PHP3

Funciones

FUNCIONES

Son una sección separada de código que tiene un propósito específico, a la cual se le asigna un nombre. Se utilizan para dividir el código de un script en partes menores (modularidad).

Sintaxis:

```
function nombrefuncion (parámetros){
sentencias;
return valor;
}
```

PASO DE PARÁMETROS A LAS FUNCIONES

Por valor

La función recibe una copia del valor de la variable, al terminar de ejecutarse la función no se ha alterado el valor de la variable que fue pasada como argumento.

Ejemplo:

```
function iExponente($a){
return $a*$a;
}
```

PASO DE PARÁMETROS A LAS FUNCIONES

Por referencia

Se altera el valor de la variable pasada como argumento a la función, esto es debido a que se pasa en realidad una referencia de la variable y no una copia de su valor como en el caso anterior.

Ejemplo:

```
function iExponente(&$a){
return $a*$a;
}
```

PASO DE PARÁMETROS A LAS FUNCIONES

Por defecto

Son parámetros opcionales en la llamada a las funciones, este tipo de parámetros toma un valor predefinido en caso que no se especifique el argumento en la llamada a la función.

Ejemplo:

```
function iExponente($a,$exponente= 2){
return $a*$a;
}
```

FUNCIONES PARA VARIABLES

gettype(variable) : Devuelve el tipo de dato del parámetro.

settype(variable, tipo) : Establece el tipo de dato a guardar en una variable, realiza conversiones de tipo de datos.

isset(variable) : Devuelve true si una variable ha sido inicializada con un valor, de lo contrario devuelve false.

empty(variable) : Devuelve true si la variable no ha sido inicializada, si tiene un valor 0 o si es una cadena vacia, de lo contrario devuelve

false.

FUNCIONES PARA VARIABLES

is_int(variable) : True si la variable es entera.
is_float(variable) : True si la variable es flotante.
is_numeric(variable) : True si la variable es un número o una cadena numérica.
is_bool(variable) : True si la variable es de tipo lógico.
is_array(variable) : True si la variable es de tipo arreglo.
is_string(variable) : True si la variable es de tipo cadena.
is_object(variable) : True si la variable es de tipo objeto.

funciones.php

```
< ht m l>
< head>
< t itle> FUNCIONES< / tit le>
< / head>
< body>
< h1 align= "cent er"> FUNCIONES< / h1> < hr>
<?
f unction iEx ponent ePorValor($inumero){
echo "Duplicando el valor del
argum ent o< br> ";
echo "Resultado = " . $inum ero*$inumero .
"< br> ";
ret urn $inum ero*= $inum ero;
}
f unction iExponentePorReferencia
(&$inumero2){
echo "Duplicando la referencia pasada como
argument o< br> ";
echo "Resultado = " .
$inum ero2*$inumero2 . "< br> ";
return $inumero2*= $inumero2;
}
f unction iExponentePorDefect o
($inumero,$iEx ponente= 2){
$resultado= 1;
f or($i= 1;$i< = abs($iExponent e);$i+ + )
$resultado*= $inumero;
if ($iExponent e< 0)
$resultado= 1/ $result ado;
ret urn $result ado;
}
$ioperador1= 2;
$iresult ado= iExponentePorValor
($ioperador1);
echo "Valor de \ $iresultado =
$iresult ado< br> ";
echo "Valor de \ $ioperador1 =
$ioperador1< br> ";
$iresultado2= iExponentePorReferencia
($ioperador1);
echo "Valor de \ $iresultado2 =
$iresultado2< br> ";
echo "Valor de \ $ioperador1 =
$ioperador1< br> ";
echo "Sin pasarle ningún valor al argumento:
";
$iresultado3= iExponentePorDefect o
($ioperador1);
echo "Valor de \ $iresultado3 =
$iresultado3< br> ";
echo "Pasandole un valor al argumento: ";
$iresult ado3= iExponentePorDefect o
($ioperador1,3);
echo "Valor de \ $iresultado3 =
$iresultado3< br> ";
?>
< / body>
< / ht m l>
```

PRACTICA DE FUNCIONES

1.-Implementar una función que regrese el mayor de una lista de 4 números.

2.- Implementar una función que valide las variables enviadas por un formulario e imprima el tipo.

FUNCIONES CON NÚMERO VARIABLE DE PARÁMETROS

Se permite definir funciones en las que no esté fijado el número de parámetros, su funcionamiento se basa en las siguientes funciones:

func_num_args() Devuelve el número de argumentos pasados a la función

Func_get_args() Devuelve un array con los argumentos pasados a la función

Func_get_arg() Devuelve un elemento de la lista de argumentos pasados a la función, los argumentos comienzan en la posición cero, si se solicita un argumento de una posición que no existe devuelve false.

funcionnumerovariabledeparametros.php

```
< html>
< head>
< title> FUNCION NÚMERO VARIABLE DE PARÁMETROS< / title>
< / head>
< body>
< h1 align= "center"> FUNCION NÚMERO VARIABLE DE PARÁMETROS< / h1> < hr>
<?
function iElMayor(){
$inumeroargumentos= func_num_args();
$arrargument os= func_get_args();
$imayor= func_get_arg(0);
for($icontador= 1;$icontador< $inumeroargumentos;$icontador+ + ){
if($imayor< func_get_arg($icontador))
$imayor= func_get_arg($icontador);
}
return $imayor;
}
echo iElMayor(4,5,6,34,56,89,765);
?>
< / body>
< / html>
```

PRACTICA DE FUNCIONES CON NÚMERO VARIABLE DE PARÁMETROS

Crear una función que realice la validación de los datos pasados por un formulario verificando su tipo, número de variables e imprimir los resultados en una tabla. Si faltan datos colocar los correspondientes mensajes de error

FUNCIONES RECURSIVAS

Son funciones que en algún punto de su cuerpo se llaman a si mismas, con este tipo de funciones es necesario asegurarse que termine la recursión (condición de parada).

recursivafactorial.php

```
< html>
< head>
< title> FUNCIONES RECURSIVAS< / title>
< / head>
< body>
< h1 align= "center"> FUNCIONES RECURSIVAS< / h1> < hr>
<?
function iFactorial($inumero){
if($inumero= = 0)
return 1;
return $inumero*iFactorial($inumero- 1);
}
echo iFactorial(5);
?>
< / body>
< / html>
```

INCLUSIÓN DE ARCHIVOS

Se utilizan principalmente para la definición de librerías comunes a varios scripts, permitiendo la reutilización del código.

Funciones para la inclusión de archivos:

include(“nombre_del_archivo”)

Esta función incluye y evalua una archivo externo cada vez que es interpretada.

Include_once(“nombre_del_archivo”)

Es igual a include pero sólo se puede incluir el archivo una vez en el script.

inclusiondearchivos.php

```
< html>
< head>
< title> INCLUSIÓN DE ARCHIVOS< / title>
< / head>
< body>
< h1 align= "center"> INCLUSIÓN DE ARCHIVOS< / h1> < hr>
<?
include("cabecerasitio.php");
echo "< h1 align= 'center'> PÁGINA PRINCIPAL< / h1> ";
include("cabecerasitio.php");
include_once("pie.inc");
include_once("pie.inc");
include_once("pie.inc");
?>
< / body>
< / html>
```

PRACTICA INCLUSIÓN DE ARCHIVOS

Generar una archivo que permita incluir los segmentos de encabezado, pie de página y contenido utilizando funciones Include de php para cambiar de manera automática la imagen que se presenta al cargar la página

NOTA: Se pueden utilizar el uso de perfiles usuario->tipo de encabezado o cambios aleatorios para cada visitante de la web.

Se pueden manejar distintos CSS.

MANEJO DE ARCHIVOS

Operaciones que se pueden realizar con archivos:

Abrir.

Cerrar.

Leer.

Escribir.

ABRIR UN ARCHIVO

Sintaxis:

fopen(nombre_archivo,modo_apertura)

VALORES PARA EL PARÁMETRO MODO APERTURA:

r Lectura.El apuntador se coloca al inicio del archivo.

r+ Lectura y escritura.El apuntador se coloca al inicio del archivo.

w Escritura.Si no ex iste el archivo se crea, si ya existe se borra su contenido.

w+ Lectura y escritura.Si no existe el archivo se crea, si ya existe se borra su contenido.

a Escritura.Si no existe el archivo se crea, si ya existe se coloca al final del archivo para añadir datos.

a+ Lectura y escritura.Si no existe el archivo se crea, si ya existe se coloca al final del archivo para añadir datos.

NOTA: fopen() devuelve un apuntador al archivo, a través de este se recorre dicho archivo.

CERRAR UN ARCHIVO

Es recomendable cerrar un archivo cuando ya no se va a usar para no consumir recursos del sistema, si no se cierra PHP lo hará al terminar de ejecutar el script.

Sintaxis:

fclose(apuntador)

LECTURA DE ARCHIVOS

fgetc(apuntador)

Devuelve un caracter del archivo referenciado por apuntador, si se ha llegado al final del archivo devuelve false.

Una forma habitual de utilizar esta función es :

```
while($caracter= fgetc($apuntador)){
sentencias;
}
```

LECTURA DE ARCHIVOS

fgets(apuntador,[total_car_a_leer])

Devuelve una cadena de total de caracteres a leer- 1 o de menor longitud si se ha encontrado un cambio de línea que se incluiría en la cadena a devolver o si se ha llegado al final del archivo.

fread(apuntador,[total_car_a_leer])

Igual a fgets() sólo que no deja de leer cuando encuentra un cambio de línea y devuelve total de caracteres a leer.

LECTURA DE ARCHIVOS

feof(apuntador)

Devuelve true si se ha llegado al final del archivo. Se usa comunmente como:

```
while(!feof($apuntador)){
sentencias;:
}
```

file(nombre_archivo)

Lee todo el contenido de un archivo y lo devuelve en forma de array: una línea en cada posición del arreglo.

readfile(nombre_archivo)

Lee el cont enido de un archivo y lo m uestra por la salida estándar.

leerarchivo.php

```
< html>
< head>
< title> LEER ARCHIVO< / title>
< / head>
< body>
< h1 align= "center"> LEER
ARCHIVO< / h1> < hr>
<?
echo "< hr> < hr> < h1 align= center> fgetc()
< / h1> ";
$apuntador= fopen("prueba.txt","r");
while($caracter= fgetc($apuntador)){
echo $caracter;
}
fclose($apuntador);
echo "< hr> < hr> < h1 align= center> fgets()
< / h1> ";
$apuntador= fopen("prueba.txt","r");
$stexto= fgets($apunt ador);
fclose($apuntador);
echo $stex to;
echo "< hr> < hr> < h1 align= center> fread()
< / h1> ";
$apuntador= fopen("prueba.txt","r");
$stexto= fread($apuntador,5000);
fclose($apuntador);
echo $stexto;
echo "< hr> < hr> < h1
align= center> feof< / h1> ";
$apuntador= fopen("prueba.txt","r");
while(!feof($apuntador)){
$stex.= fgets($apuntador);
}
fclose($apuntador);
echo $stex;
echo "< hr> < hr> < h1 align= center> file()
< / h1> ";
$arrtexto= file("prueba.tx t");
foreach($arrtex to as $value)
echo "$value< br> ";
echo "< hr> < hr> < h1
align= center> readfile()< / h1> ";
readfile("prueba.txt");
?>
< / body>
< / html>
```

ESCRITURA DE ARCHIVOS

fwrite(apuntador,cadena)

fputs(apuntador,cadena)

Ambas funciones escriben la cadena pasada como parámetro, devuelven el total de caracteres escritos o false si se produjo algún error.

DESPLAZARSE EN ARCHIVOS

rewind(apuntador)

Sitúa el apuntador de lectura/ escritura al principio del archivo.

fseek(apuntador,desp[,desde_pos])

Desplaza al apuntador desp posiciones a partir de su posición actual, el tercer parámetro puede tomar los valores SEEK_SET, SEEK_CUR y SEEK_END, que le inidan que se desplaza n desp a partir del principio, posición actual o final del archivo respectivamente (en este caso desp debe ser negativo).

ftell(apuntador)

Devuelve la posición actual del apuntador.

contador.php

```
< html>
< head>
< title> CONTADOR< / title>
< / head>
< body bgcolor= "#3399cc">
< h1 align= "center"> CONTADOR< / h1> < hr>
<?
$archivo= "contador.txt";
$apuntador= fopen($archivo,"r+ ");
$contador= fread($apuntador,6);
echo "< h4 align= 'center'> USTED ES EL VISITANTE NÚMERO :
< br> $contador< / h4> ";
$contador+ + ;
rewind($apuntador);
fwrite($apuntador,$contador);
fclose($apuntador);
?>
< / body>
< / html>
```

agregaralarchivo.php

```
< html>
< head>
< title> ESCRIBIR ARCHIVO< / title>
< / head>
< body>
< h1 align= "center"> ESCRIBIR ARCHIVO< / h1> < hr>
<?
$apuntador= fopen("prueba.txt","a+ ");
fputs($apuntador,"\ n AUTOR : \ n CRI- CRI");
fclose($apuntador);
$apuntador= fopen("prueba.txt","r");
$stexto= fread($apuntador,1000);
fclose($apuntador);
echo "Tex to mofdificado";
echo $stex to;
?>
< / body>
< / html>
```

ALGUNAS FUNCIONES PARA ARCHIVOS

file_exists(archivo)

Regresa True si el archivo existe

filesize(archivo)

Devuelve el tamaño en bytes del archivo.

copy(archivo_origen,archivo_destino)

Devuelve true si pudo copiar el archivo.

ALGUNAS FUNCIONES PARA EL MANEJO DE CADENAS

strlen(cadena) Devuelve la longitud de la cadena.

substr_count(cadena,patron) Devuelve el número de apariciones del patrón en la cadena.

chop(cadena) Devuelve la cadena sin espacios en blanco ni caracteres de fin de línea

trim(cadena) Devuelve la cadena sin los espacios en blanco que se encontaran al inicio y al final de la cadena

strtoupper(cadena) Convierte una cadena a mayúsculas

strtolower(cadena) Convierte una cadena a minúsculas

split(patron,cadena) Devuelve un array, resultado de dividir cadena en subcadenas debido al criterio de separación dado por patrón

explode(patron,cadena) Devuelve un array, resultado de dividir cadena en subcadenas debido al criterio de separación dado por patrón

implode(nexo,cadena) Devuelve una cadena, resultado de unir todos los elementos de un array separados por un nexo.

funcionescadenas.php

```
< ht m l>
< head>
< t itle> FUNCIONES PARA EL MANEJO DE CADENAS< / t it le>
< / head>
< body bgcolor= "#3399cc">
< h1 align= "cent er"> FUNCIONES PARA EL MANEJO DE CADENAS< / h1> < hr>
<?
$scad1= "La rana croa alegre en la pradera";
echo "< cent er> Recorriendo t odos los caracteres de cadena \ $scad1< / cent er> < br> ";
f or($i= 0;$i< st rlen($scad1);$i+ + )
echo $scad1[$i] . "< br> ";
echo "MAYÚSCULAS: " . strt oupper($scad1) . "< br> ";
echo "m inúsculas: " . st rt olower($scad1) . "< br> ";
$scad2= "casa:perro:grande:verde";
$arr1= split (":",$scad2);
f oreach($arr1 as $value)
echo $value . "< br> ";
$arr2= ex plode(":",$scad2);
f oreach($arr2 as $value)
echo $value . "< br> ";
?>
< / body>
< / htm l>
```

PRACTICA DE MANEJO DE ARCHIVOS

1.- Generar un contador de visitas que despliegue el número de visitante, permita en base a esto mostrar para el visitante numero par un estilo de página y para el visitante impar otro estilo diferente.

El visitante 20 tendrá un premio, el cual consistirá en un anunció o promoción.

2.- Generar una tabla que despliegue el contenido del archivo de visitas, el cual mostrará los datos de cada visitante: su nombre, el número de visita y la fecha en que visitó el sitio.