**Obrada stringova**

Jedinica: 13 od 19

**Rezime**

* *print( ) i echo()*su najjednostavnije funkcije za ispis teksta;
* *printf()*koja prikazuje statički i dinamički tekst;
* Funkcija *sprintf*() tekst smešta u novu promenljivu;
* *trim( )* – uklanja prazna mesta na početku i na kraju stringa;
* *ltrim*( ) – uklanja prazna mesta na početku stringa;
* *rtrim*( ) – uklanja prazna mesta na kraju stringa;
* *str\_word\_count*( ) – razbija tekst na reči i smešta ih u niz ili prebrojava reči u stringu;
* *strpbrk(string, char)* – traži karakter char u tekstu string;
* *strtoupper(string)* – prebacuje sva slova u velika;
* *strtolower(string)* – prebacuje sva slova u mala.

Vešta manipulacija stringovima predstavlja veoma važnu osobinu ozbiljnog programera, te ćemo se zato detaljnije upoznati sa ovom temom. Najjednostavnije funkcije za ispis teksta su **print**i **echo.**Njihova primena je gotovo identična, s tim što je echo za nijansu brža pri izvršavanju. Što se tiče sintakse, svejedno je da li ćete pisati

|  |  |
| --- | --- |
| 1 | print "Hello"; |

ili

|  |  |
| --- | --- |
| 1 | echo "Hello"; |

Kada je potrebno nešto komplikovanije ispisivanje teksta, koristi se funkcija **printf()**,koja prikazuje statički tekst i dinamičke informacije, a i njena primena izgleda ovako:

|  |  |
| --- | --- |
| 1 | printf("Danas je prodato %d artikala. ", 100 ); |

Nakon izvršavanja funkcije, biće ispisan sledeći tekst:

Danas je prodato 100 artikala.

**Jednostruki i dvostruki navodnici**

Kao što je već napomenuto, da biste dodelili [string](https://www.link-elearning.com/linkdl/opisPojma.php?id=139126) vrednost nekoj promenljivoj, neophodno je da je uokvirite znacima navoda. Unutar teksta koji dodeljujete nekoj promenljivoj moguće je da koristite i specijalne karaktere:

Primer dodavanja HTML tagova unutar teksta:

|  |  |
| --- | --- |
| 1  2 | $string = "PHP <br> programming.";  echo $string; |

će ispisati sledeće dve linije teksta:

*PHP  
      programming.*

Ukoliko bismo želeli da na strani bude ispisan string onako kako je napisan i u PHP kodu, dakle, bez interpretiranja HTML tagova, iskoristili bismo funkciju htmlspecialchars().

Primer:

|  |  |
| --- | --- |
| 1 | echo htmlspecialchars("PHP <br> programming."); |

Rezultat na strani će izgledati ovako:

PHP <br> programming.

Izbor jednostrukih ili dvostrukih navodnika pravi razliku prilikom ispisivanja promenljivih. Promenljiva napisana unutar dvostrukih navodnika biće pročitana, odnosno biće prikazana njena vrednost, dok će promenljiva napisana pod jednostrukim navodnicima biti tretirana kao bilo koji drugi tekst.

Pogledajmo ovo na primerima.

Primer za jednostruke navodnike:

|  |  |
| --- | --- |
| 1  2 | $x = 10;  echo 'Test $x'; |

Rezultat na strani: Test $x

Primer za dvostruke navodnike:

|  |  |
| --- | --- |
| 1  2 | $x = 10;  echo "Test $x"; |

Rezultat na strani: Test 10

Ukoliko ipak želite da koristite dvostruke navodnike, ali da unutar njih prikažete naziv promenljive, ili drugi specijalni karakter koji ne treba interpretirati, možete se poslužiti *escape* karakterom (\).

Primer:

|  |  |
| --- | --- |
| 1  2 | $x = 10;  echo "My variable is \$x."; |

Rezultat na strani: My variable is $x.

Upotreba escape karaktera je posebno korisna kada imamo potrebu da u stringu koji se prikazuje na strani postavimo navodnike.

Primer:

|  |  |
| --- | --- |
| 1  2 | $x = 10;  echo "Hello, \"World\"."; |

Rezultat na strani: Hello, "World".

**Operacije nad stringovima**

Jedna od najčešće korišćenih operacija nad stringovima jeste nadovezivanje (tačka .). Pored ovog operatora, PHP sadrži i veliki broj ugrađenih funkcija za rad sa tekstom. Neke od njih su:

* + *trim( )* – uklanja prazna mesta na početku i na kraju stringa,
  + *ltrim*( ) – uklanja prazna mesta na početku stringa,
  + *rtrim*( ) – uklanja prazna mesta na kraju stringa,
  + *str\_word\_count*( ) – vrši prebrojavanje traženih reči u stringu, ali uz dodatno podešavanje opcionih parametara može kreirati i nove nizove od rezultata pretrage,
  + *strpbrk(string, char)* – traži karakter char u tekstu string i vraća ostatak stringa nakon pojavljivanja karaktera,
  + *strtoupper(string)* – prebacuje sva slova u velika,
  + *strtolower(string)* – prebacuje sva slova u mala.

**Primer:**

Funkcija str\_word\_count( ) se može koristiti na više načina. Takođe, koristi se zajedno sa funkcijom za ispis print\_r(), koja vrši ispis nizova, ali tako da podaci budu čitljivi korisniku. Pogledajmo sledeće primere upotrebe:

|  |  |
| --- | --- |
| 1 | echo str\_word\_count("Hello World!"); |

Nakon izvršenja, funkcija vraća rezultat:

2

Što predstavlja broj reči u stringu.

Primer u kome se kao drugi argument uključuje i broj:

|  |  |
| --- | --- |
| 1 | print\_r(str\_word\_count("Hello World!", 1)); |

Nakon izvršenja ispisuje:

Array ( [0] => Hello [1] => World )

Možemo da kao treći argument, ukoliko string sadrži i neki specijalni karakter, uključimo i njega u okviru parametara funkcije, kao što je prikazano u primeru:

|  |  |
| --- | --- |
| 1  2 | print\_r(str\_word\_count("Hello World!", 1));  print\_r(str\_word\_count("Hello World!", 1, "!")); |

Nakon izvršenja, uočavamo da ukoliko kao treći argument nije uključen specijalni karakter u okviru stringa, niz na izlazu će ga zanemariti i ispisati samo niz ostalih reči. Dok, ukoliko je naveden kao argument, niz na izlazu će uzeti u obzir i spacijalni karakter.

Prikaz podataka nakon izvršavanja programa:

Array ( [0] => Hello [1] => World )

Array ( [0] => Hello [1] => World! )

**Primer:**

Sada ćemo prikazati upotrebu funkcija za rad sa stringovima u slučaju pretvaranja u velika ili u mala slova. Pogledajmo primer koda:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | $str = "PHP is my favorite progamming language.";  echo "Without functions : " . $str;  echo "<br />";  echo "Function - strtoupper : " . strtoupper( $str ) ;  echo "<br />";  echo "Function - strtolower: " . strtolower( $str ); |

Rezultat na strani:

Without functions : PHP is my favorite progamming language.  
Function - strtoupper : PHP IS MY FAVORITE PROGAMMING LANGUAGE.  
Function - strtolower: php is my favorite progamming language.

**Formatiranje prikaza teksta**

Funkcije *printf*() i *sprintf*() omogućavaju [formatiranje teksta](https://www.link-elearning.com/linkdl/opisPojma.php?id=139106) i brojeva, i njihovo kombinovanje. Osnovna sintaksa ovih naredbi je

      printf( "format", $val1, $val2,...);  
     $newStr = sprintf( "format", $val1, $val2,...);

I jedna i druga funkcija formatiraju tekst na osnovu argumenta *format*, s tom razlikom što ga funkcija *printf*() samo prikazuje, a funkcija *sprintf*() ga smešta u novu promenljivu. Prvi argument ovih funkcija, „*format*", predstavlja instrukcije za formatiranje. Svaka instrukcija za formatiranje ima sledeći oblik:

          %*pad*- *duzina.dectip*

pri čemu su:

* + *pad* – karakter koji se koristi za popunjavanje slobodnih mesta, ukoliko je string kraći od navedene dužine,
  + *znak '*-' – označava da karakteri budu levo poravnati. Ukoliko se izostavi, podrazumeva se desno poravnanje,
  + *duzina*– broj karaktera koji se koriste za prikazivanje vrednosti
  + .dec – broj decimalnih mesta,
  + tip – tip vrednosti koja se prikazuje (s za *string*, f za *float*).

Pogledajte sledeće primere:

**Primer 1.**

|  |  |
| --- | --- |
| 1  2  3 | $numOfBoys = 3;      $numOfGirls = 2;      printf("%s boys and  %s girls", $numOfBoys, $numOfGirls); |

ispisaće poruku:

      3 boys and 2 girls

**Primer 2.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | $price = 30000;      $product = "Samsung TV";      $message1 = sprintf("%s costs %09.2f dollars. ", $product, $price);      $message2 = sprintf("%'.-20s%9.2f", $product, $price);      echo $message1;      echo $message2; |

ispisaće poruku:

      Samsung TV costs 030000.00 dollars.  
     SamsungTV........... 30000.00

U prvoj poruci prva instrukcija za formatiranje je *%s* i ona se odnosi na prvu promenljivu, *$product*. Ova instrukcija će samo da ispiše vrednost promenljive. Druga instrukcija za formatiranje je *%09.2f* i odnosi se na drugu promenljivu, *$price*. Ona govori da broj treba da bude predstavljen sa 9 karaktera i dva decimalna mesta, pri čemu je 0 karakter sa kojim se popunjavaju prazna mesta.

U drugoj poruci prva instrukcija za formatiranje naziva proizvoda je *%'.-20s*. Ona govori da promenljivu *$product* treba prikazati poravnato ulevo, sa dvadeset karaktera, pri čemu se prazna mesta popunjavaju tačkom. Cena se prikazuje na sličan način kao i u prvoj poruci, s tom razlikom što nije naveden karakter koji popunjava prazna mesta, pa se koristi podrazumevani *space*karakter.

Kada želite da stringove preklapate jedne drugima, koristi se funkcija *substr\_replace(),*koja ima osnovni oblik:

substr\_replace(Promenljiva, „ karakteri kojima se menja string ”, indeks početnog karaktera, broj karaktera koji će biti zamenjeni)

Primer upotrebe:

|  |  |
| --- | --- |
| 1  2 | $numbers = "123456789123456789";  echo substr\_replace($numbers, "#########",2,8); |

Nakon izvršenja biće ispisano:

12#########23456789

**Vežba 1**

***Problem:***

U aplikaciju ulaze sledeće promenljive:

|  |  |
| --- | --- |
| 1 | $userName i $password |

Potrebno je proveriti validnost ovih promenljivih, i obezbediti da ne budu prazne, kao i da ne sadrže sledeće karaktere: < > i ‘

U slučaju da je neka od promenljivih prazna, izvršenje se prekida, dok u slučaju postojanja neželjenih karaktera, dolazi do izbacivanja tih karaktera iz vrednosti promenljive.

Inicijalne vrednosti promenljivih treba da budu:

|  |  |
| --- | --- |
| 1  2 | $userName = "myName<";  $password = "myPa>sswo'rd"; |

***Rešenje:***

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <?php  function clean\_string($string)  {      $string = str\_replace("<", "", $string);      $string = str\_replace(">", "", $string);      $string = str\_replace("'", "", $string);      return $string;  }  $userName = "myName<";  $password = "myPa>sswo'rd";  $userName = clean\_string(trim($userName));  $password = clean\_string(trim($password));  if(trim($userName) == "" || trim($password) == "")      die("invalid credentials");  echo "valid credentials";  ?> |

Ispis na izlazu:

valid credentials

U postavci zadatka su date inicijalne vrednosti za promenljive $userName i $password. Da bismo obradili ovakve vrednosti, kreiraćemo korisnički definisanu funkciju koja će imati za cilj da se bavi samo time. Funkciju ćemo nazvati clean\_string. Unutar funkcije nad parametrom pozivamo već poznatu ugrađenu funkciju str\_replace. Ovu funkciju ćemo pozvati tri puta, da bismo zamenili svaki od karaktera, koji su postavkom zadataka traženi. Nakon ovih izmena, funkcija vraća obrađeni string.  
  
Pri pozivu funkcije clean\_string kao argument prosleđujemo vrednost koju vrati ugrađena funkcija trim nakon što joj kao argument prosledimo string koji se obrađuje.  
  
Na samom kraju, proverava se da li su stringovi prazni i ukoliko jesu, vršimo prekidanje koda:

|  |  |
| --- | --- |
| 1 | die("invalid credentials"); |

**Vežba 2**

Dat je sledeći string:

|  |  |
| --- | --- |
| 1 | $lipsum = "Lorem Ipsum is simply dummy text of the printing and typesetting industry. "; |

Potrebno je formatirati tekst tako da se formira novi red nakon svakog 15. karaktera:

***Rešenje 1:***

Ovo rešenje funkcioniše samo kod kontrola koje imaju mogućnost prepoznavanja novog reda u tekstu):

|  |  |
| --- | --- |
| 1  2 | $lipsum = "Lorem Ipsum is simply dummy text of the printing and typesetting industry.";  echo wordwrap($lipsum, 15, "<br>"); |

***Rešenje 2:***

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | $lipsum = "Lorem Ipsum is simply dummy text of the printing and typesetting industry.";  $lipsumArr = explode(" ", $lipsum);  $tmpContent = "";  for( $i = 0; $i < sizeof( $lipsumArr ); $i++ )      {          if((strlen($tmpContent . " ") + strlen($lipsumArr[$i])) < 15)              $tmpContent .= " " . $lipsumArr[$i];          else              {                  if($tmpContent != "")                      {                          echo $tmpContent . "<br>";                      }                          $tmpContent = $lipsumArr[$i];              }      }  echo $tmpContent; |

$lipsum je promenljiva koja sadrži tekst koji će biti obrađen. Kako ne želimo da prelazak u novi red izvršimo na polovini reči, moramo čitav string razdvojiti na elemente niza. Upravo to postižemo metodom explode(), koja prihvata dva parametra. Prvi prametar je separator, dok je drugi parametar string koji se obrađuje. Sada imamo pripremljen niz reči.  
  
Kreiramo promenljivu $tempContent koja će privremeno zadržavati string koji treba prikazati na strani. Upotrebnom for petlje na do sada utvrđen način, prolazimo kroz sve elemente niza (sve reči).  
  
Unutar petlje proveravamo da li je broj karaktera stringa, koji trenutno predstavlja sadržaj promenljive $tmpContent, nakon konkatenacije sa razmakom i nakon sabiranja sa brojem karaktera reči u nizu, koja odogovara iteraciji, manji od 15. Ukoliko je uslov ispunjen, to znači da reč može stati u red i da treba proveriti da li je moguće u isti red postaviti još jednu reč.   
  
Ukoliko ovaj uslov nije ispunjen, to znači da za reč nema mesta u redu i da treba taj red ispisati na strani.

**Vežba 3**

***Problem:***

Dat je sledeći string:

|  |  |
| --- | --- |
| 1 | $lipsum = "Lorem Ipsum is simply dummy text of the printing and typesetting industry."; |

Potrebno je formatirati string, tako da, ukoliko ima više od 15 karaktera, bude isečen, tako da poslednja tri karaktera do 15. karaktera budu tačke. Na primer:

Lorem Ipsum ...

***Rešenje:***

|  |  |
| --- | --- |
| 1  2  3  4  5 | <?php  $lipsum = "Lorem Ipsum is simply dummy text of the printing and typesetting industry.";  $izlaz = (strlen( $lipsum ) > 15) ?substr( $lipsum, 0, 12 )."...": $lipsum;  echo $izlaz;  ?> |

Zadatak koji je ovde prikazan predstavlja jedostavan primer upotrebe funkcije substr() koja radi sa tri parametra. Prvi parametar je string koji se obrađuje, drugi parametar je početni indeks za odsecanje stringa, dok je treći broj karaktera koji će biti uzet u obzir, počevši od početnog indeksa. Takođe, iskorišćen je i ternarni operator. U zagradi za ovaj operator postavljamo uslov nakon kog sledi znak pitanja i dve moguće vrednosti za string. Stringovi se odvajaju simbolom dvotačke (:).  
  
Nakon izvršavanja ovog operatora potrebno je na strani prikazati vrednost promenljive koja čuva izlaz.

**Vežba 4**

***Problem:***

Potrebno je napisati program koji će za navedeni string vraćati broj reči koji se nalazi u njemu; string promenljiva ima sadržaj:

"Lorem Ipsum is simply dummy text of the printing and typesetting industry."

***Rešenje:***

|  |  |
| --- | --- |
| 1 | echo str\_word\_count("Lorem Ipsum is simply dummy text of the printing and typesetting industry."); |

Rešenje nakon izvršenja programa je 12!

Cilj ove vežbe je prikaz zanimljive funkcije str\_word\_count, koja prihvata jedan obavezni i dva opciona parametra. Funkcija prebrojava reči koje se nalaze u stringu. Definisanjem drugog parametra može se zahtevati od funkcije da vrati specifičan rezultat.  
  
Moguće vrednosti za drugi parametar:

* 0 – Podrazumevana vrednost. Vraća broj reči u stringu;
* 1 – Vraća niz koji kao elemente sadrži reči iz datog stringa;
* 2 – Vraća niz reči u kojem su ključevi pozicije reči u stringu.

**Kako će izgledati rezultat na strani nakon izvršenja sledećeg koda? $name = "James";   
echo 'My name is $name';**

 My name is $name

 My name is James

 'My name is $name'

 'My name is James'

 Na ekranu će biti prikazana greška