Exercice 2

- Ligne 1

Ouverture de la balise `php` .

- Ligne 2

On essaye d'exécuter la suite du code, en cas d'erreur on se dirige vers la ligne 9.

- Ligne 4

On définit les variables `$user` et `$pass` .

- Ligne 5

On se connecte à la base de données.

- Ligne 7

On ferme la connexion.

- Ligne 9

On exécute ce code en cas d'erreur.

Exercice 3

Les 2 méthodes de la classe PDO qui construisent un objet de la `classe` PDOStatement sont `query()` et `prepare()` La méthode `query()` renvoie un objet PDOStatement avec un jeu de données. Cet objet peut donc exécuter ses méthodes `fetch()` , `fetchAll()` , `fetchObject()` et `fetchColumn()` qui permettent de récupérer des données dans son `jeu d'enregistrement` . La méthode `prepare()` renvoie un objet PDOStatement qui contient une requête « compilée » mais pas de jeu de données car la requête SQL n'a pas été `éxécutée` Il ne peut donc pas utiliser les méthodes dont le nom commence par `fetch` et qui nécessitent un jeu de données. Dans tous les cas, cet objet PDOStatement doit d'abord exécuter sa méthode `execute()` en lui passant un `ensemble de` paramètres. S'il s'agit d'une requête SELECT, cela lui permettra de récupérer un `jeu de données` .

Exercice 4

1

```php
<?php
class Vol { //définition de la classe Vol
 public $numvol; //numvol est une variable. Corrigez les erreurs
 public $numpil;
 public $numav;
 public $villedep;
 public $villearr;
 public $heuredep;
 public $heurearr;
} // autres erreurs à corriger ?
$host='localhost'; $db='bd_vol_pil1'; $user='root'; $pass='';
```

```php
try{
 $pdo = new PDO("mysql:host={$host};dbname={$db}", $user, $pass);
 $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
 $sql = 'select * from vol';
 $st = $pdo->query($sql);
 $resultArray = $st->fetchALL(PDO::FETCH_CLASS,'Vol');
 //var_dump($resultArray);
 foreach($resultArray as $vol){
 echo $vol->villedep,' ',$vol->villearr,' ',$vol->heuredep,'<br/>';
 }
 $sql = 'select * from vol WHERE villedep = :ville';
 $st = $pdo->prepare($sql);
 $result_bool = $st->execute(array(':ville' =>'Nice'));
 while ($row = $st->fetch(PDO::FETCH_NUM, PDO::FETCH_ORI_NEXT)) {
 $data = 'Destination '.$row[4].'à'.$row[5].'<br/>';
 echo $data;
 }
 $st = null;
 $pdo = null;
}
catch(PDOException $e){
 echo $e->getMessage();
}
?>
```

2

- Ligne 13

  > *Nous créons un nouvel objet appartenant à la classe `PDO()` que nous affectons à la variable `$pdo` Cet objet crée une `connexion` vers la base de donnée bd_vol_pil_av.*

- Ligne 14 (0,4)

  > *Nous configurons l'objet $pdo en utilisant sa méthode `setAttribute()` Nous voulons qu'en cas d'erreur, PDO émette `une exception` .*

- Ligne 16 (1,4)

  > *C'est l'objet `$pdo` appartenant à la classe `PDO` qui appelle sa méthode `query()` avec comme paramètre la variable `$sql` contenant la chaîne `"select * from vol WHERE villedep = :ville"` Le résultat de cette exécution est un objet de la classe `PDOStatement` qui sera stocké dans la variable `$st`*

- Ligne 17 (1,6)

  > *C'est l'objet `$st` appartenant à la classe `PDOStatement` qui appelle sa méthode fetchAll avec comme arguments la constante `PDO::FETCH_CLASS` et la chaîne `Vol` . Ces arguments indiquent que chaque ligne du résultat de la requête sera recopiée dans un `objet` appartenant à la classe `PDO::FETCH_CLASS` . Tous les objets du jeu d'enregistrement correspondant aux lignes récupérées par le SELECT sont insérés dans un `tableau` qui sera conservé dans la variable `$resultArray` .*

- Ligne 20 (0,8)

> *Cette boucle foreach `parcours` le tableau $resultArray case par case. A chaque fois le contenu de la case courante qui est un `objet` appartenant à la classe `PDO::FETCH_CLASS` est copié dans la variable `$vol` ;*

- Ligne 21 (0,8)

  > *La fonction echo copie les attributs `villedep` , `villearr` et `heuredep` de l'objet $vol courant dans `le contenu éxécutable` ce qui entraînera leur affichage lorsque le code HTML sera traité le navigateur.*

- Ligne 24 (0,8)

  > *:ville est un `placeholder` Il faudra lui attribuer une `valeur` pour que la requête devienne `éxécutable` car elle ne peut pas être `éxécutée` sous cette forme.*

- Ligne 27 (2,4)

  > *Dans la condition du while il y a une affectation. Si la valeur affectée à la variable est différente de false ou 0 ou un élément vide, on considère que la condition est vérifiée et la boucle est `arrêtée` L'objet $st appartenant à la classe `PDOStatement` exécute sa méthode `fetch()` avec comme arguments `PDO::FETCH_NUM` et `PDO::FETCH_ORI_NEXT` Le premier argument PDO::FETCH_NUM indique que `le résultat sera un tableau indexé par le numéro de la colonne` . Le deuxième argument PDO::FETCH_ORI_NEXT indique que `le résultat correspondra au numéro de la ligne` . Le résultat de cette exécution est un `tableau` qui sera mis dans la variable `$row` . Mais quand il n'y aura plus de ligne à récupérer le résultat sera `null ou 0` Si $row contient une ligne alors while considère que sa condition est vérifiée et il exécute la boucle. Lorsqu'il n'y a plus de lignes et que $row contient donc `null ou 0` , while `s'arrête` .*

- Ligne 34 (0,2)

  > *Le bloc catch est exécuté uniquement si `try échoue` . Dans ce cas, l'objet $e est rempli avec les informations renvoyée par PHP et décrivant l'erreur.*