CW01- Create a function named isPrime that takes an integer n > 1 and returns an string with “number is integer” or “number is not integer”.

CW02 - Create a function named divisors/Divisors that takes an integer n > 1 and returns an array with all of the integer's divisors(except for 1 and the number itself), from smallest to largest. If the number is prime return the string '(integer) is prime' (null in C#) (use Either String a in Haskell and Result<Vec<u32>, String> in Rust).

**Example:**

divisors(12); // => [2, 3, 4, 6]

divisors(25); // => [5]

divisors(13); // => '13 is prime'

Solution:  
function divisors($integer) {

}

Sample test:  
class DivisorsTest extends TestCase {

public function testExamples() {

$this->assertEquals([3, 5], divisors(15));

$this->assertEquals([2, 3, 4, 6], divisors(12));

$this->assertEquals('13 is prime', divisors(13));

}

}

CW03-Given the triangle of consecutive odd numbers:

1

3 5

7 9 11

13 15 17 19

21 23 25 27 29

...

Calculate the sum of the numbers in the nth row of this triangle (starting at index 1) e.g.: (**Input --> Output**)

1 --> 1

2 --> 3 + 5 = 8

Sol:

function rowSumOddNumbers($n) {

}

class MyTestCases extends TestCase {

public function testFixedTests() {

$this->assertEquals(1, rowSumOddNumbers(1));

$this->assertEquals(8, rowSumOddNumbers(2));

$this->assertEquals(2197, rowSumOddNumbers(13));

$this->assertEquals(6859, rowSumOddNumbers(19));

$this->assertEquals(68921, rowSumOddNumbers(41));

$this->assertEquals(74088, rowSumOddNumbers(42));

$this->assertEquals(405224, rowSumOddNumbers(74));

$this->assertEquals(636056, rowSumOddNumbers(86));

$this->assertEquals(804357, rowSumOddNumbers(93));

$this->assertEquals(1030301, rowSumOddNumbers(101));

}

}

P04 – arraydiff

Your goal in this kata is to implement a difference function, which subtracts one list from another and returns the result.

It should remove all values from list a, which are present in list b keeping their order.

arrayDiff([1,2],[1]) == [2]

If a value is present in b, all of its occurrences must be removed from the other:

arrayDiff([1,2,2,2,3],[2]) == [1,3]

sol:

function arrayDiff($a, $b) {

}

Sample tests:  
<?php use PHPUnit\Framework\TestCase;

class MyTestCasesTest extends TestCase

{

public function testSampleTests() {

$this->assertEquals([2], arrayDiff([1,2], [1]), "a was [1,2], b was [1], expected [2]");

$this->assertEquals([2,2], arrayDiff([1,2,2], [1]), "a was [1,2,2], b was [1], expected [2,2]");

$this->assertEquals([1], arrayDiff([1,2,2], [2]), "a was [1,2,2], b was [2], expected [1]");

$this->assertEquals([1,2,2], arrayDiff([1,2,2], []), "a was [1,2,2], b was [], expected [1,2,2]");

$this->assertEquals([], arrayDiff([], [1,2]), "a was [], b was [1,2], expected []");

$this->assertEquals([3], arrayDiff([1, 2, 3], [1,2]), "a was [1, 2, 3], b was [1,2], expected [3]");

}

}

C05

Complete the function that accepts a string parameter, and reverses each word in the string. **All** spaces in the string should be retained.

## Examples

"This is an example!" ==> "sihT si na !elpmaxe"

"double spaces" ==> "elbuod secaps"

Sol:

function reverseWords($str) {

}

Sample:

public function testBasicTests() {

$this->assertEquals('ehT kciuq nworb xof spmuj revo eht yzal .god', reverseWords('The quick brown fox jumps over the lazy dog.'));

$this->assertEquals('elppa', reverseWords('apple'));

$this->assertEquals('a b c d', reverseWords('a b c d'));

$this->assertEquals('elbuod decaps sdrow', reverseWords('double spaced words'));

$this->assertEquals('desserts stressed', reverseWords('stressed desserts'));

$this->assertEquals('olleh olleh', reverseWords('hello hello'));

}

}