

EXERCISE 0

Repository

`git@github.com:lmuzinic/phplonghorn-pragmatic-tdd.git`

With Docker

```
git clone <repository>  
cd phplonghorn-pragmatic-tdd  
make  
make test
```

OK (2 tests, 2 assertions)

With PHP 7.1+

```
git clone <repository>  
cd phplonghorn-pragmatic-tdd/app  
composer install  
vendor/bin/phpunit
```

OK (2 tests, 2 assertions)

Alpine image, vendor dir ~10 MB, you can use your mobile data if WIFI does not work.



PRAGMATIC TDD

HELLO



Luka Muzinic
@lmuzinic

WORKSHOP RULES

ASK QUESTIONS

IF YOU STILL DO NOT UNDERSTAND, ASK QUESTIONS AGAIN

DISCUSS RIGHT NOW, DO NOT WAIT FOR THE “RIGHT MOMENT”

WHY WE NEED TESTING?

EXIT

EXERCISE 0

Repository
`github.com:leejohn/phyloghorn-pragmatic-tdd.git`

With Docker

```
git clone repository
cd phyloghorn-pragmatic-tdd
make test
```

OK (2 tests, 3 assertions)

With PHP 7.3+

```
git clone repository
cd phyloghorn-pragmatic-tdd/app
composer install
vendor/bin/phpunit
```

OK (2 tests, 3 assertions)

Alpine image, vendor dir ~10 MB, you can use your mobile data if WIFI does not work.

**WHY AM I
HERE?**

**ARE WE
SOFTWARE TESTERS?**

**AND YET
WE KEEP ON SAYING
WE WRITE TESTS...**

**WHERE CAN I GET
MORE OF THOSE
TESTS?**



EXCUSES, EXCUSES

TESTS SLOW US DOWN

WE WILL NEVER GET TO 100% CODE COVERAGE

WE DO NOT HAVE TIME TO LEARN TESTING, WE'RE TOO BUSY SHIPPING CODE

EXCUSES, EXCUSES

TESTS SLOW US DOWN?

STOP TESTFILEING

DO YOU OFTEN DO THIS?

~ `php test.php`

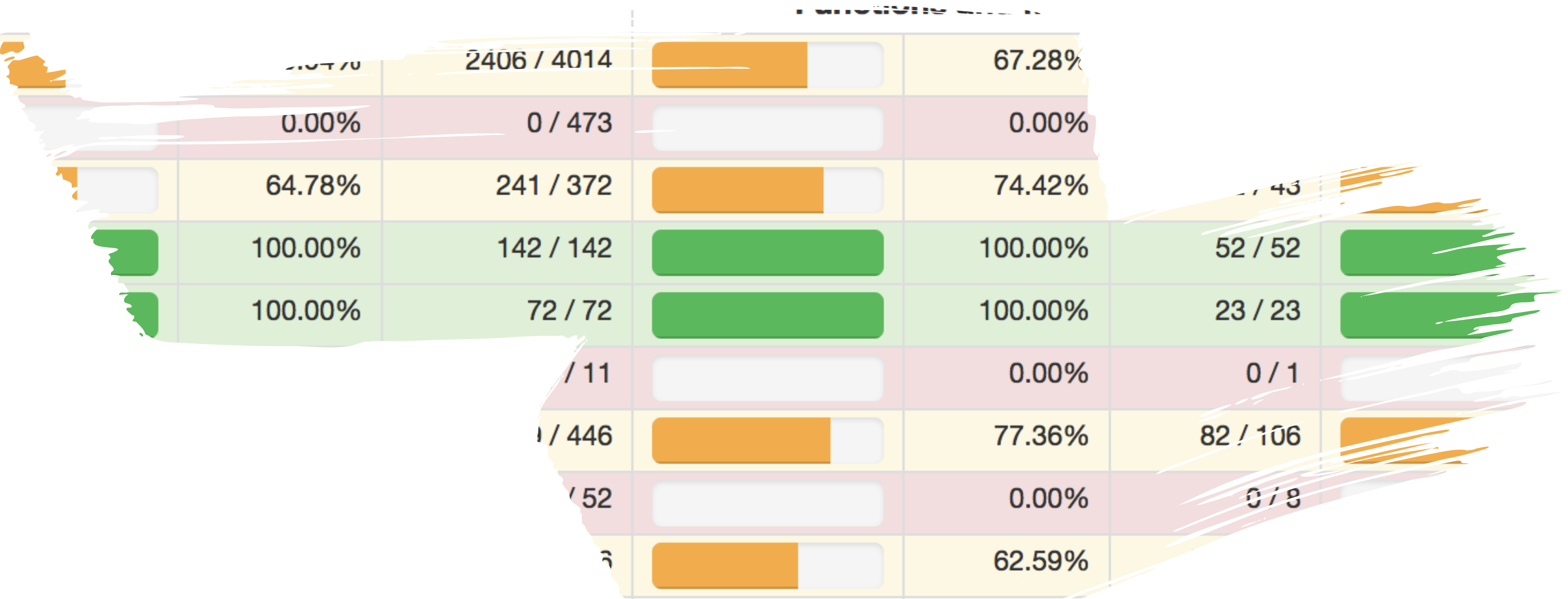
OR THIS

`http://localhost/test_problem.php`

EXCUSES, EXCUSES

**WE WILL NEVER GET TO 100%
CODE COVERAGE**

CODE COVERAGE






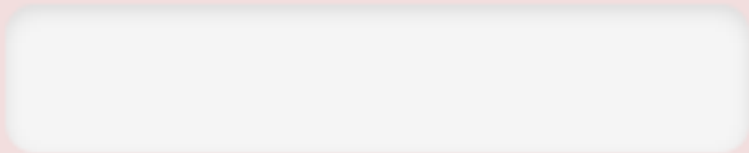
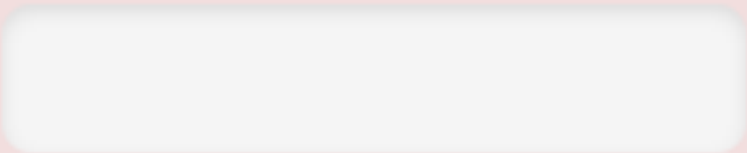
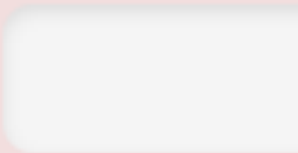















CODE COVERAGE

Classes and Traits		
<div><div></div></div>	56.88%	62 / 109
<div><div></div></div>	0.00%	0 / 8
<div><div></div></div>	64.29%	9 / 14
<div><div></div></div>	100.00%	9 / 9
<div><div></div></div>	100.00%	5 / 5
<div><div></div></div>	76.47%	13 / 17
<div><div></div></div>	48.15%	26 / 54
	n/a	0 / 0








CODE COVERAGE

Code Coverage					
Functions and Methods			Classes and Traits		
<div><div></div></div>	67.28%	368 / 547	<div><div></div></div>	56.88%	62 / 109
<div><div></div></div>	0.00%	0 / 28	<div><div></div></div>	0.00%	0 / 8
<div><div></div></div>	74.42%	32 / 43	<div><div></div></div>	64.29%	9 / 14
<div><div></div></div>	100.00%	52 / 52	<div><div></div></div>	100.00%	9 / 9
<div><div></div></div>	100.00%	23 / 23	<div><div></div></div>	100.00%	5 / 5
<div><div></div></div>	77.36%	82 / 106	<div><div></div></div>	76.47%	13 / 17
<div><div></div></div>	62.59%	179 / 286	<div><div></div></div>	48.15%	26 / 54
	n/a	0 / 0		n/a	0 / 0

CODE COVERAGE

Code Coverage						
Lines			Functions and Methods			
	59.94%	2406 / 4014		67.28%	368 / 547	
	0.00%	0 / 473		0.00%	0 / 28	
	64.78%	241 / 372		74.42%	32 / 43	
	100.00%	142 / 142		100.00%	52 / 52	
	100.00%	72 / 72		100.00%	23 / 23	
	67.04%	299 / 446		77.36%	82 / 106	
	67.54%	1652 / 2446		62.59%	179 / 286	
	n/a	0 / 0		n/a	0 / 0	

20 > 80

	Code Coverage				Cod
		Lines		Function	
Total	<div><div></div></div>	59.94%	2406 / 4014	<div><div></div></div>	
 Command	<div><div></div></div>	0.00%	0 / 473	<div><div></div></div>	
 Controller	<div><div></div></div>	64.78%	241 / 372	<div><div></div></div>	
 Entity	<div><div></div></div>	100.00%	142 / 142	<div><div></div></div>	
 Model	<div><div></div></div>	100.00%	72 / 72	<div><div></div></div>	
 Repository	<div><div></div></div>	67.04%	299 / 446	<div><div></div></div>	
 Service	<div><div></div></div>	67.54%	1652 / 2446	<div><div></div></div>	
 AppBundle.php		n/a	0 / 0		

EXCUSES, EXCUSES

**WE DO NOT HAVE TIME TO LEARN TESTING,
WE'RE TOO BUSY SHIPPING CODE**

//@TODO: STANDSTILL

DON'T BE SCARED OF PHPUNIT*

IT IS JUST A CODE RUNNER

UNIT, INTEGRATION OR ACCEPTANCE TESTS

SMOKE TESTS

WEBSITE SCRAPER

...

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

HOW TO WRITE TESTS?

GIVEN WHEN THEN

SETUP EXERCISE VERIFY TEARDOWN

TEST CASE ANATOMY

```
class TakeMeOutTest extends TestCase
{
    private $favorites;
    private $rivals;

    public function setUp(): void
    {
        $this->favorites = Team::create('Houston Astros');
        $this->rivals = Team::create('Texas Rangers');
    }

    public function testFavoriteTeam()
    {
        $this->assertSame('Houston Astros', $this->favorites->getName());
    }

    public function testNotSoFavoriteTeam()
    {
        $this->assertSame('Texas Rangers', $this->rivals->getName());
    }
}
```

WORKSHOP RULES

ASK QUESTIONS

IF YOU STILL DO NOT UNDERSTAND, ASK QUESTIONS AGAIN

DISCUSS RIGHT NOW, DO NOT WAIT FOR THE “RIGHT MOMENT”



EXERCISE 0

Repository

`git@github.com:lmuzinic/phplonghorn-pragmatic-tdd.git`

<https://github.com/lmuzinic/phplonghorn-pragmatic-tdd>

With Docker

```
git clone <repository>  
cd phplonghorn-pragmatic-tdd  
make  
make test
```

OK (2 tests, 2 assertions)

With PHP 7.1+

```
git clone <repository>  
cd phplonghorn-pragmatic-tdd/app  
composer install  
vendor/bin/php
```

OK (2 tests, 2 assertions)

EXERCISE 1

BINARY GAP

https://app.codility.com/programmers/lessons/1-iterations/binary_gap/

Find longest sequence of zeros in binary representation of an integer.

A binary gap within a positive integer N is any maximal sequence of consecutive zeros that is surrounded by ones at both ends in the binary representation of N .

How many tests we need to write?

EXERCISE 1 cont.

BINARY GAP

For example, number 9 has binary representation 1001 and contains a binary gap of length 2. The number 529 has binary representation 1000010001 and contains two binary gaps: one of length 4 and one of length 3. The number 20 has binary representation 10100 and contains one binary gap of length 1. The number 15 has binary representation 1111 and has no binary gaps. The number 32 has binary representation 100000 and has no binary gaps.

Reuse existing `TakeMeOutTest.php` to add more test cases.

SO FAR SO GOOD?

THIS IS EXTREMELY SIMPLIFIED EXAMPLE

- our applications are a bit bigger and a bit more complicated

BASEBALL

SOMEWHAT SIMPLIFIED

- two teams, Home team and Away team play a Match
- at the end of the game, each team holds a Score
 - score consists of Runs, Hits and Errors
- winner of the Match is a team that scored more Runs
 - winner gets recorded with a Win
 - loser gets recorded with a Loss
- repeat for desired number of times
- best team is one with biggest Percentage (Win/Total number of Matches)

MLB · Today

Final



Texas Rangers
(10 - 8)

2

-

7



Houston Astros
(13 - 6)

Team	1	2	3	4	5	6	7	8	9	R	H	E
Houston Astros	3	1	0	0	3	0	0	0	0	7	11	1
Texas Rangers	0	0	0	0	0	1	0	0	1	2	4	2



Game recap



RANGERS

ASTROS

NEWS

Batting

Player	AB	R	H	BB	RBI
17 Shin-Soo Choo · DH	3	0	0	1	0
38 Danny Santana · 2B	4	1	1	0	1
1 Elvis Andrus · SS	4	0	0	0	0
30 Nomar Mazara · RF	4	0	0	0	0






Season

2019 ▾

American League

National League

AL West

Team	W	L	Pct	GB	Home	Away	L10
 Astros	13	6	.684	-	6-0	7-6	9-1
 Mariners	15	8	.652	-	5-7	10-1	4-6
 Rangers	10	8	.556	2.5	8-4	2-4	5-5
 Athletics	11	11	.500	3.5	7-6	4-5	5-5
 Angels	8	12	.400	5.5	6-3	2-9	4-6

DISCUSSION

IMPLEMENT A BASEBALL LEAGUE MANAGEMENT APP

- We want to display standings table on our website

WHAT IS YOUR BIGGEST CONCERN?

EXERCISE 2

START IMPLEMENTING STANDINGS

Write a test for getting sorted standings

Talk about domain

Implementation

EXERCISE 2 cont.

START IMPLEMENTING STANDINGS

Create `tests/Domain/League/LeagueTest.php`

EXERCISE 3

IMPLEMENT TEAM POSITION

An object that will hold position inside the league table

Focus just on this class, use `--filter`

EXERCISE 3 cont.

IMPLEMENT TEAM POSITION

Create

`tests/Domain/Team/PositionTest.php`

Possible test cases

`testGetPercentageWithNoGames`

`testGetPercentageAfterRecordedWin`

`testGetPercentageAfterRecordedWinAndLoss`

...

GO TO VACATION

WHO WILL IMPLEMENT STANDINGS?

Your colleagues that stayed in the office.

Checkout branch [step-2](#)

EXERCISE 4

REFACTOR

Move checking, creating and returning a team Position into separate method

Move formatting into League

Move storing matches to repository

SO FAR SO GOOD?

REPOSITORIES ARE USUALLY SLOW

They make a costly database call.

Add `sleep(1)` to each method and run the test suite again.

EXERCISE 5

MOCK THE REPOSITORY INSIDE LeagueTest.php

```
$this->matchRepository = $this  
    ->getMockBuilder(MatchRepository::class)  
    ->disableOriginalConstructor()  
    ->getMock();
```

INTRODUCE STUBS

```
$this->matchRepository->method('findAll')->willReturn([$match]);
```

INTRODUCE SPIES

```
$this->matchRepository->expects($this->atLeastOnce())->method('save');
```

SPLIT INTO SEPARATE TEST CLASSES OR METHODS

SO FAR SO GOOD?

THERE WERE FEW BUGS

- someone entered a Team with no name
- someone created a Match between same Team

EXERCISE 6

EXCEPTIONS

- Match between same Team
- Team with empty name

EXERCISE 6 cont.

EXCEPTIONS

- Match between same Team
- Team with empty name

Introduce

```
src/Domain/Exception/BallGameException.php  
src/Domain/Exception/MatchBetweenSameTeamException.php  
src/Domain/Exception/TeamWithEmptyNameException.php
```

Test with

```
@expectedException \BallGame\Domain\Exception\BallGameException  
@expectedException \BallGame\Domain\Exception\MatchBetweenSameTeamException  
@expectedException \BallGame\Domain\Exception\TeamWithEmptyNameException
```

EXERCISE 7

CODECOVERAGE & CRAP

Run codecoverage

~ make test

~ vendor/bin/phpunit --coverage-html var/coverage

Explain CRAP

$CRAP = CC^2 \times U^3 + CC$

SO FAR SO GOOD?

THERE WAS THIS ONE EDGE CASE

- two teams were tied at the end of the season, they had an equal percentage and our system could not decide who should be first
- we should do something about it for next season

EXERCISE 8

NEXT YEAR FEATURES

League manager wants to change the rules for scoring – teams that have equal percentage, sort by wins (more wins moves up). Keep the possibility to show scoring for last year as well.

EXERCISE 8 cont.

SIMPLE RULEBOOK

Introduce

`src/Domain/RuleBook/RuleBookInterface.php`

Introduce method

`decide(TeamPosition $teamA, TeamPosition $teamB)`

Extract sorting logic into separate class, cover all cases with tests.

`tests/Domain/RuleBook/SimpleRuleBookTest.php`

Test cases

`testDecideReturnsLessThanZeroWhenFirstTeamHasMorePointsThanSecond`

`testDecideReturnsGreaterThanZeroWhenSecondTeamHasMorePointsThanFirst`

`testDecideReturnsZeroWhenTeamsHaveEqualPoints`

EXERCISE 9

ADVANCED RULEBOOK

Write the test.

Write the implementation.

Push the both rulebooks into standings tests.

EXERCISE 9 cont.

ADVANCED RULEBOOK

Write the test.

Write the implementation.

Push the both rulebooks into standings tests.

Tests

`tests/Domain/Standings/StandingsWithSimpleRuleBookTest.php`

`tests/Domain/Standings/StandingsWithAdvancedRuleBookTest.php`

Implement scenarios

WORKSHOP RECAP



QUESTIONS? /r/AMA?



Luka Muzinic

@lmuzinic

luka.muzinic.net

luka.muzinic.net/pdf/longhorn-pragmatic-tdd.pdf

READING & LIBRARIES

Reading list

<https://www.devmynd.com/blog/five-factor-testing/>

<https://martinfowler.com/articles/practical-test-pyramid.html>

<https://dev.to/theobendixson/the-problem-that-unit-tests-solve-b2l>

<https://blog.liplex.de/testing-private-and-protected-methods-with-phpunit/>

Libraries

<https://github.com/sebastianbergmann/phpunit>

<https://github.com/phpspec/phpspec>

<https://github.com/Codeception/Codeception>

<https://github.com/phpstan/phpstan>

<https://github.com/infection/infection>

HOMEWORK

COMPLETE EXERCISES

- do Exercise 1**
- find more on Codility, CodeWars, Advent of code, Exercism**
- follow along/google/ask**

STATISTICS

- extend results table, add GB, Home, Away, L5**

THANK YOU



Luka Muzinic

@lmuzinic

luka.muzinic.net

luka.muzinic.net/pdf/longhorn-pragmatic-tdd.pdf

Photos by Les Anderson, Joshua Earle, Ian Espinosa and Tom Roberts on Unsplash