Hlavní funkce pro práci s řetězci

strtolower()

Převede velká písmena v řetězci na malá

```
echo strtolower ("PROgramáTOR");          // vypíše "programátor"
```

strtoupper()

Převede malá písmena v řetězci na velká

```
echo strtoupper ("PROgramáTOR");          // vypíše "PROGRAMÁTOR"
```

strlen()

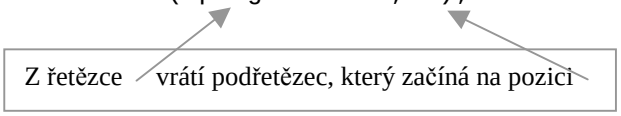
Zjistí délku řetězce

```
echo strlen ("programátor");              // vypíše 11
```

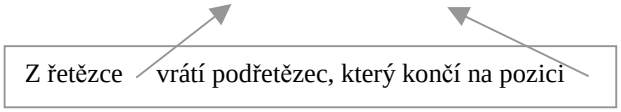
substr()

Vrátí část řetězce

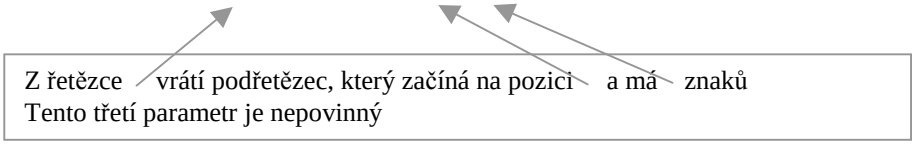
```
echo substr ("programátor", 3);           // vypíše "gramátor"
```




```
echo substr ("programátor", -3);          // vypíše "tor"
```

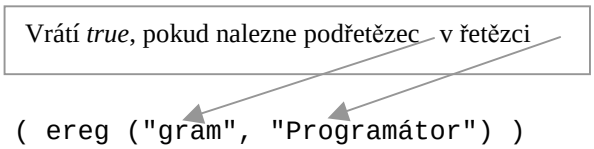



```
echo substr ("programátor", 3, 4);        // vypíše "gram"
```



ereg()

Hledá konkrétní výraz v textovém řetězci. Podobnou funkcí je **eregi()**, která se liší pouze v tom, že není citlivá na velikost písmen.

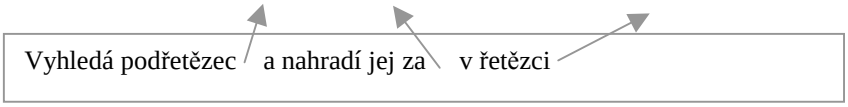



```
if ( ereg ("gram", "Programátor") )
 echo "Obsahuje daný podřetězec";        // vypíše, podmínka je splněna

if ( eregi ("^program", "Programátor") )
 echo "Řetězec začíná daným podřetězcem";

if ( eregi ("tor$", "Programátor") )
 echo "Řetězec končí daným podřetězcem";
```

str_replace

Nahradí všechny výskyty jednoho řetězce v dalším řetězci

```
echo str_replace ("@", "[zaviná č]", "programator@programuje.cz");
```




```
                                  // vypíše"programator[zaviná č]programuje.cz"
```

trim()

Odstraní netisknutelné znaky (mezery, tabulátory apod.) ze začátku a konce řetězce

```
echo trim (" programátor ");       // vypíše "programátor"
```

ltrim()

Odstraní netisknutelné znaky ze začátku řetězce

```
echo ltrim (" programátor ");      // vypíše "programátor "
```

chop ()

Odstraní netisknutelné znaky z konce řetězce

```
echo chop (" programátor ");       // vypíše " programátor"
```

strip_tags()

Odstraní z řetězce HTML a PHP tagy

```
echo strip_tags ("Máme <B>tučný</B> text.");    // vrátí k výpisu řetězec
                                                // "Máme tučný text."
```

explode()

Podle oddělovače rozdělí řetězec na několik částí a vytvoří z nich pole

```
$pole = explode ("@", "programator@programuje.cz");
echo $pole[0];          // vypíše "programator"
echo $pole[1];          // vypíše " programuje.cz "
```

implode()

Spojí prvky pole do řetězce a oddělí je oddělovačem. Funkce je totožná s funkcí **join()**.

```
$pole = array ("Programátor", "v",  "PHP");
echo implode ($pole, " ");          // vypíše "Programátor v PHP"
echo implode ($pole, "--");         // vypíše "Programátor--v--PHP"
```

chunk_split()

Rozdělí řetězec na několik částí o dané délce a oddělí je oddělovačem.

```
echo chunk_split ("programátor", 2, ".");  // vypíše "pr.og.ra.má.to.r."
echo chunk_split ("582333181", 3);         // vypíše "582 333 181"
```

Pokud není oddělovač zadán, je použita mezera.

strtr()

Přeloží určité znaky. Parametry této funkce jsou 1) vstupní řetězec 2) řada znaků, které se mají nahradit 3) řada znaků, kterými se má nahradit. Lze využít pro odstranění diakritiky:

```
$retezec = "Příliš žluťoučký kůň úpěl ďábelské ódy";
echo strtr ($retezec,  "áäčďéěëíňóöřšťúůüýžÁÄČĎÉĚËÍŇÓÖŘŠŤÚŮÜÝŽ",
                       "aacdeeeeinoorstuuuyzAACDEEEINOORSTUUUYZ");
       // vypíše "Prilis zlutoucky kun upel dabelske ody"
```

strpos()

Vrátí pozici prvního výskytu podřetězce v řetězci

```
echo strpos ("programátor", "gram");         // vypíše 3
```

Pokud není řetězec nalezen, vrátí `false`. Pro porovnávání použít operátor `===`

```
if ( strpos ("programátor", "php") === false)
 echo "Řetězec nenalezen";
```

ord()

Vrátí ASCII hodnotu znaku

```
echo ord ("@");                // vypíše 64
```

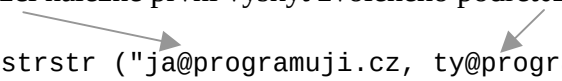
chr()

Vrátí znak určený ASCII hodnotou v parametru

```
echo chr (64);                 // vypíše "@"
```

strstr()

V řetězci nalezne první výskyt zvoleného podřetězce a vrátí jej se zbytkem řetězce

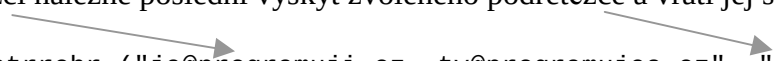


```
echo strstr ("ja@programuji.cz, ty@programujes.cz", "@");
      // vrátí "@programuji.cz, ty@programujes.cz"
```

Podobnou funkcí je **stristr()**, která se liší pouze v tom, že není citlivá na velikost písmen.
Obdobnou funkcí je **strchr()**, která však hledá první výskyt pouze jednoho znaku.

strrchr()

V řetězci nalezne poslední výskyt zvoleného podřetězce a vrátí jej se zbytkem řetězce



```
echo strrchr ("ja@programuji.cz, ty@programujes.cz", "@");
      // vrátí " @programujes.cz"
```

nl2br()

Před všechny konce řádků vloží HTML tag pro nový řádek

```
echo nl2br ("php \n v praxi");        // vypíše:
```

```
php <br />
 v praxi
```

addslashes()

Přidá do řetězce zpětná lomítka \ před uvozovky a apostrofy

```
echo addslashes ("O'Brien");            // vypíše "O\'Brien"
echo addslashes ('Petr "Obr" Novák');   // vypíše "Petr \"Obr\" Novák"
```

stripslashes()

Odstraní zpětná lomítka \

```
echo stripslashes ("O\'Brien");             // vypíše "O'Brien"
```

rawurlencode()

Zakóduje řetězec jako URL (upraví nealfanumerické znaky mimo _-.)

```
echo rawurlencode ("programátor PHP");       // vypíše "program%E1tor%20PHP"
```

rawurldecode()

Dekóduje řetězec zakódovaný jako URL

```
echo rawurldecode ("program%E1tor%20PHP");    // vypíše "programátor PHP"
```

htmlspecialchars()

Přeloží ampersand (&), uvozovku ("), apostrof ('), menší než (<) a větší než (>) na HTML entity

```
echo htmlspecialchars ('programátoři <B>v "PHP"</B>');
        // vypíše "programátoři &lt;B&gt;v &quot;PHP&quot;&lt;/B&gt;"
echo htmlspecialchars ("programátoři <B>v 'PHP'</B>");
        // vypíše "programátoři &lt;B&gt;v 'PHP'&lt;/B&gt;"
```

V posledním příkladu zůstaly apostrofy nepřeloženy. Lze připsat druhý volitelný argument (defaultně je ENT_COMPAT), který převede i je:

```
echo htmlspecialchars ("programátoři <B>v 'PHP'</B>", ENT_QUOTES);
        // vypíše "programátoři &lt;B&gt;v &#039;PHP&#039;&lt;/B&gt;"
```

Při použití argumentu ENT_NOQUOTES zůstanou bez překladu apostrofy i uvozovky.

htmlentities()

Plní stejnou funkci jako **htmlspecialchars()**, jen jinak interpretuje znakové sady (např. češtinu)

```
echo htmlentities ('programátoři <B>v PHP</B>');
        // vypíše "program&aacute;to&oslash;i &lt;B&gt;v PHP&lt;/B&gt;"
        // na obrazovce: "programátoøi <B>v PHP</B>"
```

htmlspecialchars_decode()

Opak funkce **htmlspecialchars()**. Převede speciální HTML entity zpátky na znaky. Funkce je přístupná až od PHP verze 5.1.0 a druhý nepovinný parametr může být stejný jako u funkce **htmlspecialchars()**.

substr_count ()

Spočítá, kolikrát se v řetězci vyskytuje daný podřetězec

```
echo substr_count("jak si žije kajak?", "jak");      // vypíše 2
```

strrev()

Obrátí řetězec

```
echo strrev ("programátor");              // vypíše "rotámargorp"
```

ucwords()

Změní první znak každého slova v řetězci na velké písmeno

```
echo ucwords ("programátoři v PHP");      // vypíše "Programátoři V PHP"
```

ucfirst ()

Změní první písmeno řetězce na velké

```
echo ucfirst ("programátoři v PHP");      // vypíše "Programátoři v PHP"
```

similar_text ()

Spočítá podobnost dvou řetězců. Vrátí pouze počet shodných znaků

```
echo similar_text ("programátor", "programy");       // vypíše 7
```

Pokud je zadán třetí argument (typu integer, předávaný odkazem), uloží se do něj podobnost v procentech:

```
$v_procentech = 0;
echo similar_text ("programátor", "programy", $v_procentech);   // vypíše 7
echo $v_procentech;     // vypíše 73.684210526316
```

crypt()

Jednosměrné zašifrování řetězce podle libovolného (defaultně dvoupísmenného) základu

```
echo crypt ("programátor", "ab");              // vypíše "abyvBCT2X0aSI"
```

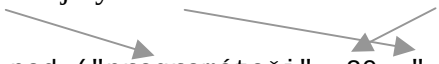
strspn ()

Spočítá počet znaků od začátku řetězce, které vyhovují zadaným znakům.

```
echo StrSpn ("007 Bond", "1234567890");  // vypíše 3
```

str_pad ()

Doplní řetězec jiným řetězcem na určitou délku.



```
echo str_pad ("programátoři", 20, ".");  // vypíše "programátoři........"
```

Při vynechání třetího parametru bude doplněn o mezery.

str_repeat

Opakuje řetězec n krát

```
echo str_repeat ("php", 3);              // vypíše "phpphpphp"
```

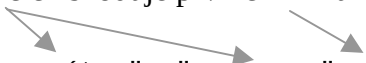
str_shuffle

Náhodně přehází znaky v řetězci

```
echo str_shuffle ("programátor");        // vypíše např. "pramráotgro"
```

strncmp()

Porovná, zda se v řetězcích shoduje prvních n znaků



```
echo strncmp ("programátor", "programy", 6);    // vypíše 0  (shoda)
echo strncmp ("programátor", "abcd", 2);        // vypíše 1  (1. řetězec < 2.)
echo strncmp ("programátor", "zxyw", 2);        // vypíše -1 (1. řetězec > 2.)
```

Podobnou funkcí je **strncasecmp ()**, která se liší pouze v tom, že není citlivá na velikost písmen.