```php
<?php

  /*get_class_methods
    Retorna um vetor com os nomes dos métodos de uma determinada classe.
      array get_class_methods(string nome_classe)
  */

  class Funcionario
  {
   function SetSalario()
   {
   }

   function GetSalario()
   {
   }

   function SetNome()
   {
   }

   function GetNome()
   {
   }

  }

  print_r(get_class_methods('Funcionario'));

  /*
   get_class_vars:

  Retorna um vetor com os nomes das propriedades e conteúdos de uma determinada
classe.

  array get_class_vars(string nome_classe)
  */

  class Funcionario
  {
   public $Codigo;
   public $Nome;
   public $Salario = 760;
   public $Departamento = 'Contabilidade';

   function SetSalario()
   {
   }

   function GetSalario()
   {
   }

  }

  print_r(get_class_vars('Funcionario'));
```

```php
/*
  get_object_vars:
  Retorna um vetor com os nomes e conteúdos das propriedades de um objeto.
São valores dinâmicos que se alteram de acordo com o ciclo
  de vida do objeto.
  array get_object_vars(object nome_objeto)
*/

class Funcionario
{

 public $Codigo;
 public $Nome;
 public $Salario = 760;
 public $Departamento = 'Contabilidade';

 function SetSalario()
 {
 }

 function GetSalario()
 {
 }
}

$jose = new Funcionario;
$jose->Codigo = 44;
$jose->Nome = 'José da Silva';
$jose->Salario += 100;
$jose->Departamento = 'Financeiro';

print_r(get_object_vars($jose));

/*
 get_class:
 Retorna o nome da classe a qual um objeto pertence.
 string get_class(object nome_objeto)
*/
 class Funcionario
{

 public $Codigo;
 public $Nome;
 public $Salario = 760;
 public $Departamento = 'Contabilidade';

 function SetSalario()
 {
 }

 function GetSalario()
 {
 }
}

$jose = new Funcionario;
```

```php
echo get_class($jose);

/*
 get_parent_class
 Retorna o nome da classe ancestral. Se o parâmetro for um objeto, retorna
o nome da classe ancestral da classe
 à qual o objeto pertence. Se o parâmetro for uma string, retorna o nome
da classe ancestral da classe passada  como
 parâmetro.

 string get_parent_class(mixed objeto)
*/

class Funcionario
{

 public $Codigo;
 public $Nome;
}

class Estagiario extends Funcionario
{
 public $Salario;
 public $bolsa;
}

$jose = new Estagiario;

echo get_parent_class($jose);
echo "<br>\n";
echo get_parent_class('Estagiario');

/*
  is_subclass_of:
  Indica se um determinado objeto ou classe é derivado de uma determinada
classe.

  boolean is_subclass_of(mixed objeto, string classe)
*/

class Funcionario
{

 public $Codigo;
 public $Nome;
}

class Estagiario extends Funcionario
{
 public $Salario;
 public $bolsa;
}

$jose = new Estagiario;

if (is_subclass_of($jose, 'Funcionario'))
{
```

```php
   echo "Classe do objeto José é derivada de Funcionário";
  }

  echo "<br>\n";

  if (is_subclass_of('Estagiario', 'Funcionario'))
  {
   echo "Classe do objeto Estagiário é derivada de Funcionário";
  }

  /*
    methodo_exists:
    Verifica se um determinado objeto possui o método descrito. Podemos
verificar a existência de um método antes de executar por engano
    um método existente.

    boolean method_exists(object objeto, string método)
  */

  class Funcionario
  {

   public $Codigo;
   public $Nome;
  }

  $jose = new Funcionario;

  if (method_exists($jose, SetNome))
  {
   echo "Objeto José possui método SetNome()";
  }

  if (method_exists($jose, SetSalario))
  {
   echo "Objeto José possui método SetSalario()";
  }

  /*
    call_user_func:
    Executa uma função ou um método de uma classe passado como parâmtero. Para
executar uma função, basta passar seu nome como uma string, e,
    para executar um método de um objeto, basta passar o parâmetro como um
array contendo na posição 0 o objeto e na posição 1 o método a
    ser executado. Para executar métodos estáticos, basta passar o nome da
classe em vez do objeto na posição 0 do array.

    mixed call_user_func(callback funcao [,mixed parametro [, mixed...]])
  */

  function minhaFuncao()
  {
   echo "Minha função!!!<br>\n";
  }

  call_user_func('minhaFuncao');
```

```php
  class MinhaClasse
  {
   function MeuMetodo()
   {
    echo "Meu metodo!<br>\n";
   }
  }

  call_user_func(array('MinhaClasse', 'MeuMetodo'));

  $obj = new MinhaClasse();

  call_user_func(array($obj, 'MeuMetodo'));


?>
```