**Язык PHP**

**PHP** – это один из самых распространённых языков сценариев с широкой сферой применения и opensource-кодом. Если упростить определение, это наиболее популярный веб язык программирования для создания веб-приложений.

**PHP** – это аббревиатура от «Hypertext Preprocessor». Корни синтаксиса уходят глубоко в C, [Java](https://itproger.com/course/java) и Perl. Язык довольно легко даётся в изучение. Главным его преимуществом является быстрый и простой доступ к генерированию веб-страниц. Ещё одно важное достоинство PHP в сравнении с Perl или C – это блоки PHP-кода, которые можно внедрять прямо в документ с HTML.

Язык PHP впервые появился в 1995 году и с тех пор является одним из лидером на рынке веб технологий. PHP является серверным языком программирования, что используется зачастую в веб проектах для обеспечения серверного функционала.

Основным преимуществом языка является его простота изучения, а также легкость применения. На основе PHP реально создавать высокопроизводительные и нагруженные веб-приложения в сжатые сроки. Готовые продукты несложно редактировать, улучшать и поддерживать.

Отличительная черта языка заключается в том, что он обрабатывается на стороне сервера. При помощи нехитрых действий можно создать такой код, что автоматически создаёт страницу, ничем не отличающуюся от чистого HTML-кода. Пользователь не сможет догадаться что перед ним обычный HTML или результат работы PHP.

PHP один из тех языков, с которых можно начинать путь программиста. Несмотря на лёгкость освоения, язык очень функционален, он готов выполнять даже сложные задачи. Даже если вы не знаете ничего о PHP и программировании в целом, особого труда с освоением возникнуть не должно. Нет сомнений, что уже спустя пару часов изучения PHP вы научитесь создавать первые скрипты.

PHP относится к тем языкам, что регулярно улучшаются и развиваются. Он точно будет занимать лидирующие места в сфере веба ещё достаточно долго. Ближайшие 10-15 лет программисты на PHP будут востребованы.

**Что значит «серверный язык»?**

Язык PHP ценят за мощность и гибкость. Он выступает связующим звеном между клиентом и базой данных, а также выполняет базовую обработку всех входных параметров от пользователя. Всё это стало возможным благодаря работе скриптов на стороне сервера.

Серверные языки умеют работать с базами данных, сессиями, обработкой форм, получением информации о пользователе: его операционная система, браузер и т. д. Сегодня создание современного сайта без использования серверного функционала не является чем-то разумным. Практически любой веб сайт требует наличие базы данных, к примеру, для хранения информации о пользователях. Работать с БД на основе лишь языка JavaScript было бы невозможно, ведь JS по своей сути является клиентским языком (в расчет не берем платформу Node JS).

**Практическая сторона вопроса:**

Удобно создает страницы или их фрагменты, которые одинаково отображаются у каждого пользователя;

PHP все равно какая производительность компьютера у пользователя или какая операционная система стоит. Вся обработка данных происходит на сервере;

Для получения новых данных на странице обязательно потребуется её обновление (если не прибегать к JS). До обновления страницы стандартными средствами PHP нельзя что-либо менять на ней. Иначе говоря, язык не интерактивный. С помощью Ajax, уже реально обновлять данные веб-страницы без её перезагрузки.

**Советы начинающим программистам**

Как вы уже поняли, PHP применяется для разработки сайтов. В создании веба участвуют и другие языки, ключевые из них [HTML](https://itproger.com/course/html) и [CSS](https://itproger.com/course/css). Исключительно с помощью PHP не создать сайт. HTML – это структура страницы, а CSS – её стили (цвета, расположение, эффекты и т. п.).

Важной частью работы с PHP является взаимодействие с базой данных (БД). В ней хранятся базовые данные о пользователе, статьи, заметки и различные другие данные. Добавление, изменение и извлечение данных с БД – это те задачи, с которыми постоянно сталкивается PHP-программист. Именно поэтому должное внимание стоит уделить изучению языка запросов [SQL](https://itproger.com/course/sql).

У нас на сайте уже есть курсы по изучению [JavaScript](https://itproger.com/course/javascript), [HTML](https://itproger.com/course/html) и [CSS](https://itproger.com/course/css), которые помогут преодолеть пропасть между начинающим и профессиональным программистом.

**Хейт и будущее языка ПХП**

Возможно, перед изучением PHP вы уже слышали плохие отзывы относительно языка. Они связаны с тем, что PHP хоть и хороший язык, но не универсальный. К примеру, поддержка крупных, высоконагруженных веб-сайтов на PHP осуществляется хуже, нежели на том же Node JS.  
  
Но не спешите закрывать страницу. Да, PHP в некоторой степени не идеальный язык, но все же он явный лидер среди серверных языков программирования. Сегодня PHP сохраняет актуальность, при чём такая тенденция будет сохранятся и далее. Хоть и существуют достойные альтернативы языку, но согласно[статистике w3techs](https://w3techs.com/technologies/details/pl-php/all/all" \t "_blank), 75% сайтов применяют именно PHP в качестве основного серверного языка. Это значит, что разработчику на PHP всегда найдётся работа с конкурентной оплатой.

В ближайшие годы и даже десятилетия язык никуда не исчезнет и по прежнему будет стоящим для изучения языком. К тому же, язык постоянно обновляется и, возможно, вскоре он заберет еще больший процент на рынке.

По большей степени его хейтят из-за того, что он слишком большой и есть огромное сообщество. В нем находятся как почитатели, так и хейтеры. В любом случае, начав изучение PHP вы не прогадаете.

Вот вам небольшой список сайтов, что написаны на PHP. Среди них:

* сайт Facebook:
* социальная сеть Вконтакте;
* библиотека Wikipedia;
* движок Wordpress;
* сеть Tumblr.

**Локальный сервер. Вывод информации и комментарии**

Разработка проектов на PHP требует наличие сервера. Сервер может быть двух видов:

удаленный сервер (хостинг);

локальный сервер.

**Удаленный сервер** – это сервер, что находится на неком удаленном ПК. Доступ к такому серверу имеете вы и все участники сети Интернет. При публикации сайта на удаленный сервер вы выгружаете его для всеобщего обозрение. Все смогут перейти к вам на сайт. Такая услуга выгрузки зачастую платная и приобретается у разных хостинг провайдеров.

PS: мы для сайта используем [данного хостинг провайдера](https://www.ukraine.com.ua/?page=132576" \t "_blank).

Локальный сервер – это сервер, что устанавливается на ваш компьютер. Такой сервер доступен только вам. Его зачастую используют в целях разработки проекта. Именно такой сервер будем использовать мы в ходе всего курса.

Для его установки нужна специальная программа. Вы можете использовать любую удобную вам. Ниже несколько из них:

* Программа [MAMP](https://www.mamp.info/en/" \t "_blank). Удобная и быстрая, но ее работа может быть не точной на ОС Виндовс. Рекомендуем для установки на Мак ОС;
* Программа [OpenServer](https://ospanel.io/" \t "_blank). Схожая программа, что идеально работает на ОС Виндовс;
* Программа [XAMPP](https://www.apachefriends.org/ru/download.html" \t "_blank). Работает на всех ОС, но имеет немного сложный интерфейс. Рекомендуем к установке на ОС Linux.

 Детальную документацию по установке и работе с OpenServer вы можете найти на их [официальном сайте](https://ospanel.io/docs/" \t "_blank).

Если вы пользователь компьютера Мак, то лучшим локальным сервером будет программа [MAMP](https://www.mamp.info/en/" \t "_blank).

Вывод информации

В PHP для вывода информации на экран используется оператор echo. Он позволяет выводить как простой текст, так и HTML-теги.

Пример использования:

// Одинарные кавычки принято использовать для текста

echo 'Просто текст';

// Если нужно вывести дополнительно HTML-теги,

// то используются двойные кавычки

echo "Обычный текст, но с <b>HTML</b>";

Стоит также отметить, что с появлением новых версий PHP вывод тегов стал возможен во всех типах кавычек. Через оператор echo вы можете выводить большие куски HTML кода.

Комментарии важны в любом языке программирования. Они позволяют описать какой-либо блок кода, чтобы в дальнейшем у вас не возникало никаких непоняток. Комментирование очень важно на первой стадии изучения, так как вы ещё не умеете писать красивый и читабельный код, поэтому стоит оставлять подсказки хотя бы для самого себя.

**Комментарии**

Язык PHP поддерживает три типа комментариев:

// Два слэша для комментирования одной строки

/\*

В такой конструкции вы

можете записывать множество

строчек комментариев

\*/

# Решетка служит для комментирования одной строки

Используя комментарии вы сможете добавлять описание к вашей программе и всегда быть в курсе всех участков кода в приложении.

**Исходной код: Выполнить задание**

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>PHP</title>

</head>

<body>

<?php

echo '<b>Hello</b> World!';

echo "<br>Hello";

echo '\'';

// Comment

# Comment

/\*

Comment

Comment

\*/

?>

</body>

</html>

**Переменные и типы данных в PHP**

Переменные присутствуют во всех языках программирования. PHP не является исключением. В ходе урока мы научимся создавать переменные и работать с константами. Также мы изучим типы данных для переменных.

Переменные невероятно важны, так как позволяют хранить информацию и использовать её в дальнейшем. Вначале может быть не совсем понятно зачем вообще что-то записывать в переменную, если можно просто оперировать значениями без них. Понимание переменных придет немного позже, когда мы начнем создавать более сложные программы и нам потребуется хранить информацию в каком-либо месте.

Типы переменных в языке **PHP** не объявляются очевидно, тем не менее они присутствуют. Интерпретатор понимает что записывается в переменную и на основании этого добавляет тип к этой переменной.

В ходе программы мы можем перезаписывать значение переменной, при этом мы можем указывать новый тип переменной.

К примеру, изначально был записан тип **float** (число с точкой), но потом можно записать другой тип данных:

$first\_num = 23.2; // Тип данных float

$first\_num = "1"; // Тип данных string

Ниже приводим несколько существующих типов данных в языке PHP:

$some = 1 Integer - целые числа;

$some = 1.12 Float - числа с плавающей точкой;

$some = "Привет" String - строки;

$some = true Boolean - принимают только true или же false.

Создание констант

В PHP помимо создания простых переменных можно создавать константы.

Константы – это значения, что нельзя изменить входе выполнения программы. Создав один раз константу и присвоив ей значение, мы не сможем больше изменить это значение, так как будет выдаваться ошибка.

Чтобы создать константу необходимо использовать метод define():

define("SOME", "Значение");

echo SOME; // Вывод константы

Первым параметром указывается название константы, а вторым параметром - значение. Принято записывать названия констант в верхнем регистре, хотя при написании в нижнем регистре ошибок выдано не будет.

Вы можете проверить существует ли константа использовав метод defined().

**Исходной код: Выполнить задание**

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>PHP</title>

</head>

<body>

<?php

define(MY\_AGE, 24);

echo MY\_AGE . '<br>';

$number = 5; // int

// $number = 45;

$num = -0.55; // float

$str = "Переменная"; // string

$bool = false; // boolean

$a = 0.5;

$b = "0.5";

echo $a + floatval($b);

echo $str . ': ' . $number . '. Var 2: ' . $num;

?>

</body>

</html>

**Задание:**

**1.Работа с переменными**

Создайте переменные для хранения:

* имени и фамилии;
* даты рождения;
* возраста;
* уровня радости от 0 до 1.

После установки значений выведите все переменные на экран.

Для имени и фамилии необходимо использовать строковый тип данных:

$name = “Павел”;

$surname = “Иванов”;

Чтобы вывести дату рождения вы можете использовать строковую переменную или три числовых переменных:

$date = 1;

$month = 11;

$year = 1999;

Количество лет и уровень радости можно записать следующим образом:

$age = 20;

$happy = 0.76;

Теперь осталось лишь вывести все данные:

echo “Имя: $name, фамилия: $surname”;

echo “Дата рождения: $date $month $year”;

echo “Возраст: $age”;

echo “Уровень счастья: $happy”;

**2. Необычные переменные**

Создайте переменные со значениями: false, -0.78902.

Выведите все значения в консоль.

Скрыть

Решение задания:

$hasCar = false;

$number = -0.78902;

// Вывод в консоль

echo "Есть машина? $hasCar";

echo "Число - $number";

**3.Простая константа**

Создайте константу с название NUMBER\_E и со значением 2,71828. Выведите константу на экран.

Скрыть

Решение задания:

define("NUMBER\_E", 2.71828);

echo NUMBER\_E;

**4.Проверка данных**

Проверьте существует ли константа NUMBER\_E. Результат проверки выведите на экран.

Скрыть

Решение задания:

echo defined("NUMBER\_E");

# **Математические действия. Встроенные функции**

Выполнение математических действий важная составляющая любого языка. За урок мы научимся выполнять как базовые математически операции, так и более сложные математические действия на основе языка PHP.

В **PHP** можно выполнять все математические операции. Это такие операции, как:

* вычитание - $z = $y - $x;
* сложение - $z = $y + $x;
* умножение - $z = $y \* $x;
* деление - $z = $y / $x (на ноль делить нельзя);
* остаток при делении - $z = $y % $x;

Если операция проходит над одной и той же переменной, то можно запись сократить:

$z = $z + 20;

$z += 20; // Эта запись аналогична той, что находиться выше

Если необходимо увеличить или уменьшить число на 1, то можно прописать **++** и **--** соответсвенно:

$z = 10;

$z += 1;

$z++; // Аналогично тому, что прописано выше

## Математические функции

**PHP** имеет большой функционал для работы с математическими операциями. Для их использования есть множество встроенных функций, которые выполняют различные математические действия.

Мы можем получить значения математических чисел (число Пи, число Е) или же воспользоваться функциями для нахождения конкретных значений (синус, косинус, квадратный корень и так далее).

**Исходной код Выполнить задание**

**<?php**

$x = 10;

$y = 20;

echo $x + $y.'<br>';

echo $x - $y.'<br>';

echo $x \* $y.'<br>';

echo $x / $y.'<br>';

echo $x % $y.'<br>';

$x += 10; // $x = $x + 10;

$y -= 10; // $y = $y - 10;

echo $x.'<br>';

echo $y.'<br>';

$x++; // $x += 1;

$x--; // $x -= 1;

echo M\_PI.'<br>';

echo M\_E.'<br>';

echo abs(-22).'<br>';

echo round(3.5556734, 2).'<br>';

$rand = mt\_rand(1, 20);

$rand++;

echo $rand.'<br>';

echo max(2, 5, 6, 1, 8, 3, 6).'<br>';

**?>**

**Простая математика**

Создайте две переменные со значениями 5.6 и 8.9.

Выведите в консоль все математические операции над переменными: сумма, разница, умножение, деление и остаток при делении.

Скрыть

Решение задания:

$first = 5.6;

$second = 8.9;

$summ = $first + $second;

$diff = $first - $second;

$mult = $first \* $second;

$divide = $first / $second;

$ostatok = $first % $second;

echo "Сумма: $summ";

echo "Разница: $diff";

echo "Умножение: $mult";

echo "Деление: $divide";

echo "Остаток: $ostatok";

**Операции над переменной**

Создайте переменную со значением -2.

Умножьте переменную на 3 и добавьте 1.

Все операции провидите при помощи сокращенных форм арифметических действий.

Решение задания:

$a = -2;

$a \*= 3;

$a++;

echo "Результат: $a";

**Наибольшее число**

Найдите самое большое число в следующем множестве:

9, 23, 0, -24.5, 34, 11, 41, 3, -3.2, 3

Решение задания:

echo "Максимум - " . max(9, 23, 0, -24.5, 34, 11, 41, 3, -3.2, 3);

**Немного математики**

Получите квадратный корень из числа 2 и округлите его к меньшему числу. Выведите результат в консоль.

Решение задания:

// Метод ceil округляет к большему

// Метод floor округляет к меньшему

echo floor(sqrt(2));

# **Строковые операции в PHP**

PHP обладает большим функционалом для работы со строками. В ходе урока мы научимся работать со строками и встроенными функциями для работы с ними.

Мы можем выводить на экран абсолютно все символы через оператор echo. В строках можно выводить как простой текст, так и **HTML-теги**.

Для вывода слэша, кавычек или пропуска строки используйте дополнительный слэш - \\, \", \', \n.

Для объединения строк используйте точку:

echo "Some" . " text";

Каждая строка представляет из себя массив символов. Зная это вы можете работать с любой строкой как с обычным массивом.

Пример работы со строкой:

$string = "Строка";

echo strlen($string); // Выведет 6, так как 6 символов

echo trim($string . " плюс пробелы ");

// Удалит все лишние пробелы

Также мы можем искать символы в строке или же возводить символы в верхний (strtoupper) или нижний регистр (strtolower).

**Исходной код Выполнить задание**

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>PHP</title>

</head>

<body>

**<?php**

$str = "Hello";

echo "VAR: $str" . '!';

echo "<input type=\"text\"><br>";

$length = strlen($str);

echo strtoupper(trim(" some "));

echo $length;

echo md5("qwerty");

**?>**

</body>

</html>

**Вывод информации**

Выведите на экран следующие символы: двойная кавычка, одинарная кавычка, слэш, пропуск строки, а также тег div.

Решение задания:

echo "\"";

echo " ' ";

echo "\\";

echo "\n";

echo "<div>Просто блок</div>";

**Объединение строк**

Выполните предыдущее задание, но все строки объедините в одну строку используя специальный оператор для объедения строк.

Решение задания:

echo "\"" . " ' " . "\\" . "\n" . "<div>Просто блок</div>";

**Работа со строкой**

У вас есть строка "ASDПросто текстASD". Сделайте с ним следующие операции:

Удалите три последних символа в строке;

Сделайте первые три символа в нижнем регистре "asdПросто текст";

Выведите в консоль лишь первые три символа.

Решение задания:

$string = "ASDПросто текстASD";

// Первое задание

$newStr = substr($string, 0, strlen($string) - 3); // Обрезаем все символы, кроме 3 последних

echo "Обрезанная строка - $newStr";

// Второе задание

$lowerCase = substr($newStr, 0, 3);

// Обрезанную часть в нижний регистр + добавляем прежнюю строку, но без первых трех элементов

$result = strtolower($lowerCase);

$result .= substr($newStr, 3, strlen($newStr));

echo "Нижний регистр - $result";

// Третье задание

echo "Первые три символа - " . substr($result, 0, 3);

# **Условные конструкции (if-else). Логические операторы**

Условные конструкции позволяют проверить некое выражение и в зависимости от его результата выполнить необходимый код. В уроке мы познакомимся с конструкцией «if - else» и изучим все тонкости данной конструкции.

Логические операции используются для проверки данных. Их часто применяют в условных операторах, которые мы будет изучать в дальнейшем. Каждая логическая операция возвращает либо 0 (**false**), либо 1 (**true**).

В логических операциях есть следующие операнды:

$a == $b - равенство (если а равно b);

$a != $b - неравенство (если а **не** равно b);

$a > $b - больше (если а больше b);

$a < $b - меньше (если а меньше b);

$a >= $b - больше или равно (если а больше или равно b);

$a <= $b - меньше или равно (если а меньше или равно b);

!$a - ложь (если a равно false);

$a - истинна (если a равно true).

Если понадобиться проверить несколько условий в одной логической операции, то мы можем воспользоваться логическим "и" или же логическим "или":

// Если и первое, и второе условие верные,

// то результат будет true

$a != $b && $a > $b

// Если или первое, или второе условие верные,

// то результат будет true

$a < $b || $a == $b

Оператор эквивалентности

В **PHP** существует специальный оператор эквивалентности, который позволяет проверить на соответствие значений и на соответствие типов данных.

Для проверки на эквивалентность используйте тройной символ равенства - ===. В результате он проверить значение переменной, а также сравнит тип данных. Если оба значения будут совпадать, то результат будет **true**, иначе – **false**.

Условные конструкции

Условные операторы очень важны, ведь они позволяют выполнять разный код в зависимости от исхода условия. В языке PHP существует три конструкции позволяющих осуществлять проверку. Сейчас рассмотрим две из них.

Конструкция if - else

За счёт if else можно проверить одно или несколько условий и в случае их успешной проверки будет выполнен один, иначе другой.

Рассмотрим пример:

$a = 2;

$b = 10;

if ($a == $b) { // Если $a будет равным $b, тогда будет выполнен код

// Здесь код который будет выполнен

// Если все одна строка кода, то фигурные скобки {}

// можно не ставить

echo "$a == $b";

} else if ($a <= $b) { // Если $а будет меньшим или равным $b

// Если предыдущее условие не будет выполнено,

// а здесь условие окажется верным,

// то будет выполнен этот код

echo "$a <= $b";

} else {

// Этот код сработает, если другие условия не будут выполнены

echo "$a > $b";

}

Вы можете прописать структуру лишь с одним условием if, а можете дописать в неё сколько-угодно вложенных условий else if.

Внутри каждого из условий можно прописывать другие конструкции if else, что будут проверять новые выражения.

Если необходимо проверить несколько условий в одном операторе, то можно воспользоваться логическим «и» или же логическим «или»:

if ($a != $b && $a > $b) {

// Код будет выполнен, если и первое, и второе условие

// окажутся верными

}

if ($a < $b || $a == $b) {

// Код будет выполнен, если или первое, или второе условие

// окажется верным

}

Тернарная операция

Это сокращенная форма if else. Записывается следующим образом:

$x = 90;

$a = 8;

$res = $x < $a ? ($x + $a) : ($x - $a);

echo $res;

Мы проверяем, если "x" будет меньше за "a", то мы в переменную res устанавливаем их сумму, иначе - их разницу.

**Исходной код Выполнить задание**

**<?php**

$a = 5;

$str = "Hello";

$isWeatherGood = false;

if($str == "Hello" && !$isWeatherGood)

echo 'First';

else if($a == 5) {

$res = "Second";

echo $res;

if($str == "Hello") {

echo "<br>Yes";

}

}

else if($a > 50)

echo '$a > 50';

else if($a <= 45)

echo '$a <= 45';

else

echo 'Else';

**?>**

**Задание**

1. Простое условие

У вас есть переменная:

$a = 23;

Создайте условие if else, которое:

будет выводить число "10", если переменная равна 5;

будет выводить число "15", если переменная равна 10;

будет выводить число "20", если переменная равна 30;

если все проверки не подойдут, то необходимо вывести предложение: «Что-то не то!».

Решение задания:

$a = 23;

if ($a == 5)

echo "10";

else if ($a == 10)

echo "15";

else if ($a == 30)

echo "20";

else

echo "Что-то не то!";

2. Логическая операция

У вас есть две переменные:

$x = 4.2;

$y = 5.6;

Создайте несколько логических операций, которые будут проверять:

является ли $x равным числу 15;

является ли $y большим или равным за 6;

равные ли между собой переменные $x и $y;

Все результаты проверок выведите на экран.

Решение задания:

$x = 4.2;

$y = 5.6;

echo $x == 15;

echo $y >= 6;

echo $x == $y;

1. Большая проверка

Создайте переменную «a» со значением 34.

Создайте проверку, которая будет выполняться при условии, что число будет меньше 50, но не равно 37. При этом не меньше или равно 32.

Допишите в условие проверку, что результат будет также со значением «true» в случае когда переменная «a» будет со значением 0 или 15.

Решение задания:

$a = 34;

echo ($a < 50 && $a != 37 && $a >= 32) || ($a == 0 || $a == 15);

1. Проверка данных

У вас есть две переменные:

$x = 7;

$y = "7";

Проверьте их на равенство лишь значений, а также на равенство и значений, и типов данных.

Выполните проверку и посмотрите на отличия.

Решение задания:

$x = 7;

$y = "7";

$res = $x === $y;

echo "$x === $y = $res";

Если проверять на соответствие лишь значений, то данная операция выведет true. При проверке на эквивалентность будет выведено false.

1. Нечетное число

Из двух чисел с разной четностью вывести на экран нечетное число. a, b - данные числа

Ск

Решение задания:

$a = 2;

$b = 5;

if ($a % 2 == 0)

echo $b.' - нечетное число';

else

echo $a.' - нечетное число';

1. Деление на ноль

Создайте две переменных со значениями 5 и -3.

Создайте переменную, в которую установите значение в зависимости от результата тернарного оператора:

установку деления переменных в случае если делитель не равен числу 0;

установку число 1 в случае если делитель равен числу 0.

Решение задания:

$a = 5;

$b = -3;

$res = $b != 0 ? ($a / $b) : 1;

echo "Результат: $res";