1) Abra a classe AvaliadorTest e adicione os três métodos abaixo, que representam os provedores de dados para os testes:

public function leilaoEmOrdemCrescente()

{

$leilao = new Leilao('Fiat 147 0KM');

$maria = new Usuario('Maria');

$joao = new Usuario('João');

$ana = new Usuario('Ana');

$leilao->recebeLance(new Lance($ana, 1700));

$leilao->recebeLance(new Lance($joao, 2000));

$leilao->recebeLance(new Lance($maria, 2500));

return [

'ordem-crescente' => [$leilao]

];

}

public function leilaoEmOrdemDecrescente()

{

$leilao = new Leilao('Fiat 147 0KM');

$maria = new Usuario('Maria');

$joao = new Usuario('João');

$ana = new Usuario('Ana');

$leilao->recebeLance(new Lance($maria, 2500));

$leilao->recebeLance(new Lance($joao, 2000));

$leilao->recebeLance(new Lance($ana, 1700));

return [

'ordem-decrescente' => [$leilao]

];

}

public function leilaoEmOrdemAleatoria()

{

$leilao = new Leilao('Fiat 147 0KM');

$maria = new Usuario('Maria');

$joao = new Usuario('João');

$ana = new Usuario('Ana');

$leilao->recebeLance(new Lance($joao, 2000));

$leilao->recebeLance(new Lance($maria, 2500));

$leilao->recebeLance(new Lance($ana, 1700));

return [

'ordem-aleatoria' => [$leilao]

];

}

2) Ainda no AvaliadorTest, adicione no início um novo atributo $leiloeiro e método setup, para inicializar o Avaliador:

private $leiloeiro;

protected function setUp(): void

{

$this->leiloeiro = new Avaliador();

}

3) Refatore os testes na classe AvaliadorTest, para usar os *dataProviders* e o atributo $leiloeiro. Na classe teremos apenas três testes:

* testAvaliadorDeveEncontrarOMaiorValorDeLances
* testAvaliadorDeveEncontrarOMenorValorDeLances
* testAvaliadorDeveBuscar3MaioresValores

Cada método deve usar os @dataProvider, receber um parâmetro Leilao e usar o atributo $leiloeiro. Por exemplo, o teste testAvaliadorDeveEncontrarOMaiorValorDeLances ficará da seguinte forma:

/\*\*

\* @dataProvider leilaoEmOrdemAleatoria

\* @dataProvider leilaoEmOrdemCrescente

\* @dataProvider leilaoEmOrdemDecrescente

\*/

public function testAvaliadorDeveEncontrarOMaiorValorDeLances(Leilao $leilao)

{

// Act - When

$this->leiloeiro->avalia($leilao);

$maiorValor = $this->leiloeiro->getMaiorValor();

// Assert - Then

self::assertEquals(2500, $maiorValor);

}

4) Novamente, execute os testes na linha de comando:

vendor\bin\phpunit --colors tests

Veja a saída do **PHPUnit**. Repare que temos mais testes e mais *asserts*.

5) Na raiz do seu projeto, crie um novo arquivo chamado **phpunit.xml**, com o seguinte conteúdo:

<phpunit

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation="https://schema.phpunit.de/8.1/phpunit.xsd"

colors="true">

<testsuites>

<testsuite name="unit">

<directory>tests</directory>

</testsuite>

</testsuites>

<!--

<logging>

<log type="testdox-text" target="testes-executados.txt"/>

</logging>

-->

</phpunit>

6) Execute novamente os testes na linha de comando, mas usando apenas o comando phpunit:

vendor\bin\phpunit

7) No arquivo **phpunit.xml**, *descomente* a seção do *logging*.

Agora chame novamente o phpunit na linha de comando e verifique se o arquivo **testes-executados.txt** foi criado.