1. Lớp (Class) và Đối tượng (Object)

Ví dụ 1: Quản lý Sách

Hướng dẫn:

1. Chúng ta sẽ định nghĩa một lớp tên là Book. Lớp này sẽ có hai thuộc tính (đặc điểm) là $title (tiêu đề) và $author (tác giả).
2. Định nghĩa hành động (Phương thức): Lớp Book sẽ có một phương thức (hành động) tên là getDescription() để in ra thông tin của cuốn sách.
3. Tạo đối tượng thực tế: Từ bản thiết kế Book, chúng ta sẽ tạo ra hai đối tượng (cuốn sách cụ thể) là $book1 và $book2.
4. Gán thông tin và hành động: Chúng ta sẽ gán tiêu đề và tác giả cho từng cuốn sách, sau đó gọi phương thức getDescription() để xem thông tin của chúng.

- Code cụ thể:

```
<?php
// Bước 1: Định nghĩa lớp Book
class Book {
    // Thuộc tính
    public $title;
    public $author;
    // Phương thức
    public function getDescription() {
        // $this->title có nghĩa là "tiêu đề của chính đối tượng này"
        return "Cuốn sách '{$this->title}' được viết bởi tác giả {$this->author}.";
    }
}
```

```
// Bước 2: Tạo các đối tượng từ lớp Book
$book1 = new Book();
$book2 = new Book();
// Bước 3: Gán giá trị thuộc tính cho đối tượng thứ nhất
$book1->title = "Đắc Nhân Tâm";
$book1->author = "Dale Carnegie";
// Bước 4: Gán giá trị thuộc tính cho đối tượng thứ hai
$book2->title = "Nhà Giả Kim";
$book2->author = "Paulo Coelho";
// Bước 5: Gọi phương thức từ mỗi đối tượng
echo $book1->getDescription(); // In ra thông tin của book1
echo "<br>";
echo $book2->getDescription(); // In ra thông tin của book2
?>
```

Ví dụ 2-5: Chỉ có hướng dẫn

2. Lớp Student (Sinh viên)
    - Hướng dẫn: Tạo một lớp Student với các thuộc tính name (tên), studentId (mã sinh viên). Thêm một phương thức showProfile() để in ra "Xin chào, tôi là [tên], mã sinh viên của tôi là [mã sinh viên]". Sau đó, tạo một đối tượng sinh viên, gán thông tin và gọi phương thức showProfile().
3. Lớp Dog (Chú chó)
    - Hướng dẫn: Định nghĩa một lớp Dog có thuộc tính name (tên) và breed (giống loài). Viết một phương thức bark() để in ra "Gâu gâu! Tôi là [tên]". Tạo hai đối tượng chó khác nhau, đặt tên cho chúng và gọi phương thức bark() của mỗi con.

4. Lớp Rectangle (Hình chữ nhật)
    - Hướng dẫn: Tạo lớp Rectangle với hai thuộc tính width (chiều rộng) và height (chiều cao). Viết một phương thức calculateArea() để tính và trả về diện tích (width * height). Tạo một đối tượng hình chữ nhật, gán chiều rộng và chiều cao, sau đó gọi phương thức tính diện tích và in kết quả ra màn hình.
5. Lớp Song (Bài hát)
    - Hướng dẫn: Định nghĩa lớp Song có thuộc tính title (tên bài hát) và artist (ca sĩ). Thêm một phương thức play() để in ra "Đang phát bài hát: '[tên bài hát]' của ca sĩ [ca sĩ]". Tạo một đối tượng bài hát, gán thông tin và gọi phương thức play().

---

2. Hàm tạo (__construct) và Hàm hủy (__destruct)

Ví dụ 1: Kết nối Cơ sở dữ liệu (Có code mẫu)

- Hướng dẫn:
    1. Hàm tạo: Chúng ta sẽ tạo một lớp DatabaseConnection. Khi một đối tượng của lớp này được tạo, hàm __construct() sẽ tự động chạy và mô phỏng việc "mở kết nối" đến CSDL bằng cách in ra một thông báo.
    2. Hàm hủy: Khi đối tượng không còn được sử dụng nữa (ví dụ khi script kết thúc), hàm __destruct() sẽ tự động được gọi để "đóng kết nối", đảm bảo tài nguyên được giải phóng.
- Code cụ thể:

PHP

```
<?php
class DatabaseConnection {
    // Hàm tạo, được gọi khi "new DatabaseConnection()"
    public function __construct() {
        echo "Đã mở kết nối đến cơ sở dữ liệu.<br>";
```

```
    }
    public function query($sql) {
        echo "Thực thi câu lệnh: $sql <br>";
    }
    // Hàm hủy, được gọi tự động khi script kết thúc
    public function __destruct() {
        echo "Đã đóng kết nối cơ sở dữ liệu. Giải phóng tài nguyên.";
    }
}
// Khi dòng này được thực thi, hàm __construct() sẽ tự động chạy
$connection = new DatabaseConnection();
// Sử dụng đối tượng
$connection->query("SELECT * FROM users");
// Khi script chạy xong, hàm __destruct() sẽ tự động được gọi
?>
```

Ví dụ 2-5: Chỉ có hướng dẫn

2. Lớp User (Người dùng)
    - Hướng dẫn: Tạo một lớp User có một thuộc tính username. Viết một hàm __construct($name) để khi tạo đối tượng mới, bạn phải truyền vào tên người dùng và nó sẽ tự gán vào thuộc tính $username. Hàm __destruct() có thể in ra thông báo "[username] đã đăng xuất."
3. Lớp Car (Xe hơi)
    - Hướng dẫn: Tạo lớp Car. Hàm __construct($brand) sẽ nhận vào tên hãng xe và in ra "Một chiếc xe hãng [brand] vừa được sản xuất." Hàm __destruct() sẽ in ra "Chiếc xe [brand] đã được đưa vào bãi phế liệu."
4. Lớp TemporaryFile (Tệp tạm)

- Hướng dẫn: Tạo lớp TemporaryFile. Hàm __construct($filename) sẽ nhận tên tệp và in ra "Đã tạo tệp tạm: [filename]". Hàm __destruct() sẽ in ra "Đã xóa tệp tạm: [filename]", mô phỏng việc dọn dẹp các tệp không cần thiết.

5. Lớp Character (Nhân vật game)
    - Hướng dẫn: Tạo lớp Character. Hàm __construct($name) nhận tên nhân vật và in ra "[name] đã tham gia vào trò chơi." Hàm __destruct() sẽ in ra "[name] đã rời khỏi trò chơi."

---

3. Phạm vi truy cập (public, protected, private)

Ví dụ 1: Quản lý Nhân viên (Có code mẫu)

- Hướng dẫn:
    1. Định nghĩa phạm vi: Tạo lớp Employee với public $name, protected $salary (lương), và private $pinCode (mã PIN).
    2. Tạo phương thức công khai: Viết các phương thức public để có thể gán và xem lương một cách an toàn.
    3. Tạo lớp con: Tạo một lớp con Manager kế thừa từ Employee để xem lớp con có thể truy cập được những thuộc tính nào.
- Code cụ thể:

PHP

```
<?php
class Employee {
    public $name; // Ai cũng thấy được
    protected $salary; // Chỉ lớp này và lớp con thấy được
    private $pinCode; // Chỉ lớp này thấy được
    public function __construct($name, $salary, $pinCode) {
        $this->name = $name;
```

```
        $this->salary = $salary;
        $this->pinCode = $pinCode;
    }
    // Phương thức public để gán lương
    public function setSalary($amount) {
        if ($amount > 0) {
            $this->salary = $amount;
        }
    }
    // Phương thức public để xem lương
    public function getSalary() {
        return $this->salary;
    }
}
class Manager extends Employee {
    public function showInfo() {
        echo "Tên: " . $this->name; // OK, vì name là public
        echo "<br>Lương: " . $this->salary; // OK, vì salary là protected và Manager là lớp con
        // echo "<br>Mã PIN: " . $this->pinCode; // LỖI! Không thể truy cập private của lớp cha
    }
}
$manager = new Manager("Nguyễn Văn A", 5000, "1234");
echo "Tên quản lý: " . $manager->name; // OK, vì name là public
```

```
// echo "Lương: " . $manager->salary; // LỖI! Không thể truy cập protected từ bên ngoài

// echo "Mã PIN: " . $manager->pinCode; // LỖI! Không thể truy cập private từ bên ngoài

echo "<br>---<br>";

$manager->showInfo(); // Hiển thị thông tin từ bên trong lớp Manager

?>
```

Ví dụ 2-5: Chỉ có hướng dẫn

2. Lớp BankAccount (Tài khoản ngân hàng)
    - Hướng dẫn: Tạo lớp BankAccount với thuộc tính private $balance (số dư) để không ai có thể thay đổi số dư một cách trực tiếp. Viết hai phương thức public là deposit($amount) (nạp tiền) và withdraw($amount) (rút tiền) để thay đổi số dư một cách an toàn. Thêm một phương thức public getBalance() để xem số dư.
3. Lớp Animal và Cat
    - Hướng dẫn: Tạo lớp cha Animal với một thuộc tính protected $name. Tạo một lớp con Cat kế thừa từ Animal. Trong lớp Cat, viết một phương thức meow() để in ra "Meo! Tôi là [name]", chứng tỏ lớp con có thể truy cập thuộc tính protected của lớp cha.
4. Lớp Article (Bài báo)
    - Hướng dẫn: Tạo lớp Article với public $title (tiêu đề) và private $content (nội dung). Viết một phương thức public function getContent() để trả về nội dung. Điều này cho phép người khác đọc được nội dung nhưng không thể sửa đổi trực tiếp thuộc tính $content.
5. Lớp Person và Adult
    - Hướng dẫn: Tạo lớp cha Person có public $name và private $age. Tạo một phương thức public trong Person tên là isAdult() để kiểm tra xem $age có lớn hơn 18 hay không. Tạo một lớp con Adult kế thừa Person, thử truy cập $age từ lớp Adult để thấy rằng nó sẽ báo lỗi.

Bài tập về tính kế thừa

1. Lớp Vehicle và Car
2. Hướng dẫn:
    1. Tạo lớp cha (Lớp cơ sở): Chúng ta sẽ định nghĩa một lớp Vehicle (Phương tiện) chung chung. Lớp này có thuộc tính brand (hãng sản xuất) và một phương thức startEngine() để mô phỏng việc khởi động động cơ.
    2. Tạo lớp con (Lớp dẫn xuất): Chúng ta sẽ tạo một lớp Car (Xe hơi) kế thừa từ lớp Vehicle bằng từ khóa extends. Lớp Car sẽ tự động có được tất cả thuộc tính và phương thức public và protected của Vehicle.
    3. Mở rộng chức năng: Lớp Car sẽ có thêm một phương thức riêng của nó là honk() (bấm còi), điều mà không phải phương tiện nào cũng có.
    4. Sử dụng đối tượng: Chúng ta sẽ tạo một đối tượng từ lớp con (Car) và gọi cả phương thức được kế thừa từ lớp cha (startEngine()) và phương thức của riêng nó (honk()).

- Code

```
<?php
// Bước 1: Định nghĩa lớp cha (lớp cơ sở)
class Vehicle {
    // Thuộc tính này sẽ được kế thừa
    public $brand;
    // Phương thức này cũng sẽ được kế thừa
    public function startEngine() {
        echo "Động cơ đã khởi động!<br>";
    }
}
// Bước 2: Định nghĩa lớp con kế thừa từ Vehicle
```

```
class Car extends Vehicle {
    // Phương thức này chỉ có ở lớp Car
    public function honk() {
        echo "Bíp! Bíp!<br>";
    }
}
// Bước 3: Tạo một đối tượng từ lớp con
$myCar = new Car();
// Bước 4: Gán giá trị cho thuộc tính được kế thừa từ lớp cha
$myCar->brand = "Toyota";
echo "Chiếc xe của tôi là hãng: " . $myCar->brand . "<br>";
// Bước 5: Gọi các phương thức
$myCar->startEngine(); // Gọi phương thức kế thừa từ lớp Vehicle
$myCar->honk();        // Gọi phương thức của riêng lớp Car
?>
```

---

2. Lớp Shape (Hình dạng) và Circle (Hình tròn)

- Hướng dẫn: Tạo một lớp cha Shape có thuộc tính color (màu sắc) và một phương thức displayColor() để in ra "Hình này có màu [màu sắc]". Sau đó, tạo một lớp con Circle kế thừa từ Shape. Lớp Circle có thêm một thuộc tính riêng là radius (bán kính) và một phương thức calculateArea() để tính và trả về diện tích hình tròn (). Cuối cùng, tạo một đối tượng hình tròn, gán màu và bán kính, sau đó gọi cả hai phương thức displayColor() và calculateArea().

---

3. Lớp MediaItem (Mục đa phương tiện) và Video

- Hướng dẫn: Định nghĩa lớp cha MediaItem với thuộc tính title (tiêu đề) và một phương thức play() để in ra "Đang phát: [tiêu đề]". Tạo một lớp con Video kế thừa từ MediaItem. Lớp Video có thêm thuộc tính riêng là resolution (độ phân giải). Viết lại (ghi đè) phương thức play() trong lớp Video để nó in ra "Đang phát video '[tiêu đề]' với độ phân giải [độ phân giải]". Tạo một đối tượng Video, gán thông tin và gọi phương thức play() của nó.

---

4. Lớp User (Người dùng) và Admin

- Hướng dẫn: Tạo lớp cha User với các thuộc tính username và email. Viết một phương thức viewProfile() để hiển thị thông tin người dùng. Sau đó, tạo lớp con Admin kế thừa từ User. Lớp Admin có thêm một phương thức đặc biệt là deletePost($postId) để in ra "Admin [username] đã xóa bài viết có ID là [postId]". Tạo một đối tượng Admin, gán thông tin, rồi gọi cả phương thức viewProfile() và deletePost().

---

5. Lớp Appliance (Thiết bị gia dụng) và Television

- Hướng dẫn: Định nghĩa lớp cha Appliance có thuộc tính brand (thương hiệu) và hai phương thức turnOn() và turnOff(), mỗi phương thức in ra thông báo tương ứng. Tạo một lớp con Television kế thừa từ Appliance. Lớp Television có thêm một phương thức riêng là changeChannel($channelNumber) để in ra "Đã chuyển đến kênh số [channelNumber]". Tạo một đối tượng Television, gán thương hiệu, sau đó gọi các phương thức turnOn(), changeChannel(), và turnOff().

    -