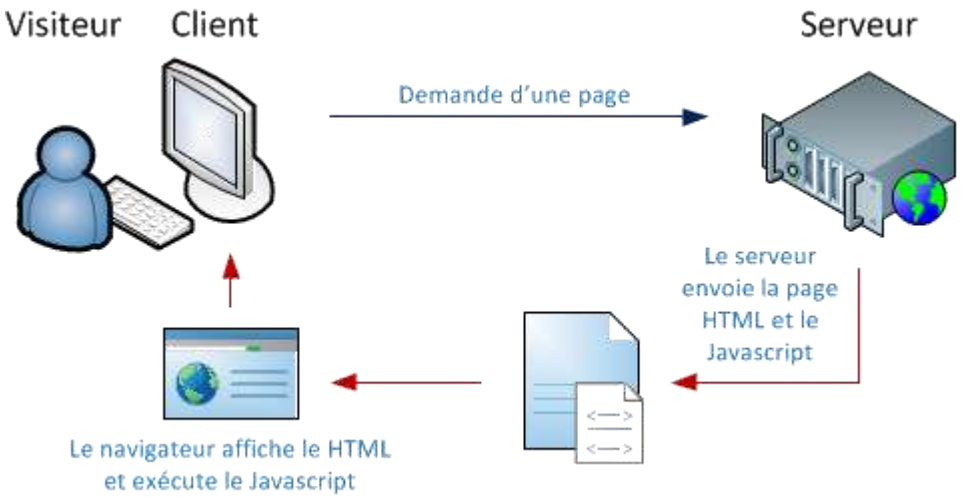
PHP

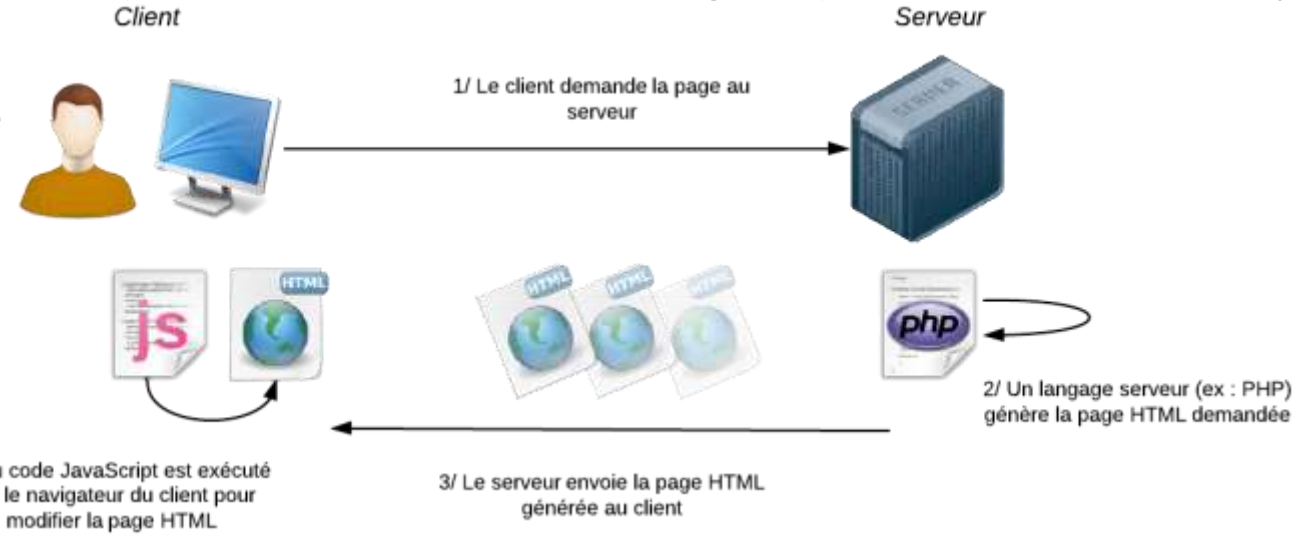
UVOD U PHP

- PHP je skriptni jezik opće namjene sa korijenima u jeziku C
- Pomoć web programerima u stvaranju dinamičkih web stranica
- PHP skripte ugrađuju se unutar HTML dokumenta (web server pomoću PHP interpretera izvršava ugrađeni PHP kod generirajući web stranicu sa HTML kodom)



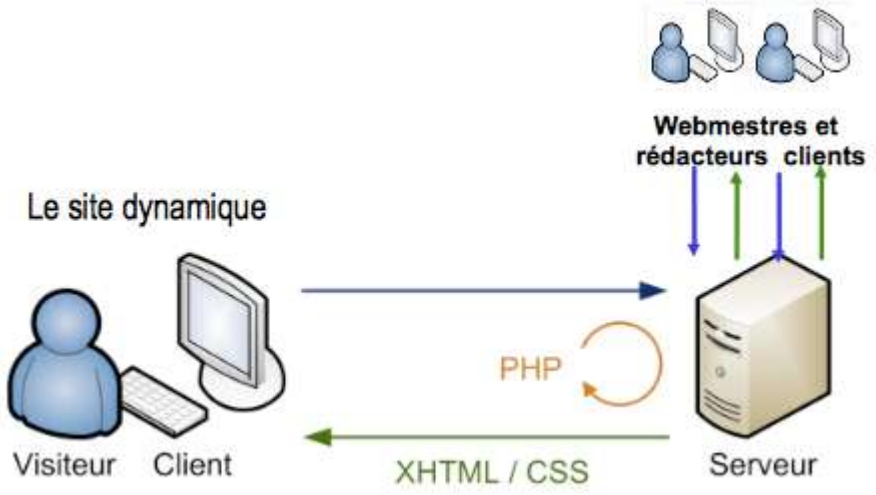
POVIJEST RAZVOJA

- Rasmus Lerdorf (1995.) – za praćenje broja posjetitelja i druge administracije web stranice (skripte je nazvao Personal Home Page Tool), moduli za komunikaciju s bazom omogućili jednostavno razvijanje dinamičkih web stranica ili aplikacija...
- Prva inačica objavljena pod PHP/FI (Personal Home Page Forms Interpreter)
- 1997. objavljena druga inačica
- PHP 3 (Andi Gutmans i Zeev Suraski) – mogućnost upotrebe ekstenzija za proširenje osnovih funkcionalnosti (ekstenzije za spajanje na razne baze podataka, funkcije za rad sa datumima, za manipulaciju nizovima znakova itd.) uvođenje objektno orijentirane sintakse
- 1999. objavljena inačica 4 – jezgra Zend Engine
- 2004. predstavljen PHP 5.0



RAZVOJNO OKRUŽENJE

- uređivač teksta (bilo koja aplikacija za pisanje teksta, PHP kod sprema se u datoteku s .php ekstenzijom)
- preglednik web stranica (služi kao komunikacijski kanal između posjetitelja i web poslužitelja, prikazuje HTML sadržaj)
- web poslužitelj (aplikacija instalirana na poslužitelju na kojem su smještene stranice, izvršava kod i šalje rezultate posjetitelju u web pregledniku)



PHP kod se izvršava ISKLJUČIVO na strani poslužitelja, korisnik u web preglednik dobiva gotov HTML kod

PRIMJER JEDNOSTAVNE PHP SKRIPTE KOJA DINAMIČKI STVARA HTML TABLICU

```
$polje = array(1=>'Tesla', 2=>'Edison',3=>'Bell');
echo '<table border=„1" cellpadding=„5">';
foreach($polje as $key=>$val)
{
echo '<tr>
        <td>'.$key.'</td>
        <td>'.$val.'</td>
      </tr>';
}
echo '</table>
```

HTML KOD PREGLEDNIKA

```
<table border=„1” cellpading=„5”>
<tr>
        <td>1</td>
        <td>Tesla</td>
</tr>
<tr>
        <td>2</td>
        <td>Edison</td>
</tr>
<tr>
        <td>3</td>
        <td>Bell</td>
</tr>
</table>
```

WEB POSLUŽITELJ - APACHE

- Web poslužitelj:
  - aplikacija instalirana na serveru čiji je zadatak izvršavanje skripti i slanje stranica posjetiteljima
  - veza između datoteka u kojima se nalaze skripte i web preglednika posjetitelja naše stranice
- Najpoznatiji web poslužitelj je Apache
  - omogućuje smještaj više različitih stranica na jednom fizičkom poslužitelju
- XAMPP paket za razvoj aplikacija, pretvara računalo u mali web poslužitelj
  - X označava da mogu raditi na različitim operativnim sustavima
  - A označava web poslužitelj Apache
  - M je MariaDB MySQL baza podataka
  - P je za PHP
  - P je za Perl

OSNOVNA PRAVILA PISANJA PHP KODA

- PHP skriptni jezik uključuje PHP kod u HTML kod stranice. Nakon obrade web poslužitelj dio s PHP kodom zamjenjuje dobivenim HTML rezultatom i zajedno sa ostatkom HTML dokumenta šalje korisniku u njegov web preglednik
- Unutar dokumenta treba posebno naznačiti početak i završetak PHP koda

```
<?php
//.....
?>
```

- Web poslužitelj pri obradi skripti prepoznaje ove oznake i izvršava samo kod unutar njih, dok ostatak zanemaruje i tretira kao običan HTML koji samo prosljeđuje posjetitelju
- Primjer uključivanja PHP koda: <?php echo 'PHP tekst'; ?>
- echo u PHP-u služi za ispis teksta

KOMENTIRANJE KODA

- Dio programskog koda koji se ne izvršava
- Ubrzavaju snalaženje kad je potrebno izvesti promjene ili dorade

```
# Ovo je komentar od znaka do kraja retka
ili
// Ovo je komentar sve do kraja retka
/*
Ovo je jedan
malo duži komentar
koji se proteže u više linija
*/
```

- Označavanje pogodno za veće opise koji se protežu kroz više redova

PHP - VARIJABLE

OSNOVNI TIPOVI PODATAKA U PHP-U

- Cjelobrojne vrijednosti - skup cijelih brojeva
- Brojevi s pokretnim zarezom - realni brojevi
- Nizovi znakova – znakovi neograničene duljine: $oznaka='123-487/898-5' (ili „)
- Logičke vrijednosti (boolean) – služe za određivanje ishoda (ispitivanje istinitosti) uvjeta kao if($var){ //...}
  - Lažnom se tretira i nula, prazni niz znakova i niz „0", polje s nula elemenata, objekt bez vrijednosti ili funkcije i NULL
- Polja – može sadržavati puno različitih vrijednosti drugih tipova podataka: $polje=array('PHP', 'Java', 'C++', 'Perl');
  - //polje nizova znakova, vrijednost varijable zadane sa $polje=array(); PHP promatra kao laž
- Objekti – PHP podržava i OOP
- Resursi
- NULL vrijednost: $var = NULL; // može se provjeriti sa if(is_null($val)){echo 'ova varijabla ima vrijednost NULL';}

DEKLARIRANJE VARIJABLI

- Varijable mogu sadržavati vrijednost određenog tipa
- U kodu se varijabla definira pomoću znaka $, npr. $varijabla1=123; ili $Ime='Ivan';
- Deklariranje varijabli obuhvaća definiranje naziva i dodjeljivanje vrijednosti pomoću operatora dodjele. Završavamo sa ;
- PHP razlikuje velika i mala slova u nazivu varijabli

DINAMIČKO DEKLARIRANJE VARIJABLI

- Varijable nije nužno deklarirati na početku skripte već se mogu deklarirati u trenutku kada su potrebne
- Primjer:
  - $k =‘kolicina’;
  - $$k=100;
  - echo ‘kolicina; //ispisuje 100
  -

DODJELJIVANJE VRIJEDNOSTI DRUGE VARIJABLE

- Primjer:
  - $a=10;
  - $b=15;
  - $b=$a;
  - echo $b;

VJEŽBE

- <?php
- $a = 10;
- $b = 20;
- $c = $a + $b;
- echo $c;
- ?>

Try It Out — Calculate the Properties of a Circle

Save this simple script as `circle_properties.php` in your Web server's document root folder, then open its URL (for example, `http://localhost/circle_properties.php`) in your Web browser to run it:

```
<?php
$radius = 4;

$diameter = $radius * 2;
$circumference = M_PI * $diameter;
$area = M_PI * pow( $radius, 2 );

echo "This circle has...<br />";
echo "A radius of " . $radius . "<br />";
echo "A diameter of " . $diameter . "<br />";
echo "A circumference of " . $circumference . "<br />";
echo "An area of " . $area . "<br />";
?>
```

DEFINIRANJE REFERENCE NA VARIJABLU

- $a=10;
- $b=&$a;
- echo $b;
- & - operator dohvaćanja adrese

- $a=10;
- $b=&$a;
- $b=200;
- echo $a;

UKLANJANJE VARIJABLI IZ MEMORIJE

- PHP nudi i mehanizme uklanjanja varijabli iz memorije tijekom izvođenja skripte pomoću funkcije unset
- $a=10;
- $b=15;
- unset($b);

ISPIS VARIJABLE

- $a = 10;
- $b = 15;
- echo $a;
- echo $b;
- Ili
- $a = 10;
- $b = 15;
- echo 'Vrijednost varijable a: ';
- echo $a;
- echo '
';
- echo 'Vrijednost varijable b: ';
- echo $b;
- ili pomoću operatora spajanja nizova znakova (.) možemo ugraditi varijablu unutar HTML koda:
- echo 'Vrijednost varijable a: '.$a;

KONSTANTE

- Konstante se definiraju samo jednom i nakon toga im se više ne može mijenjati vrijednost
- define('PDV', 23);
- Koristi se funkcija define. Prvi parametar je naziv konstante, a drugi vrijednost.
- define('NAZIV_USTANOVE', 'Privredna banka Zagreb');
- echo PDV;
- echo NAZIV_USTANOVE;
- Za konstante se koristi samo naziv bez znaka $, zato ih je dobro pisati velikim tiskanim slovima i tako lakše uočavati u kodu.

IZRAZI I OPERATORI

- Izrazi su dijelovi koda koji predstavljaju određenu cjelinu koja se može vrednovati i daje neki rezultat (vrijednost).
- Operatori uzimaju neke vrijednosti i s njima izvode određene operacije da bi dobili neki rezultat (međurezultat).
- Operatori imaju prioritete. Upotrebom zagrada možemo naznačiti da se neka operacija obavi prva.

OPERATOR DODJELJIVANJA VRIJEDNOSTI

- Za dodjeljivanje vrijednosti varijabli (operandu) koja se nalazi lijevo od operatora. Desni operand može biti broj, niz znakova, izraz čiju vrijednost želimo pohraniti u varijablu ili ispisati...
- $a=10;
- $b='PHP'
- $c=1+2*3;

ARITMETIČKI OPERATORI

- Zbrajanje (+)
- Oduzimanje (-)
- Množenje (*)
- Dijeljenje (/)
- Modulo (%) – ostatak pri dijeljenju

OPERATOR ZA NASTAVLJANJE NIZOVA (.)

- Za povezivanje dvije tekstualne vrijednosti ili za umetanje varijable unutar nekog niza znakova.
- Primjer 1: echo 'Vase ime je '.$ime.',a prezime '.$prezime;
- Primjer 2: $c=$a.$b; echo $c;

OPERATOR AUTOMATSKOG POVEĆAVANJA I SMANJIVANJA

- Automatsko povećavanje: $a++; ili ++$a;
- Automatsko smanjivanje: $a--; ili --$a;

OPERATORI USPOREDBE

- Za provjeru odnosa vrijednosti dva operanda
- Rezultat je neka logička vrijednost (izraz): true ili false
- Najčešće se koriste u kombinaciji s naredbom if()-else

- Jednakost (==), istina kad su vrijednosti dva operanda jednake
- Identično (===), istina kada oba operanda imaju istu vrijednost i istog su tipa
- Nejednakost (!=), istina kad su vrijednosti oba operanda različite
- Veće od (>), vraća istinu ako je vrijednost lijevog operanda veća od desnog
- Veće ili jednako (>=), istina ako je vrijednost lijevog operanda veća ili jednaka vrijednosti desnog
- Manji od (<), istina ako je vrijednost lijevog operanda manja od vrijednosti desnog
- Manje ili jednako (<=), istina kada je vrijednost lijevog operanda manja ili jednaka vrijednosti desnog operanda

LOGIČKI OPERATORI

- Za povezivanje dva ili više uvjeta unutar jednog izraza

- Logičko i (&&, and), istina samo onda kada su oba operanda istinita
- Logičko ili (||, or), istina ako je bilo koji od dva operanda istinit
- Logička negacija (!) mijenja logički rezultat, pretvara istinit operand u lažan i obrnuto

OPERATORI PRETVORBE

- U PHP-u varijable se deklariraju prema potrebi, a PHP im odredi tip podataka prema vrijednosti koja im se dodjeljuje
- Operatori dodjeljivanja omogućuju definiranje točnog tipa podataka neke vrijednosti zbog:
  - Kontrole nad dodjeljivanjem tipa podataka
  - Prevođenja vrijednosti neke varijable iz jednog tipa podataka u drugi (npr. iz niza znakova u broj ili obrnuto)
- (int), (float), (string), (bool), (array), (object)

OPERATORI DODJELJIVANJA S OPERACIJOM

- Služe za ubrzavanje pisanja nekih aritmetičkih operacija

- +=, za dodavanje vrijednosti desnog operanda lijevom
- -=, oduzima se vrijednost desnog operanda od lijevog
- /=, lijevi operand prima vrijednost kvocijenta lijevog i desnog
- *=, dodjeljuje lijevom operandu umnožak lijevog i desnog operanda

KONTROLNE STRUKTURE I PETLJE

- Programske konstrukcije koje omogućuju kontrolu tijeka i ponavljanje izvršavanja naredbi (while, for, if-else, switch-case...)

NAREDBA IF-ELSEIF-ELSE

- Provjerava da li je izraz u zagradi istinit, pa izvršava odgovarajući dio koda
- Pomoću ključne riječi else definira se alternativni blok naredbi koje će se izvršiti ako izraz u zagradama nije istinit
- Dozvoljeno je i ugnježđivanje if-else naredbi
- Može se dodati i više alternativnih programskih blokova pomoću ključne riječi elseif

```
$x=5;
if ($x==5)
    {echo 'Vrijednost varijable je 5';}
else
    {echo 'Vrijednost varijable različita je od 5';}
```

ALTERNATIVNO OZNAČAVANJE BLOKOVA KOD IF-ELSE NAREDBE

```
$x=5;
if ($x==5):
    echo ‘Vrijednost varijable je 5’;
else:
    echo ‘Vrijednost varijable različita je od 5’;
endif:
```

NAREDBA SWITCH-CASE

- Ispituje se opća vrijednost izraza (cjelobrojna ili niz znakova) i prema njoj izvršava određeni blok programskog koda
- blok nakon riječi default izvodi se ako ne postoji case sa vrijednošću varijable

```
$a=1;
switch ($a)
{
case 1:
   echo 'Pozz';
break;
case 2:
    echo 'PozzPozz';
break;
default:
    echo 'Bye';
}
```

```
$a=1;
switch ($a):
case 1:
    echo 'Pozz';
break;
case 2:
     echo 'PozzPozz';
break;
default:
     echo 'Bye';
endswitch;
```

KLJUČNA RIJEČ BREAK I PROPADANJE (FALL-THROUGH)

- Oznaka da završava blok koda (case dio) koji je trebalo izvršiti
- Izlazi se iz switch-case naredbe
- Ako ne zatvorimo case dio pomoću ključne riječi break dolazi do propadanja, odnosno omogućujemo da se dio koda izvršava za dvije ili više vrijednosti varijable
- Ako propadanje radimo namjerno, uputno je to komentirati

PETLJA WHILE

- Za ponavljanje nekog dijela programskog koda (npr. kod stvaranja HTML tablice s podacima u petlji iscrtavamo redove tablice dinamički)
- u svakom se koraku provjerava izraz u zagradama i dokle god je istinit izvršava se (ponavlja) odgovarajući dio programa

```
$i=1;
while($i<=100)
{
   $zbroj+=$i;
   $i++;
}
echo $zbroj;
```

```
$i=1;
while($i<=100):
   $zbroj+=$i;
   $i++;
endwhile;
echo $zbroj;
```

```
$i=1;
while($i<=100)
{
   $zbroj+=$i;
   $i++;
   if ($zbroj>=50) break;
}
echo $zbroj;
```

- Uz upotrebu ključne riječi break možemo prekinuti izvršavanje petlje i ranije

BITNO KOD PETLJE WHILE

- Postoji varijabla koja ima **početnu** vrijednost i čija se vrijednost **provjerava** u uvjetu, odnosno izrazu za provjeru vrijednosti te varijable (kad njegova vrijednost prestane biti istinita, petlja se prekida)
- Važno je imati i korak petlje, odnosno **mijenjati** vrijednost varijable koja se ispituje (čime ćemo spriječiti pojavu beskonačne petlje ili rušenje aplikacije)

PETLJA DO-WHILE

- Uvjet provjerava na kraju i ako je istinit vraća se na početak petlje; ključna riječ break može ranije prekinuti izvođenje
- Omogućava da se programski kod unutar petlje izvrši barem jednom

```
$a=1;
do
{
    echo 'a='.$a;
    echo '<br />';
    $a++;
} while ($a<=5);
```

FOR PETLJA

- Ima više izraza odvojenih sa ; prvi deklarira brojač i dodjeljuje mu početnu vrijednost, drugi predstavlja uvjet koji određuje do kada će se petlja izvoditi, a treći je korak petlje

```
for($i=1;$i<=5;$i++)
{
    echo 'i='.$i.'<br />';
}
```

PETLJA FOREACH()

- Za specifičnu namjenu ispisa elemenata polja

```
$polje=array('Ivo', 'Ana', 'Petar');
foreach($polje as $ime)
{
    echo $ime.'<br />';
}
```

POLJA

- Polje je poseban tip podataka, više podataka posloženih u skup parova tipa ključ-vrijednost (ključ služi za pristup podacima)
- Primjer:

  $a=array(123, 15, 140); ....ovdje su ključevi: 0, 1 i 2

DEFINIRANJE I VRSTE POLJA

- Polja razlikujemo prema tipu ključa: **indeksna** (imaju brojčane ključeve) i **asocijativna** (ključevi su tekstualni)

INDEKSNA POLJA

```
$ip=array('Ana','Ivan','Petar');
```

u zagradi su vrijednosti članova, brojčani ključevi ovdje se dodjeljuju automatski

- Moguće je ključevima definirati vrijednosti:

```
$ip=array();
$ip[10]='Ana';
$ip[11]='Ivan';
$ip[12]='Petar';
```

Ili:

```
$ip=array(10=>'Ana', 11=>'Ivan', 12=>'Petar');
```

- Novu vrijednost (na ključ za jedan veći od trenutno najvećeg) možemo dodati sa `$ip[]='Iva';`

FUNKCIJA ARRAY_PUSH

- Gotova funkcija za dodavanje vrijednosti u polje:

- `$ip=array('Ana', 'Ivan', 'Petar');`
- `array_push ($ip, 'Marko', 'Petra');`

ASOCIJATIVNA POLJA

```
$ap=array();
$ap['ime1']='Ana';
$ap['ime2']='Ivan';
$ap['ime3']='Petar';
```

Ili:

```
$ap = array('ime1'=>'Ana', 'ime2'=>'Ivan', 'ime3'=>'Petar');
```

KOMBINIRANJE INDEKSNIH I ASOCIJATIVNIH KLJUČEVA

- PHP „ne zna” automatski uvećavati asocijativne ključeve, nego novododanom elementu dodjeljuje prvi slobodni indeksni ključ tako da neki elementi istog polja mogu imati asocijativne, a neki brojčane ključeve

DOHVAĆANJE I ISPIS ELEMENATA POLJA

- Elemente polja možemo dohvatiti preko vrijednosti ključa i pomoću petlji

echo $pi[1];

echo $pa['ime2'];

for($i=0;$i<=2;$i++) echo $pi[$i]; ispisuje sve elemente polja kada znamo koliko polje ima elemenata

- PHP funkcija count za predano polje vraća koliko u njemu ima elemenata i tako omogućuje ispravno funkcioniranje ispisa u programu i kod dodavanja novih vrijednosti u polje
- for($i=0;$i<count($pi);$i++) echo $pi[$i];

ISPIS ASOCIJATIVNIH POLJA

- Polje koje ima asocijativne ključeve ispisujemo pomoću foreach petlje

```
foreach($pa as $key=>$val) echo $val;
```

- Ili

```
foreach($pa as $val) echo $val;
```

ZADACI

- 1. Unijeti imena 5 učenika u polje pa ispisati sve učenike pomoću petlje na slijedeći način: Ime 1. učenika je Ante.. Zadatak riješiti bez upotrebe nove varijable-brojača.
- 2. Unijeti u polje podatke za udaljenosti Čakovca od najbliža susjedna četiri grada. Ispisati ime najbližeg grada, te koliko vremena je potrebno da bi se tamo stiglo ako je prosječna brzina 60 km/h.
- 3. Unijeti u asocijativno polje imena i visine za 5 učenika pa ispisati imena najnižeg i najvišeg učenika.

VIŠEDIMENZIONALNA POLJA

- U jednodimenzionalnim poljima vrijednosti elemenata su brojčane ili tekstualne.
- Kod višedimenzionalnih polja vrijednost elementa polja je novo polje.

```
$a=array();
$a[]=array(10, 15, 20);
$a[]=array(1, 5);
$a[]=array(100, 200);
```

- Ispis echo $a[2][1]

0			1		2	
0	1	2	0	1	0	1
10	15	20	1	5	100	200

PRIMJER DEFINIRANJA

```
$p=array();
$p[1][2][]=100;
$p[1][2][]=200;
$p[1][2][]=300;
```

- Za ispis polja upotrebljavaju se ugniježđene foreach petlje:

```
foreach($p as $element)
{
     foreach($element as $val)
     {
          echo $val.’<br />’;
     }
}
```

INDEKSNA I ASOCIJATIVNA VIŠEDIMENZIONALNA POLJA

```
$p=array();
$p[1]['ime']='Ana';
$p[1]['prezime']='Antic';
$p[2]['ime']='Ivo';
$p[2]['prezime']='Ivancic';
$p[3]['ime']='Petar';
$p[3]['prezime']='Petrinovic';
```

- Glavno polje je indeksno i ima ključeve od 1 do 3. Svako polje unutar glavnih elemenata ima asocijativne ključeve koji se kod svakog novog polja ponavljaju.

NAČINI ISPISA POLJA

- Elemente polja možemo ispisati pomoću for i foreach petlje

```
for($i=1;$i<=3;$i++)
{
    echo 'Ucenik br. '.$i.'<br />';
    echo 'Ime: '.$p[$i]['ime'].'<br />';
    echo 'Prezime: '.$p[$i]['prezime'].'<br />';
}
```

```
foreach($p as $key=>$ucenik)
{
    echo 'Ucenik br. '.$key.'<br />';
    echo 'Ime: '.$ucenik['ime'].'<br />';
    echo 'Prezime: '.$ucenik['prezime'].'<br />';
}
```

PRIMJER DEKLARACIJE POLJA

Title	Price	Number
rose	1.25	15
daisy	0.75	25
orchid	1.15	7

```
<?php
$shop = array( array("rose", 1.25 , 15),
               array("daisy", 0.75 , 25),
               array("orchid", 1.15 , 7)
             );
?>
```

```
<?php
$shop = array( array( Title => "rose",
                      Price => 1.25,
                      Number => 15
                    ),
               array( Title => "daisy",
                      Price => 0.75,
                      Number => 25,
                    ),
               array( Title => "orchid",
                      Price => 1.15,
                      Number => 7
                    )
             );
?>
```

PRIMJERI ISPISA

```
<?php
echo "<h1>Manual access to each element</h1>";

echo $shop[0][0]." costs ".$shop[0][1]." and you get ".$shop[0][2]."<br />";
echo $shop[1][0]." costs ".$shop[1][1]." and you get ".$shop[1][2]."<br />";
echo $shop[2][0]." costs ".$shop[2][1]." and you get ".$shop[2][2]."<br />";

echo "<h1>Using loops to display array elements</h1>";

echo "<ol>";
for ($row = 0; $row < 3; $row++)
{
    echo "<li><b>The row number $row</b>";
    echo "<ul>";

    for ($col = 0; $col < 3; $col++)
    {
        echo "<li>".$shop[$row][$col]."</li>";
    }

    echo "</ul>";
    echo "</li>";
}
echo "</ol>";
?>
```

```
<?php
echo "<h1>Manual access to each element from associative array</h1>";

for ($row = 0; $row < 3; $row++)
{
    echo $shop[$row]["Title"]." costs ".$shop[$row]["Price"]." and you get
".$shop[$row]["Number"];
    echo "<br />";
}

echo "<h1>Using foreach loop to display elements</h1>";

echo "<ol>";
for ($row = 0; $row < 3; $row++)
{
    echo "<li><b>The row number $row</b>";
    echo "<ul>";

    foreach($shop[$row] as $key => $value)
    {
        echo "<li>".$value."</li>";
    }

    echo "</ul>";
    echo "</li>";
}
echo "</ol>";
?>
```

FUNKCIJE

- Funkcija je izdvojeni blok programskog koda koji za zadatak ima izvesti neki posebni zadatak (ispisivati neke vrijednosti, računati i vratiti vrijednosti i dr.) i upotrebljava se kad pri pisanju skripte neki blok koda koristimo više puta ili se pojavljuje na više mjesta (lakše je i pouzdanije izvoditi izmjene).
- Funkcije se obično definiraju na početku skripte i pozivaju prema potrebi.
- PHP ima dvije vrste funkcija: ugrađene (count(), rand(), array_push(),...) i one koje definiramo sami.
- Da bi koristili ugrađene funkcije potrebno je poznavati njihov poziv i koji će rezultat vratiti. Nije potrebno poznavati način rada. Načinom rada upravljamo kod funkcija koje definiramo sami.
- Funkcije pozivamo tako da im napišemo ime i u zagradama proslijedimo potrebne parametre.

PRIMJERI UPOTREBE UGRAĐENIH FUNKCIJA

- Funkcija time() – vraća broj sekundi koji je prošao od ponoći 1. siječnja 1970. (UNIX timestamp): echo time();
- Funkcija strlen() – prebrojava znakove u nizu znakova: echo strlen($rijec);
- Funkcija is_array() – vraća logičku vrijednost, provjerava je li predani parametar polje: if (is_array($p))...
- Funkcija date() – ovisno o parametru može ispisati neki podatak vezan uz trenutni datum i vrijeme, tako na primjer echo date('f'); prikazuje naziv trenutnog mjeseca, a echo date('d.m.Y'); ispisuje formatirani trenutni datum.
- $zaTjedanDana=time()+7*24*60*60; echo date('d.m.Y', $zaTjedanDana); određen je format i timestamp koji se ispisuje
- Funkcija rand() – vraća slučajno odabrani cijeli broj, za nasumični prikaz nekog broja: echo rand(); ako je potrebno odrediti interval iz kojeg će se generirati nasumični broj piše se echo rand(1, 10);

DEFINIRANJE FUNKCIJE

- Funkcije definiramo pomoću ključne riječi function nakon koje slijedi ime funkcije (prema pravilima za stvaranje imena). Naziv funkcije nije osjetljiv na velika i mala slova.
- Nakon imena funkcije pišu se zagrade unutar kojih se definiraju varijable koje će primiti vrijednosti proslijeđenih parametara. Programski kod koji funkcija treba izvršiti piše se unutar vitičastih zagrada. Ako funkcija mora vratiti neku vrijednost, to se označava pomoću ključne riječi return.
- Primjer: napisati funkciju koja će vraćati sumu dva broja koja dobiva kao parametre.
- Primjer: napisati funkciju koja ispisuje u obliku tablice podatke o učenicima koje dobiva kao parametar tipa polje.

DOSEG VARIJABLE

- Funkcije iz primjera imaju svoj doseg i ne mogu pristupati varijablama iz glavnog programa, a glavni program ne može primjenjivati varijable deklarirane u funkciji.
- Primjer:

```
$a=2;
function inc_a(){ $a++;echo $a;}
inc_a(); echo $a;
```

GLOBALNE I STATIČKE VARIJABLE U FUNKCIJAMA

- Vrijednosti koje funkcija treba iz glavnog programa mogu joj se proslijediti pomoću parametara
- Varijablama iz glavnog programa funkcija može pristupiti tako da se u funkciji deklarira kao global $a;
- Statičke varijable u funkciji dopuštaju da se vrijednost varijabli sačuva za slijedeći poziv funkcije u skripti. Tada se mora deklarirati varijablu unutar funkcije pomoću ključne riječi static $a=0;

PARAMETRI FUNKCIJA – PREDAJA PARAMETARA PO VRIJEDNOSTI I PO REFERENCI

- **Predaja parametara po vrijednosti**: kod poziva funkcije u zagradi se navedu vrijednosti ili varijable koje se predaju funkciji. Funkcija kopira te vrijednosti u svoje nove varijable i dalje radi s njima.

- **Predaja parametara po referenci**: funkcija pomoću operatora & uzima referencu do memorijske adrese u kojoj je ta vrijednost spremljena. Varijabla koja je parametar funkcije sada sadržava memorijsku adresu stvarnog parametra, odnosno sadrži referencu na samu varijablu, tako da se na mjesto stvarnog parametara upisuje nova vrijednost.

```
function potencija (&$val){$val=$val*$val;} $a=2; potencija($a); echo $a;
```

ZADANI PARAMETRI

```
function suma($a=0, $b=0){ $c=$a+$b;return $c;} echo suma(14);
function showNames($imena=array()){ ... } showNames();
```

- Ako se neki od očekivanih parametara ne proslijedi funkciji, varijablama ostaje zadana vrijednost

FUNKCIJE S VARIJABILNIM BROJEM PARAMETARA

- Funkciju definiramo bez parametara u zagradama i koristimo neku od tri ugrađene funkcije koje pomažu pri manipulaciji parametrima proslijeđenim funkciji
- func_get_args(): vraća sve parametre poslane funkciji i stavlja ih u jedno polje
- func_num_args(): vraća broj proslijeđenih parametara
- func_get_arg(): vraća točno određeni parametar koji je proslijeđen funkciji
- Primjer:

```
function sum()
{for($i=0;$i<func_num_args();$i++)
$sum+=func_get_arg($i); return $sum;}
echo sum(1, 10, 100);
```

Ili:

```
{$parametars=func_get_args();foreach($parametars as $param)$sum+=$param;return $sum;}
```

VARIJABILNE FUNKCIJE

```
$func_name='sum';
if (function_exists($func_name)) $func_name();
```

- PHP prvo dohvati vrijednost varijable te pokušava pozvati funkciju koja se zove isto kao vrijednost varijable
- Dobro je kod toga iskoristiti ugrađenu PHP funkciju `function_exists()` koja provjerava postoji li tražena funkcija.

OBJEKTNO ORIJENTIRANO PROGRAMIRANJE

- Pregledniji dizajn aplikacije
- Lakše održavanje
- Timska interakcija
- ...

KLASE I OBJEKTI

- Klasa: oblik nacrta, opisuje stvari (objekte) na apstraktan način, sadrži atribute (svojstva, properties) i ponašanja (funkcije, methods)
- Objekt: pripada klasi i kreira se pomoću nje (stvara, instancira). Objekt je instanca klase.

OSNOVNI POJMOVI

Pojam	Objašnjenje
Apstrakcija	Ideja pomoću koje možemo sagledati stvari i svesti ih na apstraktna svojstva i metode (ponašanja) uočavanjem i opisivanjem zajedničkih svojstava i metoda sličnih stvari.
Učahurivanje (enkapsulacija)	Stvaranje paketa svojstava i metoda u jednoj klasi i upotreba tih stvojstava i metoda bez da se poznaje kako su implementirane, odnosno kako rade.
Nasljeđivanje (inheritance)	Mogućnost da klase naslijede ponašanja (metode) i svojstva drugih klasa uz mogućnost proširenja novima. Bitno jer donosi velike prednosti kod nadogradnje programskog koda i suradnje većeg broja programera na istoj aplikaciji.
Polimorfizam	Ideja da se nešto tretira kao nešto drugo, npr. dvije klase mogu imati iste nazive za svoja ponašanja, a zapravo stvari rade na različit način.

DEFINIRANJE KLASE INSTANCIRANJE OBJEKTA

- Primjer klase koja crta jednostavnu HTML tablicu
- $this sadrži referencu na objekt na kojem se metoda poziva
- Stvaranje novog objekta iz klase: $tablica1=new tablica();
- Poziv metode za stvoreni objekt: $tablica1->prikaz();
- Mijenjanje svojstava tablice: $tablica1->broj_redaka=10;

```
class tablica
{
var $broj_redaka=4;
var $broj_stupaca=4;

function prikaz()
{
	echo '<table border="1">';
	for($i=0;$i<this->broj_redaka;$i++)
	{
		echo '<tr>';
		for($j=0;$j<this->broj_stupaca;$j++)
		{
			echo '<td>';
			echo '</td>';
		}
		echo '</tr>';
	}
	echo '</table>';
}
}
```

KONSTRUKTOR I DESTRUKTOR

- Posebne metode koje je moguće definirati unutar klase, a koje nije potrebno pozivati. Konstruktor se poziva pri instanciranju objekta, dok se destruktor pokreće na kraju skripte.
- Konstruktor se definira pomoću ključne riječi __construct i može omogućiti npr. izmjenu svojstava objekta odmah nakon instanciranja.
- Pogodan je za inicijalizacije preko vrijednosti koje su mu proslijeđene kao parametri: $tablica1=new tablica(10, 5);
- Metoda destruktora definira se pomoću ključne riječi __destruct i poziva kod uništenja objekta (kad više nema referenci ili kad je kraj skripte): function __destruct(){...};
- https://www.codecademy.com/courses/web-beginner-en-bH5s3/0/1
- http://www.tutorialspoint.com/php/php_object_oriented.htm

```
function __construct($r, $s)
{
        $this->broj_redaka=$r;
        $this->broj_stupaca=$s;
}
```

NASLJEĐIVANJE

- Jedna klasa može preuzeti sva svojstva i metode druge klase te ih dodatno proširiti svojim svojstvima i metodama
- Nasljeđivanje se obavlja putem ključne riječi extends
- Klasa novaTablica nasljeđuje sva svojstva i metode klase tablica, ali ima i novu metodu prikaz() (ima mogućnost ispisa zaglavlja tablice)
- Obje klase dijele i programski kod konstruktora
- Provjera da li je određeni objekt instanca neke klase obavlja se pomoću operatora instanceof: if($tbl instanceof tablica)...

```
class novaTablica extends tablica
{
function prikaz()
{
        echo '<table border="1">';
        echo '<tr>';

        for($j=0;$j<this->broj_stupaca;$j++)
                {
                        echo '<td>';
                        echo '<b>Stupac '.$j.'</b>';
                        echo '</td>';
                }
        echo '</tr>';

        for($i=0;$i<this->broj_redaka;$i++)
        {
                echo '<tr>';
                for($j=0;$j<this->broj_stupaca;$j++)
                {
                        echo '<td>';
                        //podaci za prikaz
                        echo '</td>';
                }
                echo '</tr>';
        }
        echo '</table>';
}
}
```

SUČELJE

- Sučelje (interface) je posebna vrsta klase koja sadrži samo popis metoda koje neka klasa mora implementirati, ali ne definira kako te metode moraju izgledati i što trebaju raditi.
- Definiranje sučelja radi se pomoću ključne riječi interface ispred imena, a sve metode unutar te klase moraju ispred svoje definicije imati ključnu riječ public.
- Ovo sučelje definira pravilo da svaka klasa koja ga implementira mora imati metodu koja se zove prikaz(). Nakon toga se kod pisanja vlastite klase pomoću ključne riječi implements naglašava da ta klasa implementira upravo to sučelje.

```
interface iTable
{
        public function prikaz();
}

class tablica implements iTable
{
        var $broj_redaka=4;
        var $broj_stupaca=4;

        function __construct($r, $s)
        {
        $this->broj_redaka=$r;
        $this->broj_stupaca=$s;
        }

        function prikaz()
        {
        //
        }
}
```

APSTRAKTNA KLASA

- I u apstraktnoj klasi definira se kako će izgledati neka klasa koja implementira ili nasljeđuje metode.
- Sučelje sadrži samo smjernice, a apstraktna klasa osim smjernica koje sve metode mora sadržavati klasa koja ju naslijedi, može sadržavati i neke zajedničke metode.
- set_red_stup() je zajednička metoda pomoću koje se postavljaju vrijednosti svojstava.
- Apstraktnu klasu ne možemo instancirati u objekt.
- Svaka klasa koja ima barem jednu apstraktnu metodu mora isto biti deklarirana kao apstraktna.

```
abstract class aTable
{
        abstract function prikaz();

        function set_red_stup($r, $s)
        {
        $this->broj_redaka=$r;
        $this->broj_stupaca=$s;
        }
}

class tablica extends aTable
{
        var $broj_redaka=4;
        var $broj_stupaca=4;

        function __construct($r, $s)
        {
        $this->broj_redaka=$r;
        $this->broj_stupaca=$s;
        }

        function prikaz()
        {...}
}
```

VIDLJIVOST SVOJSTAVA I METODA

- Odnosi se na njihovu dostupnost.
- Kontrolira se tako da se ispred deklaracije postavi jedna od tri moguće ključne riječi (public, protected i private)

Ključna riječ	Opis
public	Svojstvima i metodama deklariranim kao public može se pristupiti s bilo kojeg mjesta u kodu.
protected	Svojstvima i metodama deklariranim kao protected može se pristupiti samo iz pripadajuće klase ili iz klase koja ju je nasljedila, ili iz klase koju je ta klasa nasljedila.
private	Svojstvima i metodama deklariranim kao private može se pristupiti samo iz klase u kojoj su deklarirani.

PRIMJER

- Vidljivost se može precizirati i za svojstva i za metode.

```
class Test
{
        public $a='public';
        protected $b='protected';
        private $c='private';

        function print_var()
        {
        echo $this->a;
        echo $this->b;
        echo $this->c;
        }
}

$obj=new Test();
echo $obj->a;
echo $obj->b;//greska
echo $obj->c;//greska
$obj->print_var();
```