PHP 練習問題. 12 データベース処理のクラス化

※以下の練習問題は、「練習問題 11-5」の解答をコピーしてお使いください。

練習問題 11-5 まで進んでいただいてお気づきのことはありませんでしょうか？
データベースにアクセスする PHP ファイルには、必ず下記の記載があります。

```
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', 'root');
$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

規模が小さいシステムならあまり問題が発生しないかもしれません。しかし、大量の PHP ファイルでデータベースに接続する処理があり、データベースのホスト名やユーザー名、パスワードを変更する必要がでたとき、すべての PHP ファイルを修正しなければならないのは大変な作業です。

そういった問題を解決するために、共通の処理を一つのファイルにまとめて記述し、必要な箇所でそのファイルを読み込んで再利用する、ということが可能になっています。

また、PHP は「オブジェクト指向」に対応していますので、データベースに接続する処理のみをスーパークラス（親クラス、基底クラス）として作成し、各テーブルに対する処理をスーパークラスを継承したサブクラス（子クラス）として作成することが可能です。

練習問題 12 では、練習問題 11-5 で作成したアプリケーションのデータベース処理の部分を、別のファイルにクラスとして分けて記述してみます。

(1) 命名規則について

プログラムのソースコードは、各個人が好き勝手に書いていると、非常に読みにくい事になってしまいます。
そこで、変数や定数、クラス名には「命名規則」といって、名前の付け方のルールを決めます。
PHP では、下記のようなルールを使うことが多いです。

1. 変数の命名規則

変数名は、「アルファベット小文字と大文字の組み合わせ」と「アルファベット小文字と_（アンダースコア）」を使う方法があります。

① 単語はアルファベット小文字にします。複数の単語を結合する場合は、結合する単語の先頭を大文字にします。（アルファベット小文字と大文字の組み合わせ）
このような命名規則を「キャメルケース」といいます。
例）
$str
$todoItem
$expirationDate
$isCompleted

② 単語はアルファベット小文字にします。複数の単語を結合する場合は、単語と単語の間に_（アンダースコア）を挿入します。
このような命名規則を「スネークケース」といいます。
例）
$str
$todo_item
$expiration_date
$is_completed

③ いずれの方法を使っても構いませんが、どちらかに統一されている方が可読性が高まります（ソースコードが読みやすくなります）。
PHP では、どちらかというと、スネークケースを使うことが多いです。（Java ではパスカルケースが多いですね。）

2. 定数の命名規則

定数名は、基本的にアルファベット大文字にします。単語で結合する場合は、「_」（アンダーバー）で区切ります。
例）
DB_HOST
DB_USER
USER_NAME
USER_PASS

3. クラス名の命名規則
   クラス名は、先頭を大文字にして、あとは小文字にします。単語で結合する場合は、結合する単語の先頭を大文字にします。

   例）
   Base
   TodoItems
   UserWorks

   また、クラスひとつにつきファイルひとつにすることが多いです。その場合は、クラス名と同じファイル名にします。

   例）
   Base　　→ Base.php
   TodoItems　　→ TodoItems.php
   UserWorks　　→ UserWorks

(2) クラスのファイルを置くディレクトリを作成する

クライアントからアクセスできる PHP ファイルとは別に、クラスのファイルを置くディレクトリを作成しましょう。
別のディレクトリにする理由としては、

1. 管理をしやすくする
2. ディレクトリ全体をリモートからアクセスできないようにすることによって、セキュリティを保つ

です。

index.php があるディレクトリに、下記のディレクトリを作成してください。

class/db/

こちらのディレクトリの中に、クラスのファイルを作成します。

(3) データベースにアクセスするスーパークラスを作成する

class/db/Base.php を新規作成してください。

1. Base.php に Base クラスを作成します。クラスには下記のメンバを作成してください。

   - 接続データベース名（定数）
   - データベースホスト名（定数）
   - データベース接続ユーザー名（定数）
   - データベース接続パスワード（定数）
   - PDO クラスインスタンスを代入する変数（protected）

2. コンストラクタを作成します
   コンストラクタでは、下記の処理を行います。

   - データベース接続文字列の作成
   - PDO クラスのインスタンスを作成し、「PDO クラスインスタンスを代入する変数」に代入
   - PDO クラスのインスタンスのエラーモードを、例外をスローするように設定

「PDO クラスインスタンスを代入する変数」は「protected」になっているので、このスーパークラスを継承するサブクラスで使用することができます。サブクラスのインスタンスを生成することで、親クラスで生成した PDO クラスのインスタンスにアクセスできるようになるわけです。

(4) Base クラスを継承する TodoItems クラスを作成する

class/db/TodoItems.php を新規作成してください。

1. TodoItems.php に Base クラスを継承した TodoItems クラスを作成します。

2. コンストラクタを作成します
   TodoItems クラスのコンストラクタを作成します。コンストラクタでは、スーパークラスである Base クラスのコンストラクタを呼び出します。
3. クラスメソッドを作成します。
   todo_items テーブルに対して処理を行うクラスメソッドを作成します。PHP ファイルで行っている 1 つの処理につき、1 つのメソッドを作ると

いいでしょう。

例）

```
/**
 * レコードを全件取得する（期限日の古いものから並び替える）
 *
 * @return array
 */
public function selectAll()
{
    // SQL 文を作成する
    // SQL 文を実行する準備
    // SQL を実行する
    //  取得したレコードを連想配列として返却する
}
```

※実際の処理はご自身で書いてくださいね。

(5) PHP ファイルでクラスファイルを読み込む

作成したクラスファイルを、データベース処理を行っている PHP ファイルで読み込みます。

ファイルの先頭で書きのように記載します。

例)

```
<?php
// 必要なファイルを読み込む
require_once('./class/db/Base.php');
require_once('./class/db/TodoItems.php');
```

(6) サブクラスのインスタンスを生成し、データベースの処理を行う

例)

```
$db = new TodoItems();
$list = $db->selectAll();
```

設問1. 以上の説明を参考に、修正を行ってください。