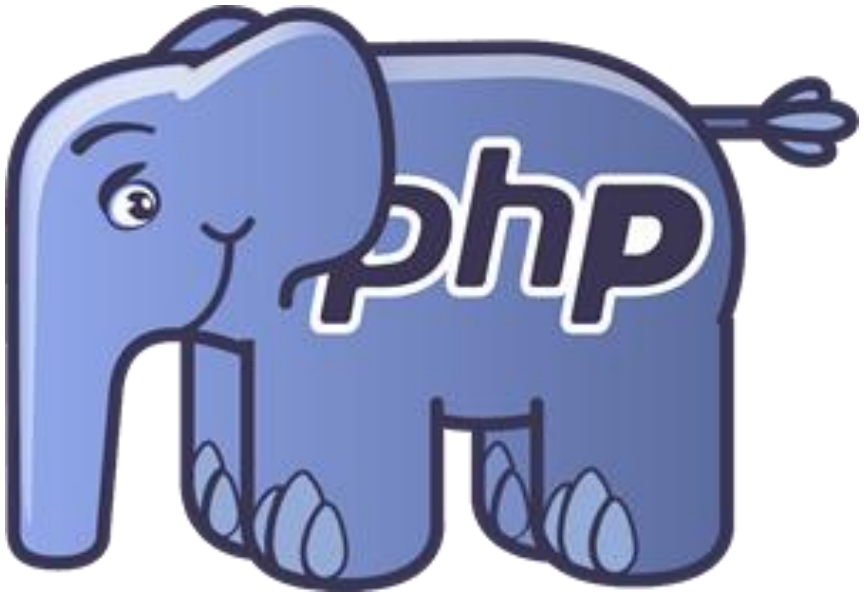
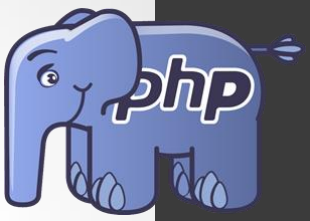




PHP

Programació en el servidor

https://www.php.net/manual/es/
https://www.w3schools.com/php/php_ref_overview.asp

ENTORN DE TREBALL

XAMPP (instal·lar com admin)

Entorn de desenvolupament web XAMPP :
https://www.apachefriends.org/es/index.html

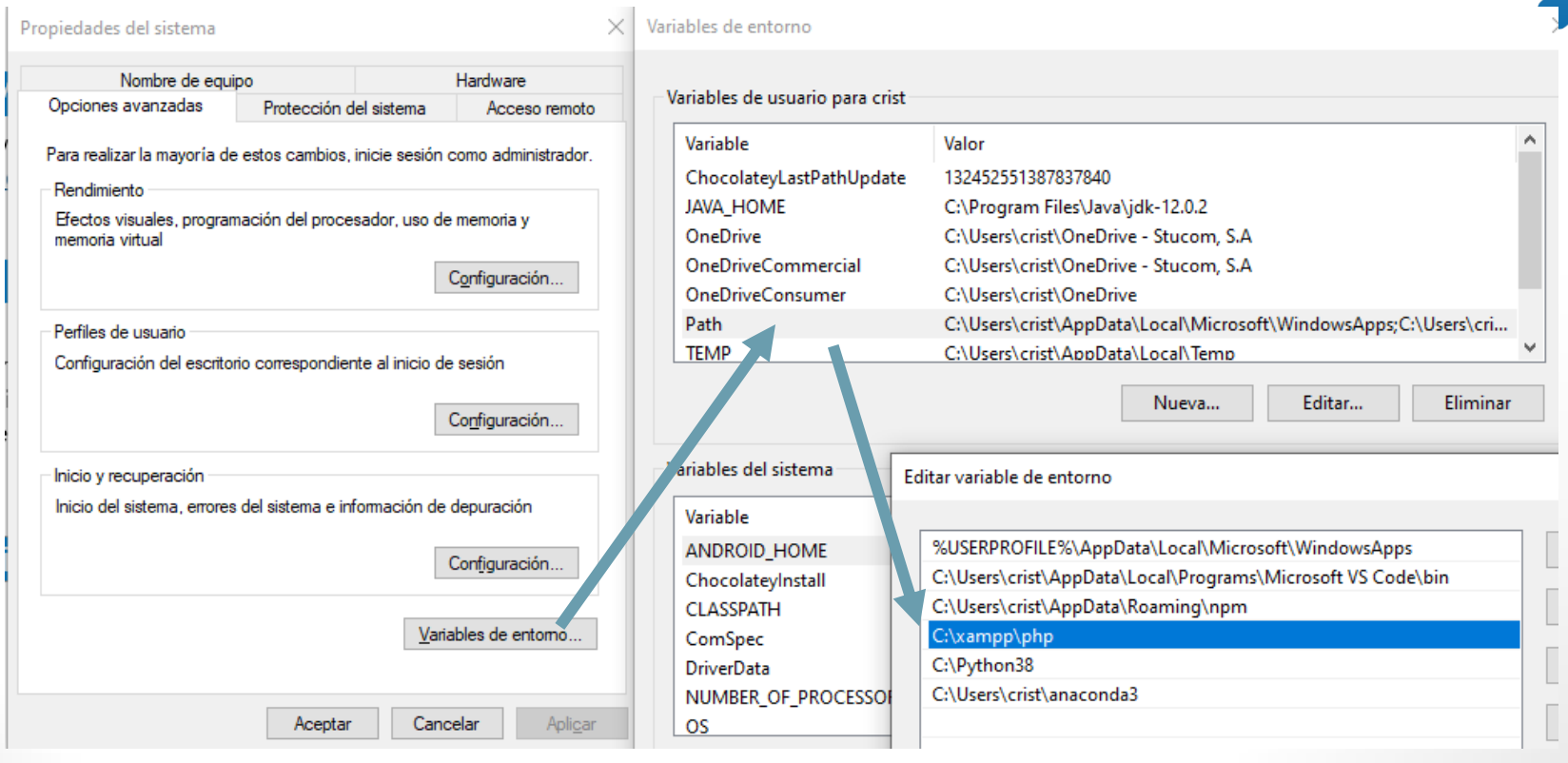
Visual Studio Code

IDE Visual Studio Code: https://code.visualstudio.com

Plugins (extensions):

- Format HTML in PHP
- XML Language Support by Red Hat

Variables entorn del sistema



MATAR PROCESOS

Step 1:

- Open up cmd.exe (note: you may need to run it as an administrator, but this isn't always necessary), then run the below command:

  netstat -ano | findstr :<PORT>

- (Replace <PORT> with the port number you want, but keep the colon)

```
C:\WINDOWS\system32>netstat -ano | findstr :8080
  TCP    0.0.0.0:8080           0.0.0.0:0              LISTENING       3740
  TCP    [::]:8080              [::]:0                 LISTENING       3740
```

- The area circled in red shows the PID (process identifier). Locate the PID of the process that's using the port you want.

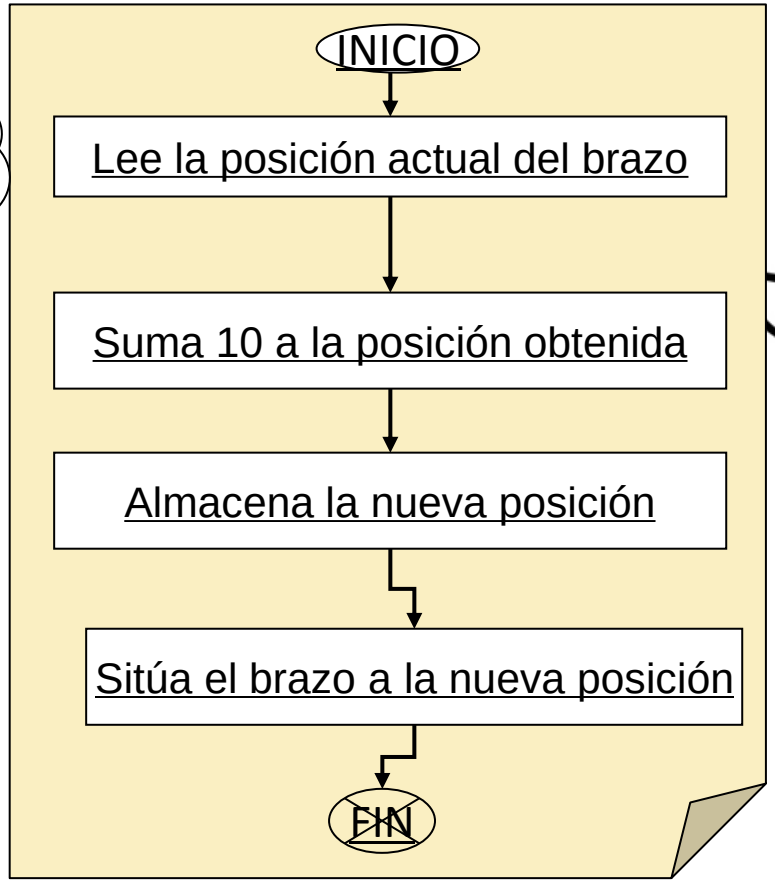
Step 2:

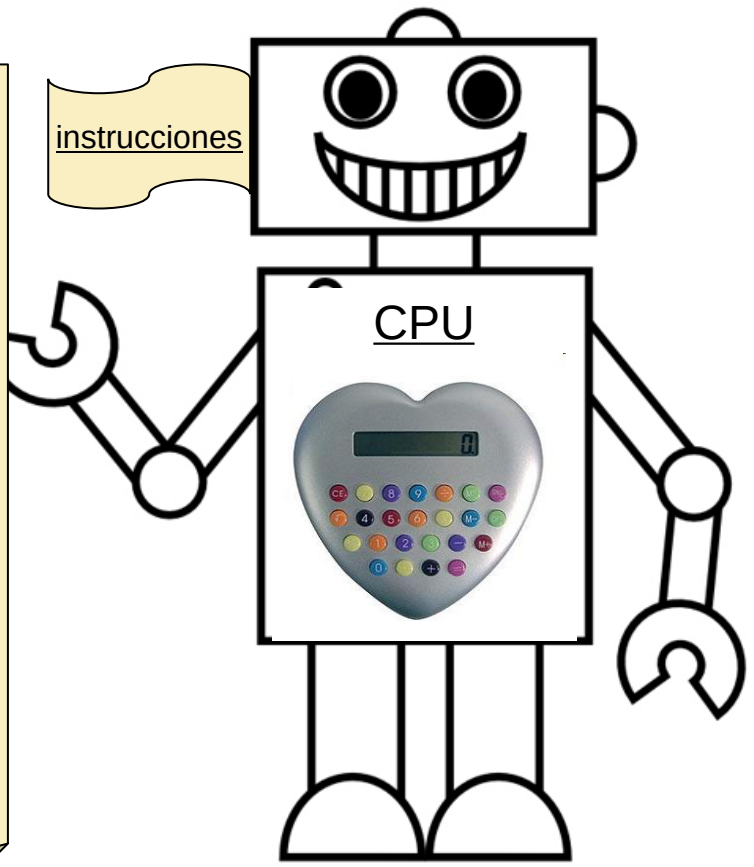
- Next, run the following command:
- taskkill /PID <PID> /F
- (No colon this time)

```
C:\WINDOWS\system32>taskkill /PID 3740 /F
SUCCESS: The process with PID 3740 has been terminated.
```

Objectius de la programació

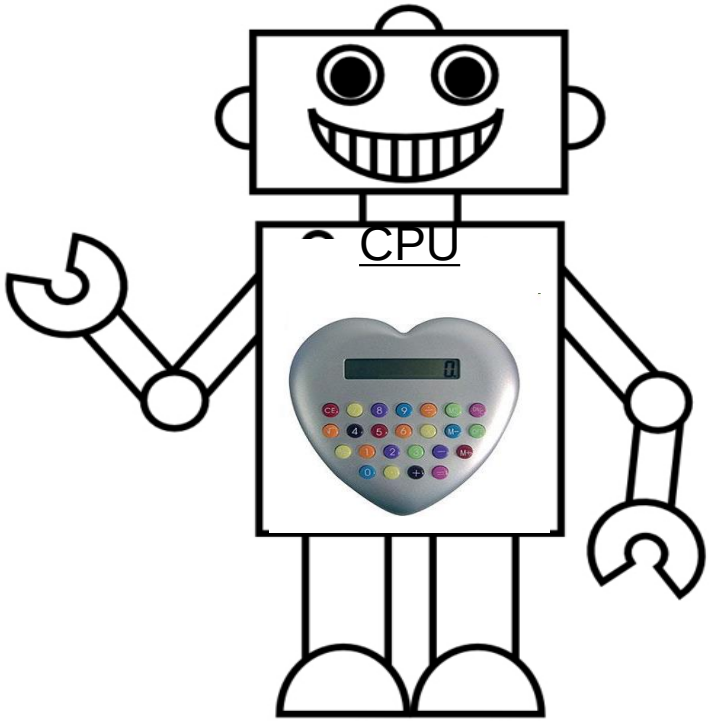
- Automatitzar operacions creant programes.
- Els programes s'executen seqüencialment.
- Un programa sempre te un inici i un final.





FASES DE LA PROGRAMAICÓ

- Dividir el problema en fraccions ordenades.
- Començar a programar la primera fracció
  - Identificar les dades d'entrada
  - Identificar quina és la sortida que volem aconseguir
  - Pensar les accions i fórmules a aplicar per aconseguir-ho.
  - Indicar a l'ordinador les accions i fórmules pensades
  - Executar el programa.
  - Revisar la sortida obtinguda i si no s'ajusta a la desitjada revisar si l'error està en el codi o en la lògica
  - Revisar el codi i tornar a executar...



Objectiu de programar en PHP

ARQUITECTURA CLIENT/SERVIDOR

- La www se basa en una arquitectura de n capas

PHP

BASES PROGRAMACIÓ

- Llenguatge de programació interpretat en el servidor.
- El servidor llegeix el document.php i executa el codi escrit entre els tags: **<?php** i **?>**
- Declarem una variable amb un $ seguit del nom de la variable
- El contingut entre comes simples ' ' sempre es tractarà com a text.
- El contingut entre comes dobles " " es tractarà com a text excepte quan continguin una variable.
- Les instruccions PHP acaben sempre amb **;**
- La instrucció **echo() ;** indica al servidor que escrigui al document el valor entre ().
- El text contingut entre /* i */ o a la dreta de **//** no s'interpretarà

VARIABLES

- Una variable es un espai de memòria identificat per un nom i que pot emmagatzemar diferent tipus d'informació.
- Les variables poden guardar números, text (anomenats *strings*), o estructures de dades més complexes.
- Assignem un valor a una variable amb l'operador =
  - **$variableEx = 10; $variableEx = "hola!"; $variableEx1='adeu';**
- Podem destruir una variable amb la instrucció unset():
  - **unset($variableEx1);**
- En qualsevol moment podem utilitzar una variable o un valor:
  - $num1=10; $resul= 5 + $num1;
- Podem concatenar strings (texts) amb el punt:
  - **$nom="Fina";**
    $cognom="Power";
    $nom_cogn=$nom." - ".$cognom;

OPERADORS

ARITMÈTICS

- **suma (+)**
- **resta (-)**
- **multiplicació (*)**
- **divisió (/)**
- **resto de la divisió (%)**

Binaris $x=5+3; — **x** és el resultat de sumar 5 més 3

D'assignació $x +=5; — **x** és el resultat de sumar **5** més el valor de **x**

Unaris $x++; — **x** és el resultat de sumar el valor de **x** més **1**

```
$variable01 = 5;
$variable01 = 10 + $variable01;
```

COMPARACIÓ

- Retornen SI o NO.
- Ens permeten fer comparacions entre dos valors.
  - major (>)
  - menor (<)
  - igual (==)
  - diferent (!=)
  - major o igual (>=)
  - menor o igual (<=)

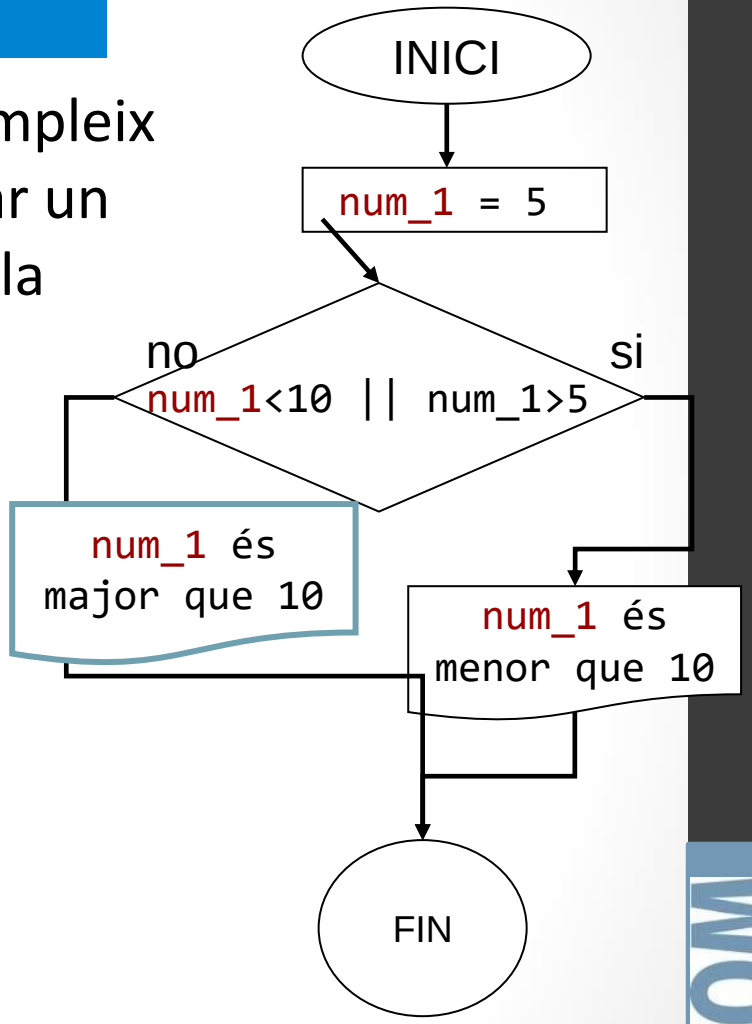
LÒGICS

- Permeten concatenar dos o més comparacions
- AND (&&) Retorna SI , si les dos comparacions son certes.
- OR (||) Retorna SI , si alguna de les dos comparacions és certa.
- NEGACION (!) Si el resultat es SI, retorna NO i viceversa.

ESTRUCTURES DE CONTROL

IF/ELSE

- Indica un bloc de codi a executar si es compleix una condició. Opcionalment es pot indicar un bloc de codi a executar si no es compleix la condició




```php
$num_1=10;
if($num1 <10){
    echo $num1." és menor a 10";
}else{
    echo $num1." és major a 10";
}
```

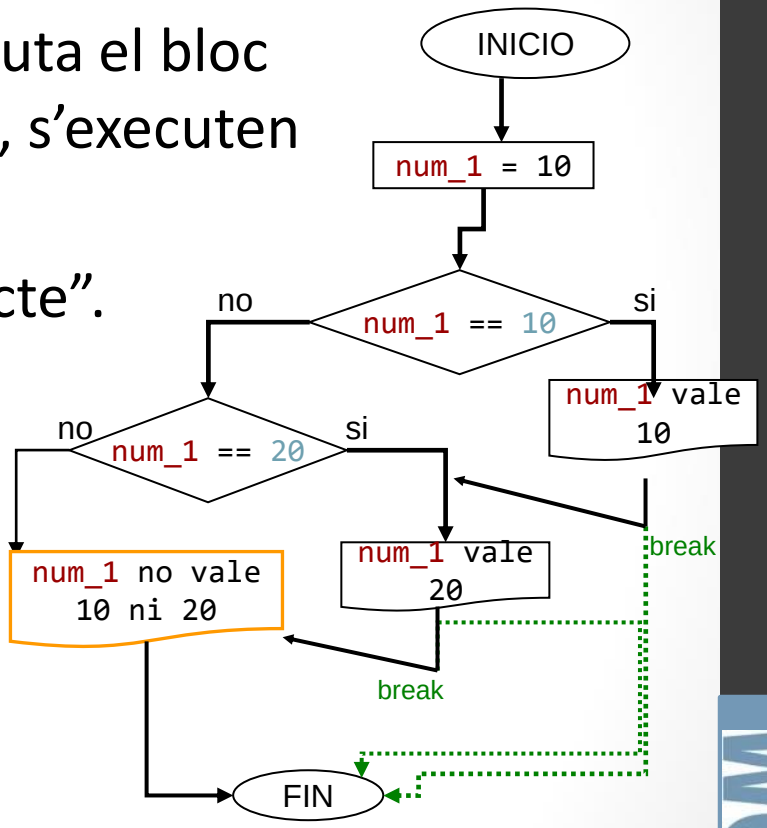
ESTRUCTURES DE CONTROL

SWITCH

- Compara una mateixa variable amb múltiples valors.
- Quan es compleix una comparació s'executa el bloc de codi associat. Si no s'indica el contrari, s'executen també la resta de blocs de codi.
- Permet indicar un bloc de codi "per defecte".

```
<?php
$num1=10;
switch($num1){
   case 10:
      echo $num1." val 10";
   break;
   case 20:
      echo $num1." val 20";
   break;

   default:
   echo $num1." no val ni 10 ni 20";
}
```



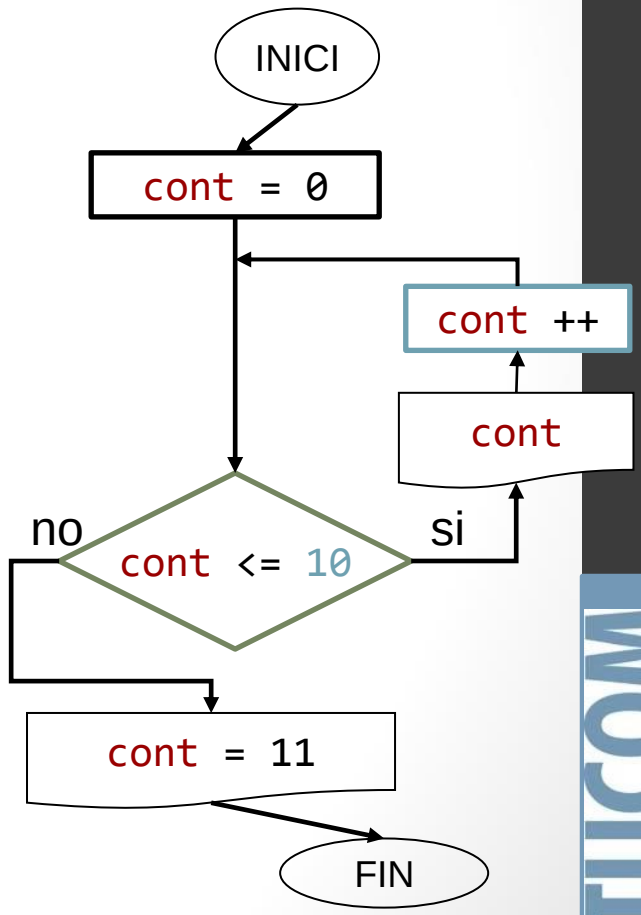
ESTRCTURES DE REPETICIÓ

WHILE i FOR

- Permet executar un bloc de codi repetidament mentre es compleixi una condició.
- Totes les estructures de repetició necessiten:
  - El codi a repetir.
  - La condició per finalitzar la repetició.
  - Les instruccions per modificar la condició.

```php
$cont = 0;
while ($cont <= 10) {
   $cont++;
}
echo $cont;
```

```php
for(int $cont =0; $cont<=10; $cont++){
   echo $cont;
}
 echo $cont;
```

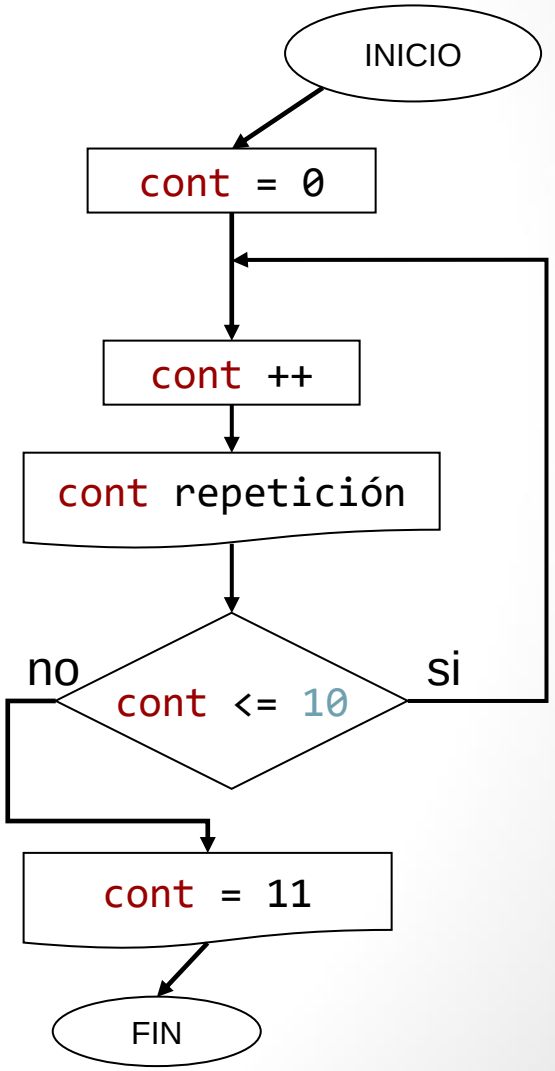


ESTRUCTURES DE REPETICIÓ

DO / WHILE

- És una estructura de repetició que executa el bloc a repetir una vegada y després comprova si l'ha de tornar a executar.

```php
$cont = 0;
do {
    $cont++;
    echo $cont;
} while ($cont <= 10);
echo $cont;
```



TREBALL AMB DATES

- Per obtenir la data, primer hem d'establir la zona horària amb la funció date_default_timezone_set()
- La funció date() ens retornará temps actual amb el fomat que li indiquiem entre paréntesis
- El següent exemple mostra el: any-dia-mes segons la zona horària de "Los Angeles

```php
<?php
    date_default_timezone_set('America/Los_Angeles');
    $data1 = date("Y-d-m");
    echo $data1;
?>
```

- Llistat de les zones horàries disponibles
  - http://php.net/manual/en/timezones.europe.php
- Llistat de les opcions de la funció date():
  - http://php.net/manual/en/function.date.php

PARÀMETRES

- Els paràmetres ens permeten passar valor a un programa PHP.
- S'escriuen amb la sintàxis:
  - nom_parametre1 = valor&nom_parametre2=valor&...
- Podem rebre paràmetres de diferents mètodes:

Mètode GET :

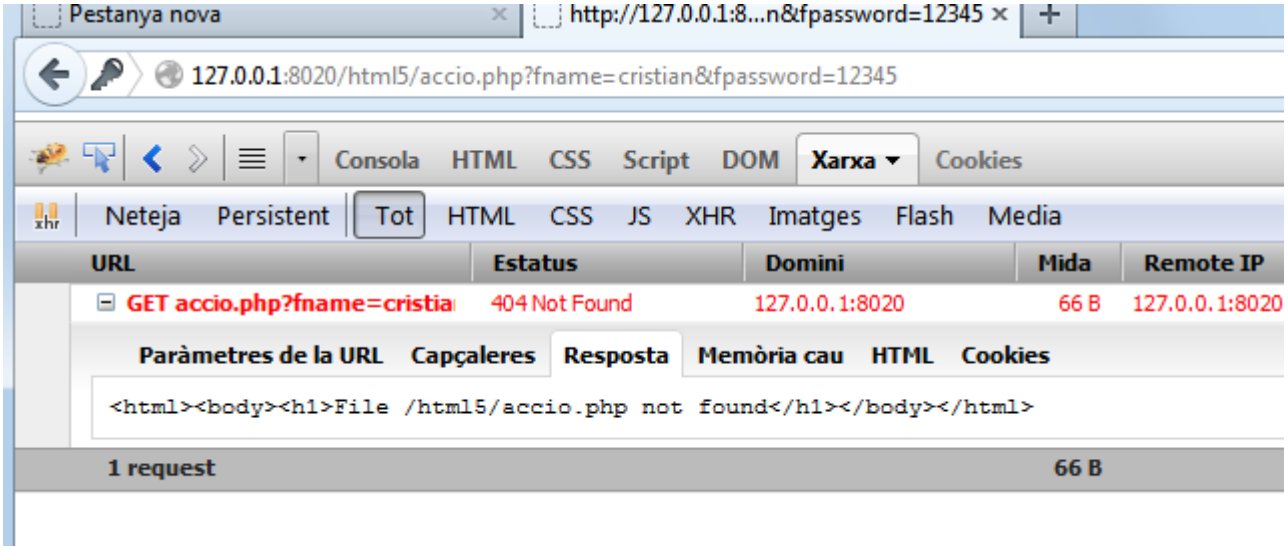
- Passa els paràmetres afegint-los al final de la URL separats per un ?
  - http://exemple.php? nom_parametre1 = valor&nom_parametre2=valor
- El PHP els guarda dins un array $_GET en una posició igual al nom del paràmetre:
  - $_GET[“nom_parametre1”];

Mètode POST :

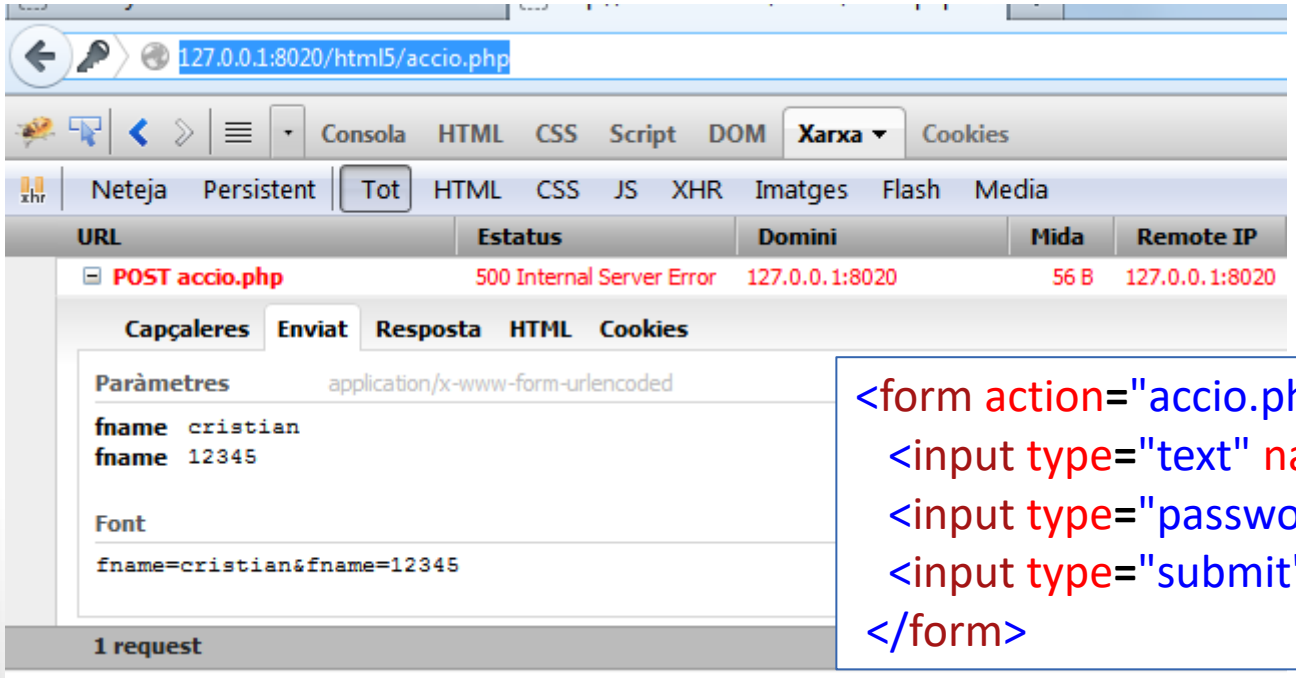
- Passa els paràmetres en la capçalera de la petició.
- El PHP els rep en un array de nom $_POST i els guarda en una posició igual al nom del paràmetre:
  - $_POST[“nom_parametre1”];

GET & POST

GET :



POST :




```
<form action="accio.php" method="post">
  <input type="text" name="fname">
  <input type="password" name="fpassword">
  <input type="submit">
</form>
```

FORMULARIS

- Els formularis ofereixen una interfície per passar paràmetres.
- Els elements principals que formen un formulari són:

FORM :

- Cada form permet generar una petició a un recurs.
- Els seus principals atributs són:
  - Action: adreça del document .php que volem obtenir
  - Method: com ho enviem?
    - Podem utilitzar el mètode “GET” o el mètode “POST” (entre d’altres)

INPUT :

- Cada input representa un paràmetre amb el seu nom y valor.
- Tots els inputs d’una mateixa petició s’han d’incloure dins un mateix tag form
- Els seus principals atributs són:
  - Type: com es mostra l'entrada de dades al formulari
  - Name: nom associat al valor que s'enviará

```
<form action="accio.php"  method="get">
  Nombre:<input type="text" name="fname"> <br />
  Passwor:<input type="password" name="fpassword"><br />
  <input type="submit">
 </form>
```

https://www.w3schools.com/tags/tag_form.asp

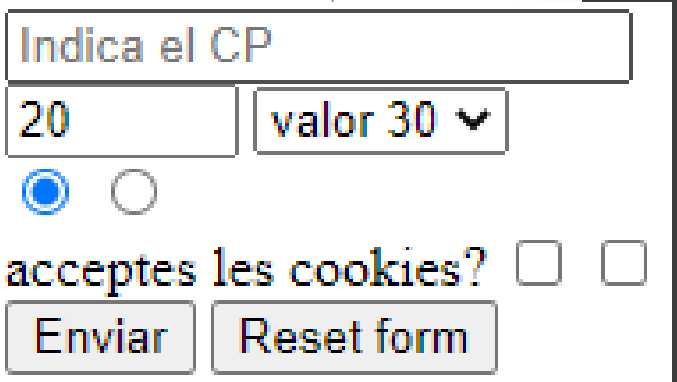
FORMULARIS : exemple

- Existeixen múltiples tipus de inputs. Afecten sobretot a com es mostra la interface a l'usuari. El valor de un input sempre s'envia com a text.

```
<form action="exempleForm.php" method="POST">
   <input type="number" value="20" name="n1" min="0" max="99">
   <input type="text" name="nom" placeholder="Indica el CP"
        required pattern="((0[1-9]|5[0-2])|[1-4][0-9])[0-9]{3}">

   <select name="n2">
      <option value="55"> valor 55</option>
      <option value="30" selected> valor 30</option>
   </select>

   <input type="radio" name="joc" value="damas" checked>
   <input type="radio" name="joc" value="escacs">
   <label> acceptes les cookies?
       <input type="checkbox" name="acceptar_cookies" value="si" />
   </label>
   <input type="checkbox" name="acceptar_propaganda" value="si" />
   <input type="submit">
   <input type="reset" value="Reset form">
</form>
```



https://www.w3schools.com/tags/tag_form.asp

INPUTS I ATRIBUTS

Els atributs dels inputs ens permeten afegir restriccions, valors per defecte, o habilitar-los y deshabilitar-los. Alguns atributs són específics segons el tipus de submit.

Alguns dels principals atributs són:

- **checked** : preselecciona un input checkbox o radio .
- **disabled** : deshabilita un input. No es pot utilitzar, seleccionar o clicar.
- **readonly** : un input no es pot modificar.
- **autofocus** : al carregar la pàgina mou el focus a l'element seleccionat.
- **maxlength**: número máxim de caracters que pot tenir un input tipus text
- **placeholder** : text que es mostra dins un input mentre estigui buit.
- **autocomplete** : permet o no predir els possibles valors d'un input a mesura que l'usuari escriu
- **required** : indica que un input és necessari per enviar el fomulari.
- **pattern** : permeten validar un input text a partir d'una expressió regular
- **form** : indica que un input forma part d'un formulari (encara que estigui fora)
- **multiple** : permet seleccionar varis valors dins un input deltipus file i select

https://www.w3schools.com/tags/tag_input.asp

ATRIBUTS DEL FORM I SUBMIT

El tag Form permet especificar alguns atributs:

- **action:** l'adreça del recurs que volem demanar
- **method:** permet indicar si enviem els parámetros per POST o GET
- **novalidate** : evita que es validin els camps del formulari (mail, url, number..).
- **autocomplete** : es recarreguen els valors dels inputs.

Per aquells inputs del tipus submit podem indicar atributs específics:

- **formaction** : al clicar sobre el input submit s'envia el formulari a l'adreça indicada
- **formenctype** : indica amb quina codificació s'enviaran les dades del formulari
- **formmethod** : get o post
- **formnovalidate** : no s'han de validar els inputs (HTML5)
- **formtarget**: indica si s'ha d'obtenir la resposta del formulari en la mateixa pàgina o en una nova.

```
$k = $_GET["p"]? $_GET["p"]:"b";
```

https://www.w3schools.com/tags/tag_input.asp

ENVIAR ARXIUS

- El formulari ha de ser POST y multipart/form-data
- Obtenim l'arxiu amb $_FILES
- Guardem l'arxiu amb move_uploaded_file()

```
$filename = $_FILES['arxiu']['name'];

$location = "upload/" . $filename;
$imageFileType = pathinfo($location,
                          PATHINFO_EXTENSION);

if (move_uploaded_file(
   $_FILES['arxiu']['tmp_name'],
   $location
)) {
   $rutaImg = $location;
} else {
   $rutaImg = false;
}
echo "Img en:" . $rutaImg;
```

```
<form method="POST"
      enctype="multipart/form-data">
  <input type="text" name="param1" />
  <input type="file" name="arxiu" />
  <input type="submit">
</form>
```

```
echo " Parametres: " . $_POST["param1"];
if ($imageFileType != "jpg" && $imageFileType != "png") {    echo "format no es jpg ni png"; }
if (file_exists($target_file)) {    echo "Existeix un arxiu amb el mateix nom";  }
if ($_FILES["arxiu"]["size"] > 500000) {    echo "Arxiu superior a 500KB."; }
```