RED ACADEMY

PHP Arrays: Function Scavenger Hunt!

Below we have two arrays containing similarly formatted data representing two movies.

Your task is to search the PHP documentation (http://php.net/manual/en/ref.array.php) for array functions and figure out what built-in function (or combination of functions) will return the data formatted as required in each of the challenges.

Paste this code into an `index.php` file(between `<?php ?>` tags of course!) to get started...

```
$movies_1 = array(
  array(
    'ID'    => 1,
    'title' => 'Jurassic Park',
    'genre' => 'Adventure',
    'year'  => '1993',
    'stars' => array( 'Sam Neill', 'Laura Dern', 'Jeff Goldblum' )
  ),
  array(
    'ID'    => 2,
    'title' => 'Back to the Future',
    'genre' => 'Sci-fi',
    'year'  => '1985',
    'stars' => array( 'Michael J. Fox', 'Christopher Lloyd' )
  )
);

$movies_2 = array(
  array(
    'ID'    => 3,
    'title' => 'Die Hard',
    'genre' => 'Action',
    'year'  => '1988',
    'stars' => array( 'Bruce Willis', 'Alan Rickman' )
  ),
  array(
    'ID'    => 4,
    'title' => 'The Breakfast Club',
    'genre' => 'Drama',
    'year'  => '1985',
    'stars' => array( 'Emilio Estevez', 'Judd Nelson', 'Molly Ringwald' )
  ),
);
```

CHALLENGE #1

It seems it might be easier to deal with all of this movie data if all four movies were combined into one array. Find the appropriate function to **combine both movie arrays** into one array in order to produce this output when running `print_r()` on the result:

```
Array
(
    [0] => Array
        (
            [ID] => 1
            [title] => Jurassic Park
            [genre] => Adventure
            [year] => 1993
            [stars] => Array
                (
                    [0] => Sam Neill
                    [1] => Laura Dern
                    [2] => Jeff Goldblum
                )
        )
    [1] => Array
        (
            [ID] => 2
            [title] => Back to the Future
            [genre] => Sci-fi
            [year] => 1985
            [stars] => Array
                (
                    [0] => Michael J. Fox
                    [1] => Christopher Lloyd
                )
        )
    [2] => Array
        (
            [ID] => 3
            [title] => Die Hard
            [genre] => Action
            [year] => 1988
            [stars] => Array
                (
                    [0] => Bruce Willis
                    [1] => Alan Rickman
                )
        )
    [3] => Array
        (
            [ID] => 4
            [title] => The Breakfast Club
            [genre] => Drama
            [year] => 1985
            [stars] => Array
                (
                    [0] => Emilio Estevez
                    [1] => Judd Nelson
                    [2] => Molly Ringwald
                )
        )
)
```

CHALLENGE #2

Now we'd like to create a new array that **only contains** all of our array **keys** for the second movie (Back to the Future) in the first movies arrays (`$movies_1`). Find the function that creates the new array and generates this output when running `print_r()` on the result:

```
Array
(
    [0] => ID
    [1] => title
    [2] => genre
    [3] => year
    [4] => stars
)
```

CHALLENGE #3

Now let's go back to `$movies_1` array and **sort the stars into alphabetical order** for both movies in that array. We want to do this as efficiently as possible, and ensure that if new movies are added to the array that they will automatically be sorted as well (use a `foreach` loop perhaps?).

And there's a gotcha built into this one...you'll want make sure that when you run your foreach loop that you working on a **reference** of each item.

```
Array
(
    [0] => Array
        (
            [ID] => 1
            [title] => Jurassic Park
            [genre] => Adventure
            [year] => 1993
            [stars] => Array
                (
                    [0] => Jeff Goldblum
                    [1] => Laura Dern
                    [2] => Sam Neill
                )
        )
    [1] => Array
        (
            [ID] => 2
            [title] => Back to the Future
            [genre] => Sci-fi
            [year] => 1985
            [stars] => Array
                (
                    [0] => Christopher Lloyd
                    [1] => Michael J. Fox
                )
        )
)
```

CHALLENGE #4

Last but not least, let's try combining the two arrays into one again, and then search through the resulting array to **return the key** for the **first movie** found that was **released in 1985**. When you `echo` out the result of the search, you should see the following (because Back to the Future is at the `1` index in the array...):

```
1
```

Bonus Points!!!

Now we actually want to create a new array of **all the keys for all the movies** in our combined array that correspond to movies **released in 1985**. This task is a bit more complex than the last, and doesn't involve any of the built-in array functions—just a plain ol' `foreach` loop with a conditional check in it.

Try writing the code required to produce this output (which is an array containing the keys for Back to the Future and The Breakfast Club from our combined array):

```
Array
(
    [0] => 1
    [1] => 3
)
```

Hint: You'll need to create an empty array to hold the output of each iteration of your loop where the year value matches what you're looking for (before you start looping!).