**Создаём пагинацию в каталоге**

На каждой странице должно быть не больше шести товаров. Разработчики называют постраничный вывод информации *пагинацией*.

Товары хранятся в массиве ***$products***, они все сейчас отображаются на странице каталога. В переменной ***$limit*** записано, сколько товаров нужно показывать на странице.

В переменной ***$offset*** («смещение») хранится порядковый номер первого товара на текущей странице. Если выводить по шесть товаров, то для первой страницы смещение будет 0, для второй — 6, для третьей — 12 и так далее.

Чтобы показать на странице лишь шесть товаров, используем встроенную функцию ***array\_slice***. Эта функция копирует часть исходного массива и возвращает её в виде нового массива.

Функция **array\_slice** принимает четыре аргумента. Чтобы получить товары для одной страницы, передадим функции следующие значения:

* исходный массив — ***$products***,
* первый элемент — ***$offset***,
* количество элементов — ***$limit***,
* сохранять ли ключи — ***true***, потому что ключ — это идентификатор товара, он нужен, чтобы показать правильную информацию о товаре.

Для первой страницы смещение ***$offset*** равно нулю, поэтому функция вернёт массив с первыми шестью товарами. Выведем полученный массив на страницу. После этого изменим значение переменной ***$offset***, чтобы посмотреть, как выглядят разные страницы каталога.

***Задание 1***

1. На 7 строке объявите переменную ***$products\_on\_page*** и запишите в неё массив с товарами одной страницы ***array\_slice($products, $offset, $limit, true).***
2. В файле components/product\_list.php на 5 строке замените ***$products на $products\_on\_page***.  
   Обратите внимание, в браузере на странице остались только первые шесть товаров.
3. В файле catalog.php на 6 строке измените значение переменной ***$offset*** на 6.  
   Обратите внимание, на странице отобразились товары с 6 по 11.

# Считаем смещение

Мы использовали функцию ***array\_slice,*** чтобы вывести на страницу лишь шесть товаров. После этого вручную изменили значение переменной ***$offset***, имитируя передвижение покупателя по страницам каталога. Но нам нужно, чтобы смещение считалось автоматически, в зависимости от страницы, на которой находится покупатель.

Номер страницы указан в параметере запроса ***page*** в адресной строке браузера.

Получим значение этого параметра с помощью ***$\_GET*** и сохраним его в переменной.

$page = $\_GET['page'];

Используем значение параметра, чтобы посчитать смещение для любой страницы каталога. Формула выглядит так:

$offset = ($page - 1) \* $limit;

Смещение начинается с нуля, а отсчёт страниц — с единицы. Чтобы не пропустить первые шесть товаров, сначала вычтем единицу из номера страницы. Полученное число умножим на количество товаров на странице

***Задание 2***

1. На 6 строке объявите переменную ***$page*** и присвойте ей значение $\_GET['page'].
2. На следующей строке измените значение переменной ***$offset*** на ($page - 1) \* $limit.
3. В браузере в адресной строке измените значение параметра page на 2.  
   Обратите внимание, на странице отобразились товары с 6 по 11.

# Функция intval

Мы посчитали смещение для любой страницы каталога и попробовали менять параметр ***page***. Если в адресной строке указать число, то всё работает, но если значение не указать или указать неправильно, то на странице выводятся шесть последних товаров. Почему так происходит?

Причина в этой команде:

$offset = ($page - 1) \* $limit;

Вычесть число можно только из другого числа. Если в переменной ***$page*** оказывается нечисловое значение, PHP пытается преобразовать его в число. А если не выходит, **подставляет на его место ноль**.

Если в каталоге в адресной строке указать неправильное значение, то вместо номера страницы в переменную ***$page*** подставится ноль. В результате смещение окажется равным -6. Это значение мы передаём вторым аргументом в функцию array\_slice:

$products\_on\_page = array\_slice($products, $offset, $limit, true);

Если второй аргумент функции — отрицательное число, отсчёт элементов начинается с конца массива: -1 — последний элемент, -2 — второй элемент с конца и так далее. Мы передаём функции -6, то есть просим начать с шестого элемента с конца. В итоге на страницу выводятся последние шесть товаров.

Это может запутать покупателей. Если адрес указан неверно, должна открываться первая страница каталога. Для этого сначала проверим, можно ли получить из параметра ***page*** число. Если можно, используем его. А если нельзя, используем число 1.

Получить число из параметра page можно с помощью встроенной функции intval. Функция принимает любое значение и всегда возвращает число. Если передать ей обычное число, она вернёт его же. Но если передать строку, функция попробует извлечь из неё число. Получить из строки число можно, если оно идёт в самом начале. Если числа в строке нет или если оно идёт не первым, функция вернёт 0.

Передадим функции intval значение из параметра page и посмотрим, что она вернёт.

***Задание 3***

1. На 6 строке преобразуйте значение переменной $page в число: intval($\_GET['page'])
2. На 10 строке с помощью команды echo выведите переменную $page  
   Обратите внимание, вывелось "1".
3. В браузере в адресной строке замените значение параметра page на keks. Обратите внимание, в консоль вывелось "0".
4. Удалите значение параметра page в адресной строке. Обратите внимание, в консоль по-прежнему выводится "0".

# Сокращённый тернарный оператор ?:

Мы преобразовали значение из адресной строки в число, используя функцию intval. Если функция вернула ноль, то адрес указан неверно. Чтобы в этом случае открывалась первая страница каталога, нужно указать переменной ***$page*** значение по умолчанию.

Используем сокращённый тернарный оператор — ***?:***. Он обозначается знаком вопроса и двоеточием:

$foo = значение-1 ?: значение-2

Оператор работает так: если значение-1 истинно, то в переменную запишется оно. А если ложно, то запишется значение-2.

Чтобы открывать по умолчанию первую страницу каталога, укажем переменной $page второе возможное значение:

$page = intval($\_GET['page']) ?: 1;

Если из параметра в адресной строке нельзя извлечь число, функция intval вернёт ноль.

// Функция intval вернула 0

$page = 0 ?: 1;

В PHP ноль считается ложным. Так как первое значение ложное, в переменную запишется второе значение — число 1.

Ложными в PHP также считаются строки '' и '0', булево значение false и массивы без элементов. Остальные числа, строки и массивы — истинные.

Используем сокращённый тернарный оператор, чтобы указать второе значение для переменной $page. После этого убедимся, что первая страница каталога открывается даже тогда, когда указано неверное значение параметра.

***Задание 4***

1. В браузере в адресной строке удалите значение параметра page. Обратите внимание, на странице отобразились последние шесть товаров.
2. На 6 строке используйте сокращённый тернарный оператор, чтобы добавить переменной $page значение по умолчанию. Замените значение переменной на intval($\_GET['page']) ?: 1  
   Обратите внимание, теперь на странице отобразились первые шесть товаров.
3. В браузере в адресной строке задайте параметру page значение keks. Обратите внимание, на странице по-прежнему отображаются первые шесть товаров.

# Цикл for

Мы позаботились, чтобы в каталоге по умолчанию открывалась первая страница. Теперь ничего не сломается, даже если забыть про параметр ***page*** в адресной строке. Но можно сделать пагинацию ещё удобнее, добавив в шаблон ссылки на все страницы каталога.

Ссылок на страницы будет несколько, поэтому используем цикл. Он пригодится нам, чтобы добавить в шаблон [ссылки](https://htmlacademy.ru/courses/305/run/1) на разные страницы каталога.

Ссылки будут элементами [упорядоченного списка](https://htmlacademy.ru/courses/301/run/3). Создадим их с помощью цикла for.

Чтобы внедрить цикл в шаблон, используем такой синтаксис:

PHP-теги пишем полностью — <?php ?>;

вместо фигурных скобок ставим двоеточие — for ():;

конец цикла обозначаем командой endfor, после которой идёт точка с запятой.

<ol>

<?php for ($i = 1; $i <= 9; $i = $i + 1): ?> // Начало цикла

<li>...</li> // Тело цикла

<?php endfor; ?> // Конец цикла

</ol>

Код в примере создаст 9 элементов списка.

Нам нужно, чтобы в каждом элементе списка была ссылка на отдельную страницу каталога. Текстом ссылки будут числа — номера страниц. Для этого выведем переменную-счётчик внутри тега <a>:

<li><a><?= $i ?></a></li>

Предположим, что у нас 5 страниц в каталоге. Используем цикл for, чтобы создать ссылки на них.

***Задание 5***

1. На 5 строке напишите начало цикла: <?php for ($i = 1; $i <= 5; $i = $i + 1): ?>
2. На 9 строке напишите конец цикла: <?php endfor; ?>
3. В теле цикла внутри тега <a> выведите переменную $i Используйте для вставки PHP-кода краткую запись.  
   Обратите внимание, в браузере появился список с пятью элементами.
4. В браузере кликните по ссылке на вторую страницу. Обратите внимание, ничего не произошло.

# Добавляем адрес в атрибут href

Мы создали список с ссылками, но ссылки пока никуда не ведут. Чтобы они заработали, нужно указать адрес страницы в [атрибуте](https://htmlacademy.ru/courses/305/run/1) href.

Адрес каждой страницы каталога состоит из адреса файла и параметра запроса. Они разделяются вопросительным знаком:

catalog.php?page=1

Адрес файла у каждой страницы одинаковый — catalog.php. Параметр запроса тоже один — page. Но вот значение этого параметра у разных страниц будет отличаться. Это значение совпадает со значением переменной-счётчика.

Используем переменную $i, чтобы указать в ссылке правильный адрес страницы:

<a href="catalog.php?page=<?= $i ?>"><?= $i ?></a>

Добавим значение атрибуту href, после чего убедимся, что ссылки на страницы каталога работают.

***Задание 6***

1. На 7 строке добавьте значение атрибуту href ссылки: catalog.php?page=<?= $i ?>.
2. В браузере кликните по ссылке на вторую страницу. Обратите внимание, открылась страница с шестью товарами.  
   Нажмите на кнопку ***← Вернуться*** обратно в браузере, чтобы вернуться на первую страницу каталога.
3. В браузере кликните по ссылке на пятую страницу. Обратите внимание, не вывелось ни одного товара.

# Считаем страницы каталога

Мы создали ссылки на страницы каталога, но ссылок получилось больше, чем нужно. Так вышло потому, что в цикле мы использовали произвольное число 5. Но ведь количество ссылок должно соответствовать числу страниц с товарами. Выясним, сколько в каталоге таких страниц.

Чтобы узнать, сколько в каталоге страниц с товарами, разделим общее число товаров на количество товаров на одной странице. Используем встроенную функцию count, чтобы узнать количество товаров в массиве $products $array\_quantity = count($array);

Разделим общее число товаров на переменную $limit, в которой храним количество товаров на одной странице. Запишем результат в переменную и используем её в шаблоне, чтобы создать правильное количество ссылок.

***Задание 7***

1. На 9 строке объявите переменную $products\_quantity и присвойте ей значение count($products).
2. На следующей строке объявите переменную $pages и присвойте ей значение $products\_quantity / $limit.
3. В файле components/products\_list на 5 строке замените число 5 на переменную $pages.  
   Обратите внимание, в браузере остались ссылки только на первые три страницы.
4. В адресной строке браузера измените значение параметра page на 4. Обратите внимание, на странице вывелись три последних товара.

# Функция ceil

Мы разделили общее число товаров в каталоге на количество товаров на одной странице и использовали результат, чтобы создать ссылки на страницы каталога. Ссылок получилось три, но на трёх страницах все товары не помещаются. Есть ещё четвёртая страница, но ссылка на неё не создалась.

Почему так произошло? Давайте посчитаем. Всего в каталоге 21 товар. На одной странице показывается шесть товаров. Если разделить 21 на 6, получим 3.5. Именно это число и попадает в переменную $pages.

В условии цикла мы указали $i <= $pages. Переменная-счётчик $i увеличивается на 1 на каждой итерации. Цикл выполняет три итерации, а на четвёртой значение счётчика оказывается 4. Это больше 3.5, поэтому код в теле цикла не выполняется, и четвёртая ссылка не создаётся.

Следует учесть, что число в $pages может быть дробным. Создавать ссылку нужно даже в том случае, если последняя страница заполнена не полностью. Для этого округлим число страниц вверх. Для такого округления в PHP есть [встроенная функция](https://www.php.net/manual/ru/function.ceil) ceil. Она принимает число и, если оно дробное, округляет его вверх до ближайшего целого.

Используем функцию ceil, чтобы получить правильное число страниц. Подставим это число в цикл в шаблоне и создадим ссылки на все страницы с товарами.

***Задание 8***

1. На 11 строке объявите переменную $pages\_total и присвойте ей значение ceil($pages)
2. В файле components/products\_list на 5 строке замените переменную $pages на $pages\_total  
   Обратите внимание, в браузере появилась ссылка на четвёртую страницу.
3. В браузере кликните по ссылке на четвёртую страницу. Обратите внимание, на странице отобразились три последних товара.

# Тернарный оператор ?:

Мы создали ссылки на все страницы с товарами, теперь покупателям доступен весь ассортимент магазина. Но как им понять, на какой странице каталога они находятся? Выделим ссылку на текущую страницу, добавив ей специальный класс.

Чтобы вычислить, какая ссылка ведёт на текущую страницу, используем тернарный оператор. Мы уже работали с его [краткой записью](https://htmlacademy.ru/courses/357/run/4). Полная запись позволяет явно указать условие:

условие ? значение-1 : значение-2;

Если условие истинно, то PHP использует значение-1, а если ложно — значение-2. После условия ставится вопросительный знак, а значения разделяются двоеточием.

Тернарный оператор работает почти так же, как [условие](https://htmlacademy.ru/courses/339/run/2) if. Когда использовать оператор, а когда — условие? Если нужно выбрать между двумя значениями, проще использовать тернарный оператор. Но если это не просто значения, а, например, несколько строчек кода, то лучше использовать if.

С помощью тернарного оператора удобно переключать классы в шаблоне:

<p class="<?= get\_product\_is\_new($id) ? 'new' : 'old' ?>">...</p>

Если функция в условии вернёт true, то абзацу добавится класс new. А если функция вернёт false, то добавится класс old.

В каталоге интернет-магазина нужно добавить класс current ссылке на текущую страницу. Ссылкам на другие страницы ничего добавлять не нужно. Для обозначения пустого значения используем пустую строку — ''.

условие ? 'current' : '';

Потренируемся добавлять и убирать класс у ссылок с помощью тернарного оператора. Для этого сначала используем в условии булево значение true, а после заменим его на false.

***Задание 9***

1. На 7 строке добавьте значение атрибуту class, используя тернарный оператор: <?= true ? 'current' : '' ?>  
   Обратите внимание, в мини-браузере выделились все ссылки на страницы.
2. Измените условие в тернарном операторе на false Обратите внимание, у всех ссылок пропало выделение.

# Выделяем ссылку на текущую страницу

Мы посмотрели, как работает тернарный оператор. Булево значение true всегда истинно, поэтому класс добавился всем ссылкам. Значение false всегда ложно, поэтому, когда мы указали его, выделение у ссылок пропало.

Нам нужно добавить класс только одной ссылке — ссылке на текущую страницу. Текущая страница — это та, у которой значение счётчика $i совпадает со значением параметра в адресной строке.

Используем [строгое равенство](https://htmlacademy.ru/courses/341/run/14), чтобы сравнить два значения:

$i === $page;

Доработаем пагинацию в каталоге. Добавим класс current ссылке на текущую страницу. После этого убедимся, что листать каталог можно как из адресной строки, так и кликая по ссылкам.

***Задание 10***

1. На 7 строке замените false в условии на $i === $page  
   Обратите внимание, в браузере выделилась ссылка на первую страницу.
2. В браузере в адресной строке измените значение параметра page на 2  
   Обратите внимание, теперь выделилась ссылка на вторую страницу.
3. В браузере кликните по ссылке на третью страницу.  
   Обратите внимание, теперь выделилась ссылка на третью страницу.

# Функция date

Добавим на страницу яркий баннер.

Дата, которая выводится внутри баннера, должна быть сегодняшней. Можно изменять шаблон каждый день ровно в полночь… Но лучше использовать встроенную функцию date.

Эта функция принимает строку-шаблон и возвращает дату в указанном формате.

$now = date('H:i:s d.m.Y');

Выведем с помощью переменной $today дату внутри баннера. Посмотрим, в каком формате она выведется, и, если нужно, изменим его.

***Задание 11***

1. В файле catalog.php подлючите файл products\_list11.php вместо products\_list.php
2. В файле products\_list11.php на 2 строке замените 01.01.1970 на переменную $today Используйте для вставки PHP-кода краткую запись.  
   Обратите внимание, в браузере вывелась сегодняшняя дата.
3. В файле catalog.php на 13 строке замените 'j F y' на 'd.m.Y'  
   Обратите внимание, дата на странице теперь выводится в другом формате.

# Меняем формат даты

Можно использовать функцию date, чтобы вывести день и год, но получить с её помощью название месяца в правильном падеже не получится. Разобьём дату на три части и для каждой используем свою переменную.

День нужен в виде числа без ведущего нуля. За это в функции date отвечает символ j:

$day = date('j');

Месяц нужно вывести словом. создайте массив с названиями месяцев в правильном падеже. С помощью функции date и символа n получим номер месяца, а затем используем его как ключ, чтобы вывести подходящее слово из массива:

$months = [1 => 'января', 2 => 'февраля', 3 => 'марта', 4 => 'апреля', 5 => 'мая', 6 => 'июня', 7 => 'июля', 8 => 'августа', 9 => 'сентября', 10 => 'октября', 11 => 'ноября', 12 => 'декабря'];

$month = date('n');

echo($month); // Выведет: "1"

echo ($months[$month]); // Выведет: "января"

Чтобы получить год в виде четырёхзначного числа, используем символ Y:

$year = date('Y');

Создадим переменные для дня, месяца и года и запишем в них значения, которые вернёт функция date. После этого объединим эти значения в переменной $today с помощью [конкатенации](https://htmlacademy.ru/courses/339/run/8):

$a = 'Слона зовут ';

$b = 'Думпо';

echo ($a . $b); // Выведет: "Слона зовут Думпо"

***Задание 12***

1. На 15 строке объявите переменную $day и присвойте ей значение date('j')
2. На следующей строке объявите переменную $month и присвойте ей значение date('n')
3. На следующей строке объявите переменную $year и присвойте ей значение date('Y')
4. Измените значение переменной $today на $day . $months[$month] . $year  
   Обратите внимание, в браузере дата вывелась в правильном формате, но все значения «слиплись».

# Используем двойные кавычки

Мы получили дату в нужном формате, но значения, которые мы использовали, «слиплись». Это произошло потому, что мы, составляя строку с датой, забыли добавить пробелы между значениями. Чтобы пробелы появились, нужно написать так:

$today = $day . ' ' . $months[$month] . ' ' . $year;

Такую запись сложно понять. И очень легко ошибиться, составляя её. К счастью, есть иной способ получить нужную строку.

В PHP для строк используют как одинарные ', так и двойные " кавычки. В большинстве случаев они работают одинаково, но не всегда. Например, если использовать переменную внутри одинарных кавычек, то выведется её имя. А если внутри двойных, то значение:

$name = 'Думпо';

echo ('Слона зовут $name'); // Выведет: "Слона зовут $name"

echo ("Слона зовут $name"); // Выведет: "Слона зовут Думпо"

Посмотрим, как можно вставить пробелы между переменными. Выведем в консоль строку Слона зовут Думпо:

$a = 'Слона';

$b = 'зовут'

$c = 'Думпо';

// Конкатенация

echo ($a . ' ' . $b . ' ' . $c);

// Двойные кавычки

echo ("$a $b $c");

Двойные кавычки помогают писать более понятный код и избежать запутанной конкатенации. Используем их, чтобы показать на странице дату вместе с пробелами.

***Задание 12***

1. На 19 строке измените значение переменной $today на "$day $months[$month] $year"  
   Обратите внимание, теперь дата в браузере выводится с пробелами.