**Конспект «Массивы и циклы в PHP»**

**Массив**

Массив — способ хранить несколько значений в одном месте. Например:

['ehm', 'cubic', 'tehnodom', 'dg']

Синтаксис массива такой: начало и конец массива обозначаются квадратными скобками, а все значения, или *элементы массива*, пишутся внутри этих скобок через запятую.

Массив можно записать в переменную, так же как строки и числа.

$название\_массива = [элемент\_1, элемент\_2, элемент\_3];

**Индексы**

Все элементы в массиве имеют порядковый номер, то есть **индекс**. Он позволяет получить доступ к какому-то конкретному элементу массива.

$films = ['Железный человек', 'Мстители', 'Тор', 'Человек-муравей'];

<p><?= $films[0] ?></p> // Выведет на страницу: "Железный человек"

Чтобы получить доступ к элементу массива, нужно написать имя массива и индекс элемента в квадратных скобках. Нумерация элементов в массиве начинается с *нуля*.

**Запись в массив по индексу**

Индексы позволяют не только получать элементы, но и записывать в массив новые данные.

Добавим в массив $favorite\_food новый элемент:

$favorite\_food = ['пюре', 'котлеты', 'борщ'];

$favorite\_food[3] = 'пельмени';

keks\_log($favorite\_food[3]); // Выведет: "пельмени"

Изменим значение элемента, который уже был в массиве:

$favorite\_food = ['пюре', 'котлеты', 'борщ'];

keks\_log($favorite\_food[1]); // Выведет: "котлеты"

$favorite\_food[1] = 'блины';

keks\_log($favorite\_food[1]); // Выведет: "блины"

**Цикл while**

Цикл — конструкция, которая позволяет выполнить один и тот же код больше одного раза. Циклы отлично работают с массивами.

Синтаксис цикла while похож на синтаксис условной конструкции — он состоит из *имени цикла*, *условия цикла* и *тела цикла*. Действия, указанные в теле цикла, будут выполняться снова и снова, пока условие не станет ложным.

while (условие цикла) {

тело цикла

}

$months = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12];

Чтобы вывести в консоль все значения из этого массива, используем цикл while:

$index = 0; // Создание счётчика

while ($index < 12) {

keks.log($months[$index]); // Вывод элемента в консоль

$index = $index + 1; // Увеличение счётчика

}

Каждое выполнение блока кода в теле цикла называется **итерацией**. Количество итераций должно быть конечным, иначе цикл будет исполняться бесконечное число раз, и страница может загружаться бесконечно долго.

Чтобы ограничить количество итераций, мы создали переменную-счётчик $index. С каждой итерацией число в $index будет увеличиваться на единицу. В итоге в консоль по очереди выведутся элементы массива $months — числа от 1 до 12. Когда значение счётчика достигнет 12, условие станет ложным и цикл прекратит свою работу.

**Цикл while в шаблоне**

Чтобы внедрить цикл while в шаблон, нужно обозначить начало и конец цикла. Затем записать условие и в теле цикла задать увеличение индекса элемента с каждой итерацией и действия с разметкой.

<?php while (условие): ?>

<li><?= $массив[$index] ?></li>

<?php $index = $index + 1 ?>

<?php endwhile; ?>

**Команда count**

Команда count позволяет посчитать количество элементов в массиве, или *длину массива*. Чтобы это сделать, нужно после count написать круглые скобки, а внутри скобок — название массива. А чтобы иметь возможность дальше работать с полученным значением, можно сохранить его в переменную.

$one\_to\_five = [1, 2, 3, 4, 5];

$number = count($one\_to\_five);

keks\_log($number); // Выведет: 5

Нужно помнить, что команда count считает именно количество элементов, а не их индексы.

**Ассоциативный массив**

Массив, у которого вместо индексов ключи, называется **ассоциативным**. Каждый ключ хранит какое-то значение, как переменная. Имя ключа пишется в одинарных кавычках, а чтобы задать ключу значение, используются символы =>.

$spiderman = [

'name' => 'Питер', // Ключ 'name', значение 'Питер'

'surname' => 'Паркер' // Ключ 'surname', значение 'Паркер'

];

Оба элемента массива из примера относятся к Человеку-пауку, поэтому удобно хранить их не по отдельности в переменных, а вместе, в одном массиве. Чтобы получить значение из такого массива, нужно написать название массива, а затем в квадратных скобках указать ключ.

keks\_log($spiderman['name']); // Выведет: "Питер"

keks\_log($spiderman['surname']); // Выведет: "Паркер"

**Вложенные массивы**

Массивы могут содержать в себе другие массивы. Такие массивы называются вложенными.

$flowers = [

0 => [

'name' => 'Ромашка',

'cost' => 'free'

],

1 => [

'name' => 'Лилия',

'cost' => 300

]

];

**Цикл foreach**

Синтаксис цикла:

foreach ($массив as $переменная) {

тело цикла

}

На каждой итерации значением переменной становится очередной элемент массива. Элементы перебираются по порядку с первого до последнего. Поэтому нам не нужны индексы, достаточно обратиться в теле цикла к переменной.

**Цикл foreach в шаблоне**

Чтобы добавить такой цикл в разметку, нужно записать его начало и конец, а в тело цикла поместить действия с разметкой.

$fruits = ['Апельсин', 'Яблоко', 'Банан'];

<?php foreach ($fruits as $fruit): ?>

<li><?= $fruit ?></li>

<?php endforeach; ?>

// На странице появится список фруктов

**Условия в цикле**

Условия можно писать внутри циклов. В таком случае на каждой итерации цикла будет проверяться, истинно ли условие или ложно. Если истинно — действия из тела условия выполнятся.

$items = ['Стол', 'Окно', 'Кровать'];

<ul class='products-list'>

<?php foreach($items as $item): ?>

<?php if ($item === 'Стол'): ?>

// Действия с разметкой

<?php endif; ?>

<?php endforeach; ?>

</ul>

Подробнее об условиях в части «[Условия в PHP](https://htmlacademy.ru/courses/339)».

**Оператор сравнения «строгое равенство»**

В условиях часто используется оператор сравнения ===. Он называется **строгое равенство** и проверяет, равны ли значения слева и справа. Если равны, то условие истинно.

$item === 'Стол'