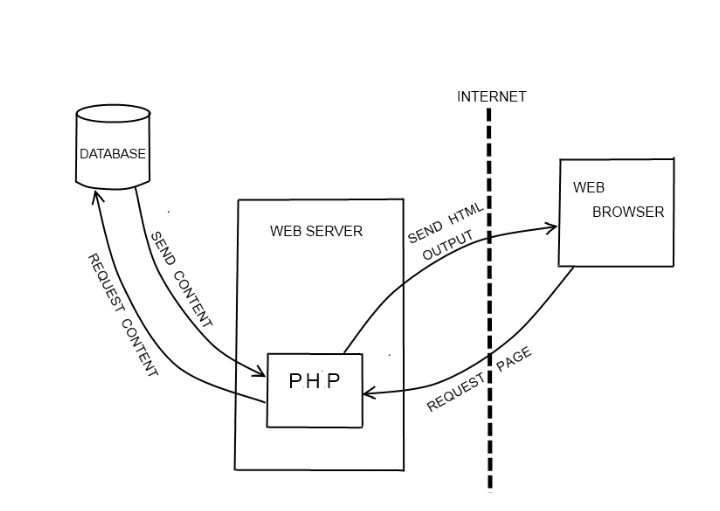
# **PHP 2**

[[1]](#footnote-0)

## **XAMPP**

Egy nyílt forráskódú, platform-független web szerver-csomag az Apache által kifejlesztve, ami tartalmaz többek között adatbázis, illetve php szervert és PHP interpretert.

## **Véletlen számok**

Véletlen pozitív egész szám:

echo(rand() . "<br>");

A véletlen szám maximuma:

echo "A véletlen szám maximuma: <br>".getrandmax(). "<br>";

10 és 13 közötti véletlen szám (*zárt intervallum* mindkét oldalon!):

echo(rand(10,13) . "<br>");

## [**Szuper globális változók**](https://www.w3schools.com/php/php_superglobals.asp)

Beépített, bárhonnan elérhetőek a programból.

* $GLOBALS - ő a főnök
* $\_SERVER
* $\_SESSION
* $\_COOKIE
* $\_REQUEST
* $\_POST
* $\_GET

$GLOBALS szintaxisa: $GLOBALS[index], ahol az index lehet szöveges típusú is, azaz asszociatív tömb elemeit bővítjük.

A $\_SERVER egy szuperglobális, amely a PHP szkriptek fejléceit, elérési útjait és helyeit tárolja.

A $\_REQUEST változó egy PHP szuperglobális, amely adatgyűjtésre szolgál egy űrlap elküldése után. Alapértelmezés szerint tartalmazza a $\_GET, $\_POST és még a $\_COOKIE tartalmát is.[[2]](#footnote-1)

## Példa $GLOBALS használatára

$x= 7;

$y = 6;

$k;

function myTest0() {

//nem működik:

$y = $x + $y;

// működik:

// $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];

}

myTest0();

## Tömb létrehozása (és kiíratása)[[3]](#footnote-2)

$a = array("red", "green", "blue");

print\_r($a);

$tomb0 = array(1,2,3,4,5);

$tomb1 = array();

$tomb2[0] = "Ady Endre";

$tomb2[1] = "József Attila";

$tomb3 = ["Milan Kundera", "Hermann Hesse"];

Hozzáfűzés kétféleképpen:

$tomb2[] = "Tóth Árpád";

array\_push($tomb2, "Tóth Árpád");

Tömb elemszáma:

$elemSzam = count($tomb2);

Tömbök összeillesztése:

$tomb4 = array\_merge($tomb2, $tomb3);

### Asszociatív tömb (dictionary)

$b = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

print\_r($b);

$tomb1["elso"]="Ady";

$tomb1["masodik"]="József";

Módosítás és létrehozás ugyanúgy:

$tomb1["masodik"]="Karinthy";

### Tömb mint konstans

define('ANIMALS', array(

'dog',

'cat',

'bird'

));

echo ANIMALS[1];

## Operátorok

//+-\*/%, && , ||

echo (8/3)."<br>";

//matematikai programcsalád

echo floor(8/3)."<br>";

Konkatenáció jele: .

Típus lekérdezése

echo gettype($hamis);

## Vezérlési szerkezetek

### Elágazás\_1:

if ($hamis) {

...

}

elseif ($feltetel){

...

}

else {

...

}

### Elágazás\_2:

$logikai = true;

echo ($logikai ? 'Ez igaz.' : 'Ez hamis.');

### Elágazás\_3:

switch ($valtozo) {

case $value1:

...

break;

case $value2:

...

break;

default:

...

break;

}

### Ciklus\_1:

for ($index1 = 0; $index1 < count($tomb); $index1++) {...

}

### Ciklus\_2:

foreach ($tomb2 as $value) {

echo $value." ";

}

### Ciklus\_3 (asszociatív tömb bejárása):

foreach ($tomb0 as $kulcs => $ertek) {

echo $kulcs. ": ".$ertek." <br>";

}

### Ciklus\_4:

$index1 = 0;

while ($index1<10){

echo $index1." ";

$index1++;

}

### Ciklus\_5:

$x = 1;

do {

echo "A szám: $x <br>";

$x++;

} while ($x <= 5);

## Undefined or null?

A PHP-ben jellemzően a be nem állított vagy hatástalanított változók null-nak számítanak. A null jelentése "nincs érték". Egyértelmű különbség van a „nincs érték” és az üresen hagyott érték között. Például, ha egy felhasználó beküldött egy űrlapot a foo=”” értékkel, akkor a $\_GET['foo'] az üres "" karakterlánc értékére lesz beállítva, ami határozottan különbözik a null-tól.

Mindezek ellenére megtudhatjuk, hogy egy változó soha nem volt-e beállítva vagy hatástalanítva, bár az is\_null értéke mindig igaz lesz.

Az is\_null az isset *negatívja*, azzal a kivétellel, hogy figyelmeztetést küld, ha az érték soha nem lett beállítva.

Egyik módszer:

echo **array\_key\_exists**( $variableName, $theArray ) ? 'variable was set, possibly to null' : 'variable was never set';

Példa 1:

$x = 75;

echo array\_key\_exists( 'x', $GLOBALS ) ? 'variable was set, possibly to null' : 'variable was never set';

eredménye:

variable was set, possibly to null

Példa 2:

echo array\_key\_exists( 'nincsIlyen', $GLOBALS ) ? 'variable was set, possibly to null' : 'variable was never set';

eredménye:

variable was never set

## Null coalescing operator

$username = $\_GET['user'] ?? 'nobody';

// ekvivalens ezzel:

$username = isset($\_GET['user']) ? $\_GET['user'] : 'nobody';

Visszaadja az első operandust, ha létezik, és nem null; ellenkező esetben a második operandust adja vissza.

Láncba is fűzhető:

$username = $\_GET['user'] ?? $\_POST['user'] ?? 'nobody';

//mit jelenthet?

## Spaceship operator

Az űrhajó operátor két kifejezés összehasonlítására szolgál. -1, 0 vagy 1 értéket ad vissza, ha $a kisebb, egyenlő vagy nagyobb, mint $b. Az összehasonlítások a PHP szokásos típus-összehasonlítási szabályai szerint történnek. Számokra és szövegre is működik.

// Integers

echo 1 <=> 1; // 0

echo 1 <=> 2; // -1

echo 2 <=> 1; // 1

// Strings

echo "a" <=> "a"; // 0

echo "a" <=> "b"; // -1

echo "b" <=> "a"; // 1

## Static (az értéke megmarad a fg-hívás után is)

function myTest() {

static $xx = 0;

echo "xx (static) értéke: $xx";

$xx++;

}

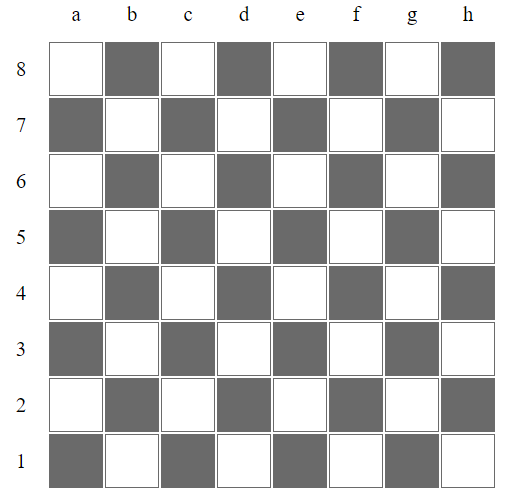
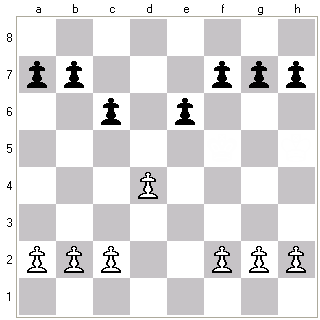
for ($i = 0; $i < 5; $i++){

echo "<br>".myTest();

}

## Feladatok

1. Hozz létre egy 5 elemű tömböt véletlen számokkal (20 és 30 közötti zárt intervallumban), majd írasd ki az összegüket!
2. Adott egy 3 elemű tömb véletlen érdemjegyekkel korábban feltöltve, majd írasd ki az érdemjegyekhez tartozó szöveges értelmezést (pl. 3: közepes)
3. Hozz létre fejIras nevű logikai tömböt ($darabSzam darabszámú) véletlen értékekkel!
   1. Számold meg a fejek számát, és írasd ki az eredményt!
4. Készítsd el egy sakktábla képét figurák nélkül (segítség: a kis “a” betűre így hivatkozhatsz: chr(97))!



1. <https://github.com/xeroxism> [↑](#footnote-ref-0)
2. <https://www.section.io/engineering-education/php-superglobals/> [↑](#footnote-ref-1)
3. <https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_var_print_r> [↑](#footnote-ref-2)