Práctico Nro. 5 – Arreglos Asociativos 2007

Aplique una buena modularización a las soluciones y realice los diagramas de estructura correspondientes.

1. Escriba al menos 3 ejemplos de arreglos asociativos que considere de utilidad. Ejemplo: datos personales

```
array(
    'nombre' => 'Juan',
    'apellido' => 'Perez',
    'nacimiento' => '12/04/1976',
    'dni' => 25346471
);
```

2. Escriba procedimientos/funciones que reciban un arreglo asociativo y construyan arreglos con índices numéricos (al estilo de los trabajados en el práctico anterior), uno con las claves y otro con los valores del arreglo dado.

Ejemplo: dado

```
$arreglo = array('a' => 'primero', 'b' => 'segundo', 'c' => 'tercero');
```

retornar

```
$claves = array('a', 'b', 'c');
$valores = array('primero', 'segundo', 'tercero');
```

3. Realice un procedimiento/función que, dado un arreglo asociativo y un valor a buscar, retorne la clave que tiene el valor en el arreglo; en caso de no encontrarlo, debe retornar `false`. ¿Puede utilizar la solución del ejercicio anterior para resolverlo?

Ejemplo: dados

```
$arreglo = array('a' => 'primero', 'b' => 'segundo', 'c' => 'tercero');
$valor = 'segundo';
```

se debe retornar “b”.

4. Escriba un procedimiento/función que dado un arreglo asociativo cree otro con las claves y valores del arreglo original intercambiados.

Por ejemplo, si se recibe:

```
array(“clave1” ⇒ “valor1”, “clave2” ⇒ “valor2”)
```

se debe retornar:

```
array(“valor1” ⇒ “clave1”, “valor2” ⇒ “clave2”)
```

¿Qué pasaría si alguno de los valores del arreglo dado es un float, un boolean o un array?

5. Suponga el siguiente arreglo: `$arreglo = array(“2007” ⇒ 5, 2006 ⇒ 20)`

- ¿Cuál es la diferencia entre las claves?
- ¿Tiene sentido acceder a `$arreglo` por posicion? Es decir, ¿`$arreglo[0]` que retornaría?
- ¿Cómo queda `$arreglo` luego de ejecutar la instrucción `$arreglo[] = 50`?

6. Dado un arreglo asociativo que contiene fechas como claves y la precipitación de lluvia en esa fecha como valores, cree un arreglo multidimensional donde cada dimensión represente los años, los meses y los días y el valor sea el de la precipitación correspondiente a la fecha. Asuma que todas las claves tienen el formato correcto de fecha: 'dd/mm/aaaa'.

Por ejemplo, dado:

```
$datos = array(
  '05/10/2007' => 11,
  '06/10/2007' => 4,
  '10/07/2007' => 6,
  '08/09/2005' => 17
);
```

El resultado debe ser:

```
$resultado = array(
  2007 => array(
    10 => array(
      5 => 11
      6 => 4
    ),
    7 => array(10 => 6)
  ),
  2005 => array(
    9 => array(8 => 17)
  )
);
```

7. Dada una matriz que contiene precipitaciones diarias, con la estructura indicada en el ejercicio anterior, y dos valores de año, crear un arreglo que contenga los promedios mensuales entre esos dos años. Asuma que todos los meses tienen 30 días y que no hubo precipitaciones en las fechas que no se indican en la matriz.

Por ejemplo, dados:

```
$precipitaciones = array(
  2006 => array(
    10 => array(
      4 => 22, 5 => 11, 6 => 4
    ),
    7 => array(10 => 6)
  ),
  2005 => array(
    10 => array(8 => 17)
  ),
  2004 => array(
    6 => array(
      14 => 20,
      15 => 12
    )
  )
);

$desde = 2005;

$hasta = 2006;
```

El resultado para `$precipitaciones` entre `$desde` y `$hasta` debe ser:

```
$resultado = array(
  1 => 0,
  2 => 0,
  3 => 0,
  4 => 0,
  5 => 0,
  6 => 0, // 14/06/2004 y 15/06/2004 no están entre 2005 y 2006
  7 => 0.1, // (6 en 10/07/2006) / 60
  8 => 0,
  9 => 0,
  10 => 0.9, // (22 en 04/10/2006 + 11 en 05/10/2006
             //  + 4 en 06/10/2006 + 17 en 08/10/2005) / 60
  11 => 0,
  12 => 0
);
```

8. Dado un arreglo de palabras, construir un arreglo asociativo que contenga las mismas palabras separadas por caracteres, de manera que la dimensión *i* del arreglo tenga el caracter *i* de cada palabra.

Por ejemplo, dado:

```
$palabras = array(
  'una', 'lista', 'unica', 'local'
);
```

El resultado debe ser:

```
$resultado = array(
  // primera dimensión: primer letra de las palabras
  'l' => array(
    // segunda dimensión: segunda letra de las palabras empezadas en 'l'
    'i' => array(
      // tercera dimensión: tercer letra de las palabras empezadas en 'li'
      's' => array(
        't' => array(
          'a' => array()))
    ),
    'o' => array(
      'c' => array(
        'a' => array(
          'l' => array()))
    )
  ),
  'u' => array(
    'n' => array(
      'a' => array(),
      'i' => array(
        'c' => array(
          'a' => array()))
    )
  )
);
```

Las estructuras de este tipo se denominan *Trie*s (http://es.wikipedia.org/wiki/Trie – http://en.wikipedia.org/wiki/Trie).

9. Dado un string y la estructura creada en el ejercicio anterior, escriba una función que permita determinar si el string es un prefijo de una palabra existente en la estructura.

Siguiendo con el ejemplo anterior, “lis” es prefijo de una palabra, pero “lod” no.