# **Anagramme**

**Fait travailler** :

* boucles
* tableaux
* parcours de string

**Enoncé du Dojo :**

Dans une chaîne de caractère donnée en paramètre, sortir un tableau de tableaux contenant la liste des différents anagrammes contenus dans la phrase (chaque ligne du premier tableau contient un tableau avec les anagrammes).

**Exemples :**

Entrée : “ab ba ac ca”

Sortie : [

[‘ab’, ‘ba],

[‘ac’, ‘ca’],

]

Entrée: ‘le chien est dans la niche’

Sortie: [[chien,niche’]]

**Correction :**

**class** Annagramme

{

**public function** find($sentence){

$words = *explode*(**' '**,$sentence);

**foreach** ($words **as** $word){

$letters = *str\_split*($word);

*sort*($letters);

$fingerprint = *implode*($letters);

**if** (!**isset**($tab[$fingerprint])){

$tab[$fingerprint]=[];

}

**if** (!*in\_array*($word,$tab[$fingerprint] )){

$tab[$fingerprint][]= $word;

}

}

$res=[];

**foreach** ($tab **as** $value){

**if**(*count*($value) > 1){

$res[]=$value;

}

}

**return** $res;

}

}

**Tests :**

*la correction du dojo doit toujours être donnée avec les tests correspondants*

**class** AnagrammeTest **extends** TestCase

{

**public function** testAnagramme()

{

$anagramme = **new** Anagramme();

$this->assertEquals([[**'chien'**,**'niche'**]],$anagramme->find(**'chien niche'**));

$this->assertEquals([[**'ab'**,**'ba'**]],$anagramme->find(**'ab ba'**));

$this->assertEquals([[**'ab'**,**'ba'**],[**'ac'**,**'ca'**]],$anagramme->find(**'ab ba ac ca'**));

$this->assertEquals([[**'chien'**,**'niche'**]],$anagramme->find(**'le chien est dans la niche'**));

$this->assertEquals(

[

[**'chien'**,**'niche'**, **'chine'**],

[**'marche'**,**'charme'**],

[**'une'**, **'nue'**],

[**'limace'**, **'malice'**],

],

$anagramme->find(**'Le chien marche vers sa niche et trouve une limace**

**de chine nue et pleine de malice qui lui fait du charme'**));

}

}