Algoritmos – TUPAR

Archivos

Memoria principal y secundaria

- **Memoria principal**: son circuitos integrados capaces de almacenar información digital, a los que tiene acceso un microprocesador.

- **Memoria secundaria**: es un conjunto de dispositivos periféricos para el almacenamiento masivo de datos.

Memoria principal y secundaria

Principal	Secundaria
Rápida.	Lenta.
Poca capacidad.	Mucha capacidad.
Utilizada para almacenar valores de variables.	Usos múltiples. Almacenan información.
Vólatil. Se pierden los valores al finalizar el programa.	No vólatil. No se pierden los datos al finalizar programa.
Memoria RAM Memoria ROM	CD/DVD Disco Duro Memoria flash

Archivos

Un archivo es una secuencia de datos almacenados en memoria secundaria.

Tienen un nombre y una ubicación dentro del sistema de archivos.

Un programa puede manipular archivos para poder crear,leer o escribir en ellos.

Archivos

Para:

- Guardar resultados.
- Utilizar datos en ejecuciones futuras.
- Comunicación entre programas.
- Configuración de un programa.
- Registro del programa (logs).

Archivos

Tipos (según el lenguaje):

- **Archivos de registros**, que almacenan una secuencia de datos del mismo tipo.
- **Archivos de texto**, que almacenan secuencia de carácteres.

Archivos

Tipos (según el lenguaje):

- **Archivos de registros**, que almacenan una secuencia de datos del mismo tipo.
- **Archivos de texto**, que almacenan secuencia de carácteres.

Archivos en PHP

Flujo de trabajo para archivos en PHP:

- Abrir un archivo
- Manipulación del archivo
- Cerrar archivo.

Archivos en PHP

Flujo de trabajo para archivos en PHP:

- Abrir un archivo
- Manipulación del archivo
- Cerrar archivo.

Archivos en PHP

Abrir un archivo:

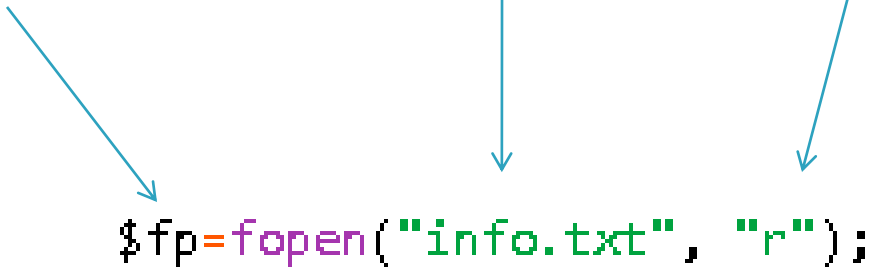
- Especificar nombre del archivo (path absoluto o relativo).
- Especificar tipo del archivo: Para qué lo voy a usar (lectura, escritura).
- Utilizar función fopen(nombre, tipo).

Archivos en PHP

Descriptor: variable que funciona como nexo entre el archivo y el programa

Nombre del archivo

Tipo de uso.



```php
$fp=fopen("info.txt", "r");
```

Archivos en PHP

Tipos de uso:

- r: Sólo para lectura. La lectura comienza al inicio del archivo.
- r+: Para lectura y escritura. Comienza al inicio del archivo.
- w: Sólo para escritura. Comienza al inicio del archivo, y elimina el contenido previo. Si el archivo no existe, intenta crearlo.
- w+: Para escritura y lectura. Comienza al inicio del archivo, y elimina el contenido previo. Si el archivo no existe, intenta crearlo.
- a: Sólo escritura. Comenzará al final del archivo, sin afectar al contenido previo. Si el fichero no existe se intenta crear.
- a+: Para lectura y escritura. Comenzará al final del fichero, sin afectar al contenido previo. Si el fichero no existe se intenta crear.

Archivos en PHP

Flujo de trabajo para archivos en PHP:

- Abrir un archivo
- Manipulación del archivo
- Cerrar archivo.

Archivos en PHP

Al abrir un archivo se crea un cursor que se posará al comienzo o fin del mismo según el tipo de uso con el cual lo cree.

Este cursor servirá para realizar las lecturas y escrituras en el archivo.

Las acciones se realizarán utilizando el descriptor creado al abrir el archivo.

Archivos en PHP

Funciones útiles:

- fgets(descriptor): devuelve un string con la linea apuntada por el cursor y avanza el cursor una linea.
- feof(descriptor): indica si el cursor llegó al final del archivo.
- fwrite(descriptor, cadena): escribe la cadena pasada en el cursor.

Archivos en PHP

Crear un archivo y agregarle 4 lineas con algun texto:

Archivos en PHP

Crear un archivo y agregarle 4 lineas con algun texto:

```php
$fp = fopen("ejercicios\\archivo.txt", "w");
fwrite($fp, "linea 1 \n");
fwrite($fp, "linea 2 \n");
fwrite($fp, "linea 3 \n");
fwrite($fp, "linea 4 \n");
```

Archivos en PHP

Mostrar por pantalla todas las lineas que contiene un archivo:

Archivos en PHP

Mostrar por pantalla todas las lineas que contiene un archivo:

```php
$fp = fopen("ejercicios\\archivo.txt", "r");

while (!feof($fp)){
    $linea=fgets($fp);
    echo $linea;
}
```

Archivos en PHP

Flujo de trabajo para archivos en PHP:

- Abrir un archivo
- Manipulación del archivo
- Cerrar archivo.

Archivos en PHP

Una vez que termino de usar el archivo, debo cerrarlo.

Para ello:

- fclose(decriptor).

Debe haber un fclose por cada fopen.

Archivos en PHP

Mostrar por pantalla todas las lineas que contiene un archivo:

```php
$fp = fopen("ejercicios\\archivo.txt", "r");

while (!feof($fp)){
    $linea=fgets($fp);
    echo $linea;
}

fclose($fp);
```

Ejercicio de ejemplo

Se tiene un archivo en el que cada linea tiene un valor entero.

Sumar todos los valores y registrar el resultado en otro archivo

Algoritmos – TUPAR

Separación de funcionalidades

Separación de código en archivos

Es una buena práctica de programación realizar una separación física del código por funcionalidades a implementar

Separación de código en archivos

Si definimos todo el código en un solo archivo se vuelve tedioso de leer

Difícil de analizar, corregir y encontrar funcionalidad.

División de código

- PHP da la posibilidad de codificar en archivos diferentes y vincularlos.
- Buenas prácticas:
  - Separar por algún criterio diferentes funcionalidades del programa en desarrollo.
  - Dejar en el programa principal sólo los llamados a los metodos/funciones/procedimientos definidos en el resto.
- Vinculación de archivos mediante include, require, require_once.

División de código

- Include: incluye el archivo dado, de no existir tira warning.

  include(“archivo.php”);

- Require: incluye el archivo dado, de no existir tira error.

  require(“archivo.php);

- Require_once: incluye el archivo dado sólo si no ha sido incluido antes, de no exisitir tira error.

  require_once(“archivo.php);

División de código

- Ej: Se necesitan programar 3 estructuras dierentes y funciones para manejar archivos: