**3. Ví dụ**

**3.1 Double click vào file lab2\_1a.html. Cho biết kết quả.**

lab2\_1a.html sẽ chạy trên trình duyệt và hiện ra chữ **HELLO**

**3.2 Double click vào file lab2\_1b.php. Cho biết kết quả.**

lab2\_1b.php sẽ được mở trên Dreamwearver và ta sẽ thấy code viết trong file bao gồm code php và html chứ nó không chạy thẳng lên trình duyệt

**3.3 Chép thư mục lab 2 vào thư mục gốc web server và chạy file lab2\_1b.php**

**từ localhost. Cho biết kết quả.**

Ta sẽ thấy chữ **HELLO** xuất hiện trên trình duyệt

**3.4 Mở file lab2\_1a.html và lab2\_1b.php để xem code. Sau đó xem source code file lab2\_1b.php trên trình duyệt.**

Mở file lab2\_1a.html và lab2\_1b.php trên notepad++ để xem code thì

file lab2\_1a.html chỉ có code html ; lab2\_1b.php thì thấy cả code php và html

source code file lab2\_1b.php trên trình duyệt sẽ chỉ thấy code html chứ không thấy code php nữa

**3.6 Ví dụ lab2\_3.php: cách sử dụng hằng số**

(ĐÃ XEM)

Dùng define để định nghĩa hằng số

**3.7 Ví dụ lab2\_4.php: toán tử so sánh.**

**\***Dấu == dùng để so sánh về giá trị

\*Dấu === dùng để so sánh về cả giá trị và kiểu dữ liệu

**3.8 lab2\_5 ví dụ về sử dụng hàm isset để kiểm tra biến có tồn tại không và cách gọi 1 file php khác.**

(Đã xem)

**4. Vận dụng**

**4.1 Trong lab2\_2.php, xóa bỏ global thì kết quả xuất ra mà hình là gì? Vì sao?**

Kết quả như sau:

a = 1  
b = 2  
kq = 0

Vì khi bỏ global thì kq sẽ không còn là biến toàn cục nữa lúc này nó sẽ trở thành biến cục bộ. Gía trị của biến kq trong hàm f() không có global thì giá trị kq trong hàm f() sau khi kết thúc hàm f() thì giá trị đó sẽ bị hủy nên cuối kq mới bằng 0 như vậy.

**4.2 Có thể thay thế dòng 10: const PI = 3.14; bằng lệnh nào khác không?**

Define(“PI”,3.14);

**4.3 Trong lab2\_4.php, biến a kiểu số nguyên, biến x kiểu chuỗi nhưng a+x hay x+a đều được do PHP tự ép kiểu. Vậy tại sao phép so sánh dòng 21 lại xác định a và x khác nhau? Muốn xuất ra “a va x giống nhau” thì phải sửa gì dòng 21.**

\*Phép so sánh ở dòng 21 xác đinh a và x khác nhau vì === là so sánh về cả dữ liệu và giá trị mà ở đây kiểu dữ liệu của a là số nguyên biến x là kiểu chuỗi.

\*Muốn a và x giống nhau thì ở dòng 21 ta sửa dấu === thành dấu ==.

**4.4 Comment dòng 10 và chạy lab2\_5.php. Giải thích kết quả**

Biến x không tồn tại.

Vì nếu comment dòng 10 lại thì lab2\_5a.php sẽ không được gọi và do đó biến $x trong file đó cũng không được truyền vào nên có kết quả là biến x không tốn tại.

**4.5 Thêm include(“lab2\_5b.php”); vào lab2\_5.php (không comment dòng 10)**

**tại dòng 11 và lưu lại thành tên vd4\_5.php. Chạy vd4\_5.php và cho biết**

**kết quả.**

Giá trị của x là: 20

Vì sau khi include("lab2\_5a.php"); ở dòng 10 thì $x=10; gọi vào nên x sẽ có giá trị là 10. Sau khi tiếp tục include("lab2\_5b.php"); thì $x+=10; sẽ được gọi vào ở dòng 11 thì x sẽ được cộng thêm 10 nên kết quả là x có giá trị là 20 như trên.

**4.6 Thêm include(“lab2\_5b.php”); vào vd4\_5.php tại dòng 12 và lưu thành**

**tên vd4\_6.php. Chạy vd4\_6.php và cho biết kết quả.**

Giá trị của x là: 30

Ví từ ví dụ **vd4\_5.php**  thì x đã có giá trị là 20 thì sau khi tới **include(“lab2\_5b.php”);** ở dòng 12 thì x lại được cộng thêm 10 nên có giá trị là 30.

**4.7 Thay include(“lab2\_5b.php”); dòng 12 trong file vd4\_6.php thành**

**include\_once(“lab2\_5b.php”); và lưu lại với tên vd4\_7.php. Chạy**

**vd4\_7.php và so sánh kết quả với kết quả câu 4.6.**

Giá trị của x là: 20

Vì **include\_once** có nghĩa là nó kiểm trafile **lab2\_5b.php** này đã được gọi chưa nếu gọi rồi thì thôi không gọi nữa.Đó là lí do x chỉ có giá trị là 20.Còn ở ví dụ **vd4\_6** thì dòng 12 là include thì file nào được gọi rồi thì gọi nữa cũng không sao nên giá trị của x là 30.

**4.8 Lần lượt thay include thành require trong vd4\_6.php và vd4\_7.php và lưu lại với tên vd4\_6b.php và vd4\_7b.php. So sánh kết quả với câu 4.6 và4.7.**

\*Ta có kết quả như sau: kết quả giống nhau

**+vd4\_6.php** giống **vd4\_6b.php** xuất ra kết quả Giá trị của x là: 30

**+vd4\_7.php** giống **vd4\_7b.php** xuất ra kết quả Giá trị của x là: 20

**4.9 Xóa file lab2\_5b.php rồi lần lượt chạy vd4\_6.php và vd4-6b.php. So sánh**

**kết quả, từ đó rút ra sự khác nhau giữa include và require.**

**+vd4\_6.php** là include nên cảnh báo nhưng vẫn chạy tiếp chương trình xuất ra x=10 còn **vd4-6b.php** thì nó là require nên nó báo lỗi và dừng chương trình.

hai câu lệnh này về chức năng hoàn toàn tương đồng nhau. Nhưng nó chỉ khác khi thông báo lỗi.

- include sẽ sinh ra cảnh báo và vẫn chạy chương trình

- require sẽ sinh ra lỗi và dừng chạy chương trình