ЛАБОРАТОРНАЯ РАБОТА №3.2. ФУНКЦИИ ДЛЯ РАБОТЫ С МАССИВАМИ

Цель работы: познакомиться с функциями для работы с массивами.

Задание 0. Создание массива, добавление и удаление элементов, сортировка.

1. Создайте массив, содержащий цвета (‘red’, ‘green’ и т.д. 10 элементов)
2. Выведите массив с помощью var_damp() и print_r()
3. Выведите второй элемент массива
4. Выведите последний элемент массива
5. Измените название третьего цвета, добавьте приставку “deep”
6. Удалите первый элемент массива
7. Удалите элемент по значению
8. Добавьте с помощью команды еще одно значение в массив.
9. Выведите на экран данный массив, записанный в обратном порядке, используйте функция array_reverse() (Возвращает массив с элементами в обратном порядке)
10. Выведите размер массива
11. Отсортируйте массив по возрастанию

Задание 1. Дан массив:

```
$goods = [
      ['id'=>1, 'title'=>'milk', 'price'=>67, 'amount'=>67],
      ['id'=>2, 'title'=>'coffe', 'price'=>500, 'amount'=>12],
      ['id'=>3, 'title'=>'tea', 'price'=>99, 'amount'=>20],
      ['id'=>4, 'title'=>'bread', 'price'=>160, 'amount'=>3]
];
```

1. Выведите с помощью функции array_column (Возвращает массив из значений одного столбца входного массива) только названия продуктов.
2. Выведите массив, содержащий ключи одного из массивов, входящих в массив $goods.
3. Выполните сортировку товаров, содержащихся в массиве по убыванию цены.

Используйте функцию usort и пример:

```php
$products = [
    ['name' => 'iPhone', 'price' => 999],
    ['name' => 'MacBook', 'price' => 1999],
    ['name' => 'iPad', 'price' => 499]
];

// Сортировка по цене (по возрастанию)
usort($products, function($a, $b) {
    return $a['price'] <=> $b['price'];
});
print_r($products);
```

Задание 2. Сформируйте массив из ста нулей. Используйте функцию array_fill(). Выведите массив на экран.

Задание 3. Дан массив $arr = [1,2,3,4,5,6,7,8,9,10]. С помощью функции array_slice создать новые массивы, состоящий из элементов [3,4,5] и [8,9,10].

Задание 4. Дан массив $arr = ['123', 45, 42, 56, 78, 45.6, 'hello', [1,5, 1.5]]

С помощью функции array_filter сформировать новые массивы.

1. Массив только из целых чисел.

Подсказка:

```php
$value = 42;
var_dump(is_int($value)); // bool(true)
```

2. Массив только из строковых значений исходного массива

3. Массив только из четных чисел.

Пример:

```php
$numbers = [-1, 2, -3, 4, -5, 6, -7, 8, -9, 10];
// Только четные числа
$even = array_filter($numbers, function($num) {
    return $num > 0;
});
print_r($even);
// Результат: [2, 4, 6, 8, 10]
```

Задание 5. Создайте массив из 100 случайных чисел.

1. С помощью функции array_map перезапишите исходный массив уменьшив его значения на 50%
2. Найти произведение всех элементов массива с помощью функции array_product().

3. Выведите массив отсортированный по возрастанию.

Задание 6. Дан массив, содержащий 10 различных слов. Определить есть ли данное слово в массиве.

Задание 7. Вы разрабатываете систему анализа пользовательских интересов на основе их активности на двух разных платформах (Соцсеть и Форум).

```php
// Интересы пользователя в Соцсети
$socialInterests = [
    'programming' => 5,    // 5 лайков
    'music' => 3,
    'sports' => 8,
    'travel' => 2,
    'gaming' => 6
];
// Интересы пользователя на Форуме
$forumInterests = [
    'programming' => 7,    // 7 постов
    'cooking' => 4,
    'sports' => 3,
    'photography' => 5,
    'gaming' => 2
];
```

1. 📊 Общие интересы (Пересечение)

Найти интересы, которые присутствуют на обеих платформах, и вывести их с суммарным количеством активностей.

Ожидаемый результат:

```
Общие интересы:
- programming: 12 (5 + 7)
- sports: 11 (8 + 3)
- gaming: 8 (6 + 2)
```

2. 🌟 Уникальные интересы (Разность)

Найти интересы, которые есть только в Соцсети и только на Форуме.

Ожидаемый результат:

```
Только в Соцсети: music, travel
Только на Форуме: cooking, photography
```

3. 🔄 Объединение с приоритетом

Создать общий массив интересов, где при конфликте берется максимальное значение.

🛠️ Функции для использования

- `array_intersect_key()` – пересечение по ключам
- `array_diff_key()` – разность по ключам
- `array_merge()` – объединение массивов

Задание 8. Массив из задания 1 вывести на страницу в форме таблицы. Последний столбец в таблице назвать Стоимость и вычислить в нем стоимость товара. Под таблицей вывести итоговую стоимость товара.

📚 Справочник функций для работы с массивами в PHP

📋 Содержание

---

1. Создание и модификация

`array()` – Создание массива

php

```php
$arr = array(1, 2, 3);
$arr = [1, 2, 3]; // Короткий синтаксис (PHP 5.4+)
```

`array_push()` – Добавить в конец

php

```php
$stack = [1, 2];
array_push($stack, 3, 4); // [1, 2, 3, 4]
```

`array_pop()` – Удалить последний

php

```php
$stack = [1, 2, 3];
$last = array_pop($stack); // $last = 3, $stack = [1, 2]
```

`array_unshift()` – Добавить в начало

php

```php
$queue = [2, 3];
array_unshift($queue, 1); // [1, 2, 3]
```

`array_shift()` – Удалить первый

php

```php
$queue = [1, 2, 3];
$first = array_shift($queue); // $first = 1, $queue = [2, 3]
```

`array_merge()` – Объединение

php

```php
$arr1 = [1, 2]; $arr2 = [3, 4];
$merged = array_merge($arr1, $arr2); // [1, 2, 3, 4]
```

`array_splice()` – Замена части

php

```php
$input = [1, 2, 3, 4];
array_splice($input, 1, 2, [5, 6]); // [1, 5, 6, 4]
```

---

2. Поиск и проверка

`in_array()` – Проверить наличие

php

```php
$values = [1, 2, 3];
$exists = in_array(2, $values); // true
```

`array_search()` – Найти ключ

php

```php
$array = ['a' => 1, 'b' => 2];
$key = array_search(2, $array); // 'b'
```

`array_key_exists()` – Проверить ключ

php

```php
$array = ['a' => 1];
$exists = array_key_exists('a', $array); // true
```

`isset()` VS `array_key_exists()`

php

```php
$array = ['a' => null];
```

```php
isset($array['a']); // false
array_key_exists('a', $array); // true
```

---

3. Обход и преобразование

`array_map()` – Применить функцию

php

```php
$numbers = [1, 2, 3];
$squared = array_map(fn($n) => $n * $n, $numbers); // [1, 4, 9]
```

`array_walk()` – Обход с изменением

php

```php
$array = [1, 2, 3];
array_walk($array, function(&$value) {
    $value *= 2;
}); // [2, 4, 6]
```

`array_reduce()` – Свертка

php

```php
$numbers = [1, 2, 3, 4];
$sum = array_reduce($numbers, fn($carry, $item) => $carry + $item, 0); // 10
```

---

4. Сортировка

`sort()` / `rsort()` – По значениям

php

```php
$numbers = [3, 1, 2];
sort($numbers);  // [1, 2, 3]
rsort($numbers); // [3, 2, 1]
```

`asort()` / `arsort()` – Сохраняя ключи

php

```php
$array = ['b' => 2, 'a' => 1];
asort($array);  // ['a' => 1, 'b' => 2]
arsort($array); // ['b' => 2, 'a' => 1]
```

`ksort()` / `krsort()` – По ключам

php

```php
$array = ['b' => 2, 'a' => 1];
ksort($array);  // ['a' => 1, 'b' => 2]
krsort($array); // ['b' => 2, 'a' => 1]
```

`usort()` – Пользовательская

php

```php
$users = [['age' => 25], ['age' => 20]];
usort($users, fn($a, $b) => $a['age'] <=> $b['age']);
```

---

5. Фильтрация

`array_filter()` – Фильтрация

php

```php
$numbers = [1, 2, 3, 4];
$even = array_filter($numbers, fn($n) => $n % 2 === 0); // [2, 4]
```

`array_unique()` – Уникальные

php

```php
$duplicates = [1, 2, 2, 3];
$unique = array_unique($duplicates); // [1, 2, 3]
```

---

6. Математические операции

`array_sum()` – Сумма

php

```php
$numbers = [1, 2, 3];
$sum = array_sum($numbers); // 6
```

`array_product()` – Произведение

php

```php
$numbers = [2, 3, 4];
```

```php
$product = array_product($numbers); // 24
```

`count()` / `sizeof()` – Количество

php

```php
$array = [1, 2, 3];
$count = count($array); // 3
```

---

7. Ключи и значения

`array_keys()` – Получить ключи

php

```php
$array = ['a' => 1, 'b' => 2];
$keys = array_keys($array); // ['a', 'b']
```

`array_values()` – Получить значения

php

```php
$array = ['a' => 1, 'b' => 2];
$values = array_values($array); // [1, 2]
```

`array_flip()` – Поменять местами

php

```php
$array = ['a' => 1, 'b' => 2];
$flipped = array_flip($array); // [1 => 'a', 2 => 'b']
```

`array_key_first()` / `array_key_last()` – Первый/последний ключ

php

```php
$array = ['a' => 1, 'b' => 2];
$first = array_key_first($array); // 'a'
$last = array_key_last($array);   // 'b'
```

---

8. Многомерные массивы

`array_column()` – Извлечь столбец

php

```php
$users = [
    ['id' => 1, 'name' => 'John'],
    ['id' => 2, 'name' => 'Jane']
];
$names = array_column($users, 'name'); // ['John', 'Jane']
```

`array_multisort()` – Множественная сортировка

php

```php
$names = ['John', 'Jane', 'Bob'];
$ages = [25, 30, 20];
array_multisort($ages, SORT_ASC, $names); // Сортировка по возрастанию ages
```

---

9. Утилиты

`array_chunk()` – Разделить на части

php

```php
$array = [1, 2, 3, 4];
$chunks = array_chunk($array, 2); // [[1, 2], [3, 4]]
```

`array_fill()` – Заполнить массив

php

```php
$filled = array_fill(0, 3, 'hello'); // ['hello', 'hello', 'hello']
```

`array_rand()` – Случайный ключ

php

```php
$array = ['a' => 1, 'b' => 2, 'c' => 3];
$randomKey = array_rand($array); // 'a', 'b' или 'c'
```

`shuffle()` – Перемешать

php

```php
$numbers = [1, 2, 3, 4];
shuffle($numbers); // Случайный порядок
```

`array_reverse()` – Обратный порядок

php

```php
$array = [1, 2, 3];
$reversed = array_reverse($array); // [3, 2, 1]
```

🔄 Операции с множествами

`array_intersect()` – Пересечение значений

php

```php
$array1 = [1, 2, 3]; $array2 = [2, 3, 4];
$intersect = array_intersect($array1, $array2); // [2, 3]
```

`array_intersect_key()` – Пересечение ключей

php

```php
$array1 = ['a' => 1, 'b' => 2]; $array2 = ['b' => 3, 'c' => 4];
$intersect = array_intersect_key($array1, $array2); // ['b' => 2]
```

`array_diff()` – Разность значений

php

```php
$array1 = [1, 2, 3]; $array2 = [2, 3, 4];
$diff = array_diff($array1, $array2); // [1]
```

`array_diff_key()` – Разность ключей

php

```php
$array1 = ['a' => 1, 'b' => 2]; $array2 = ['b' => 3, 'c' => 4];
$diff = array_diff_key($array1, $array2); // ['a' => 1]
```

🎯 Практические паттерны

Группировка по ключу

php

```php
$users = [
    ['department' => 'IT', 'name' => 'John'],
```

```php
    ['department' => 'HR', 'name' => 'Jane'],
    ['department' => 'IT', 'name' => 'Bob']
];

$grouped = [];
foreach ($users as $user) {
    $grouped[$user['department']][] = $user['name'];
}
// ['IT' => ['John', 'Bob'], 'HR' => ['Jane']]
```

Быстрый поиск по ключу

php

```php
$users = [
    1 => ['name' => 'John'],
    2 => ['name' => 'Jane']
];

// Вместо перебора используем:
$user = $users[1] ?? null; // Быстрый доступ
```

Трансформация массива

php

```php
$input = [1, 2, 3];
$transformed = array_map(fn($x) => $x * 2, $input); // [2, 4, 6]
```

📊 Чеклист выбора функции

Задача	Функция
Добавить элемент	`array_push()`, `array_unshift()`
Удалить элемент	`array_pop()`, `array_shift()`
Объединить массивы	`array_merge()`
Найти элемент	`in_array()`, `array_search()`
Проверить ключ	`array_key_exists()`
Применить функцию	`array_map()`

Задача	Функция
Отфильтровать	`array_filter()`
Отсортировать	`sort()`, `asort()`, `ksort()`
Сумма/Произведение	`array_sum()`, `array_product()`
Получить ключи/значения	`array_keys()`, `array_values()`
Работа с многомерными	`array_column()`
Операции с множествами	`array_intersect()`, `array_diff()`