Structure

file.php

```
<HTML><HEAD><TITLE></TITLE></HEAD>
<BODY>
<?
/* comment */
// comment
function eg($arg1=3,$var2="default")
global $fred;
static $bob;
        for ($i=0, $i<$arg1; $i++) {
?>
        <BR>
<?
        }
        echo "$var2\n";
        return 42;
}
?>
<H1>heading</H1>
<?
$ct=2;
eg(&$ct,"Jason"); # $ct is pass-by-ref
</BODY></HTML>
```

Variables

Integer: 34, 0x16, 020 Float: 3.49e-1
Boolean: non-zero/non-empty any type; FALSE, TRUE
String: "var=$var\n" or 'no $var\n' or <<< ENDNAME
  …. ENDNAME
Object: class bob {$var="default"; function fred…};
  $obj=new bob; echo $bob->var;bob->fred();

Miscellaneous

Operators: as in C with And,Xor,Or at lowest precedence,
  also `op sys command`
Control structures: as in C

Web

Cookies
if (isset($_COOKIE['flavor'])) { echo "Flavor is
$_COOKIE[flavor]"}
foreach($_COOKIE as $cook_name=>$cook_val) { }
setcookie('flavor','value'[,time) - time()-86400 will age
(delete)
$browser=get_browser(); if ($browser->frames or
$browser->tables …
flush() – forces send to browser while building rest of page
$_GET['paramname']
$_POST['fieldname']
if ($_SERVER['REQUEST_METHOD']=='GET') …
if (is_array($_GET['checkboxname'])) …

Arrays

Array: $var[0]="Hello"; $var["fred"]=9;$str[2]='c'
$ary=array('Fred','bob');
$ary=array('joeidx'=>'mary','danidx'=>'sally')
  echo "$ary[joeidx]";
foreach($ary as $var [=>$idx]) {}
$ary[]="endofarray";
count($ary) equals sizeof($ary)
$ary=range(2,5);$revary=range(5,2);$ltrs=range('a','z');
list($fred,$bob)=$ary;
$list($fred,$bob)=slice_array($ary,start,length)
array_pad($ary,finallength,padvalue);
$keys=array_keys($ary); $vals=array_values($ary);
if (key_exists(key,$ary)) …
$deleted=array_splice($ary,start,length[,replace]) (length
  omitted-> to end)
extract($assocary) creates vars w/ key names;
  $ary=compact('varname1','varname2') reverses
iterators: current(),key(),reset(),next(),prev(),end(),each()
  where each does a current, key and next
array_walk($ary,function);
  $res=array_reduce($ary,cumfunc)
if (in_array(to_find,$ary)) {}
sort($ary), asort (keeps indexes), ksort (by keys), ?rsort
  (rvrs)
array_sum($ary), array_reverse(), array_flip() key<->val
$ary=array_merge($ary1,$ary2,..)
array_unique, array_diff($ary1,$ary2, …) (in 1, but not 2)
array_push, array_pop
array_filter($ary,boolfunc)

Strings

echo "string";
printf("format",…)
var_dump($var) – debug formatted dump
trim, rtrim, ltrim, strtolower, strtoupper, ucfirst, ucwords
htmlentities,htmlspecialchars("str&"),strip_tags("<P>fred");
==,!=, <, <=, etc
substr(str,start[,end]);
$pos=strpos(str,search); strrpos (from end)
strtok, sscanf, sprintf
$ary=explode(separator,string[,limit]);
implode(seprtr,array);
$ary=parse_url(string)
$bool=ereg(regexp,string[,captured])
$chgd=ereg_replace(regexp,replace,string);
$ary=split(splitexp,string);
Perl type: /expr/[i] or {exp}[i] etc
  ( ) and | allowed ( ) captures unless (?:…)
anchors: ^ (beg line), $ (end line),\b (word boundary)
matches: (char), [one of], text repetitions: ? (0 or 1), * (0
or more) + (1 ore more) {n} {n,} {m,n}
$bool=preg_match('/exp/[i]',string[,subexprary])
$str=preg_replce(pat,repl,str); $str=preg_split(splpat,str)
$subary=preg_grep(pattern,$ary)