and 'SUBMIT'.

22. Array plus array

```
function arrayPlusArray(arr1, arr2) {  
  let sum1 = 0;  
  let sum2 = 0;  
  for (let i = 0; i < arr1.length; i++) {  
    sum1 += arr1[i];  
  }  
}
```

```

for (let i = 0; i < arr2.length; i++) {
    sum2 += arr2[i];
}

return sum1 + sum2;
}

```

The screenshot shows a coding challenge interface with a dark theme. On the left, the 'Instructions' tab is active, displaying 'Time: 737ms', 'Passed: 2', and 'Failed: 0'. Below this, 'Test Results' are shown, including 'Basic tests' and 'Random tests', both of which are completed. A green message box states 'You have passed all of the tests! :)'.

On the right, the 'Solution' tab is active, showing a JavaScript function `arrayPlusArray` that iterates through two arrays and returns their sum. Below the solution, a green message says 'Impressive! You may take your time to refactor/comment your solution. Submit when ready.'.

At the bottom, the 'Sample Tests' tab is active, showing a list of test cases with their expected results. The bottom bar contains buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (45)', 'RESET', 'TEST', and 'SUBMIT'.

23. Century From Year

```

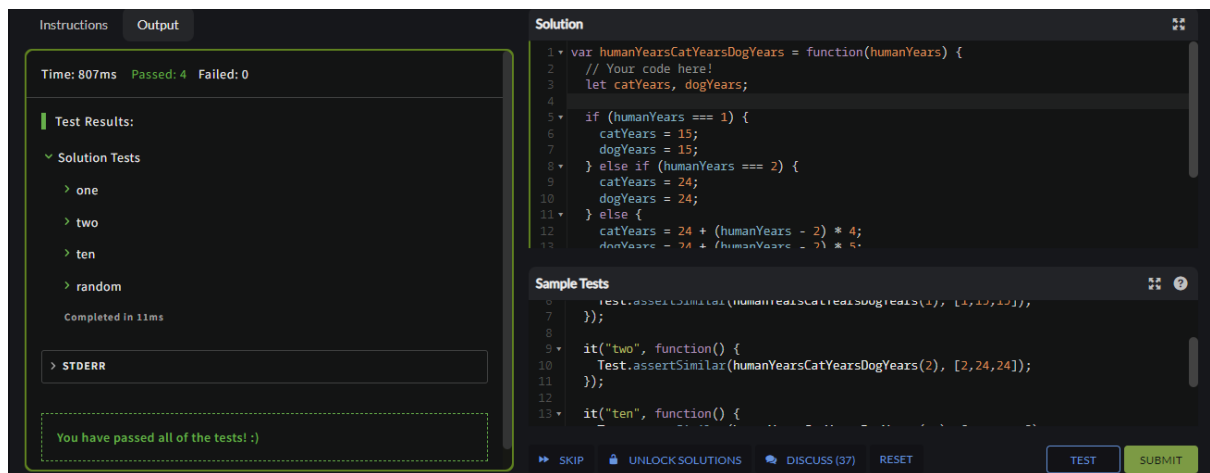
function century(year) {
    // Finish this :)
    if ((year % 100) == 0) {
        return (year / 100);
    }
    else {
        return Math.ceil(year / 100);
    }
}

```


The screenshot shows a Codewars challenge interface. On the left, the 'Instructions' tab is active, displaying 'Time: 761ms', 'Passed: 2', and 'Failed: 0'. Below this, the 'Test Results' section shows 'Basic Tests' expanded, with 'Sample Tests' and 'Random Tests' listed. A green message box states 'You have passed all of the tests! :)'. On the right, the 'Solution' tab is active, showing a JavaScript function `century(year)` that calculates the century from a year. The function uses a ternary operator to check if the year is a multiple of 100, returning `year/100` if true, and `Math.ceil(year/100)` otherwise. Below the solution, the 'Sample Tests' section shows a list of test cases for the `century` function, including years 1705, 1900, 1601, and 2000. At the bottom, there are buttons for 'SKIP', 'UNLOCK SOLUTIONS', 'DISCUSS (150)', 'RESET', 'TEST', and 'SUBMIT'.

24. Cat years, Dog years

```
var humanYearsCatYearsDogYears = function(humanYears) {  
  // Your code here!  
  let catYears, dogYears;  
  if (humanYears === 1) {  
    catYears = 15;  
    dogYears = 15;  
  } else if (humanYears === 2) {  
    catYears = 24;  
    dogYears = 24;  
  } else {  
    catYears = 24 + (humanYears - 2) * 4;  
    dogYears = 24 + (humanYears - 2) * 5;  
  }  
  return [humanYears, catYears, dogYears];  
}
```



25. Total amount of points

```
function points(games) {
  let total=0
  for(let i=0;i<games.length;i++){
    let [a,b]=games[i].split(':');
    if (a>b){
      total+=3;
    }
    else if (a==b){
      total+=1;
    }
  }
  return total;
}
```

