Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра ИТАС

Отчет

по лабораторной работе №2

«Ветвления и функции»

Вариант №17

|  |  |
| --- | --- |
| Выполнил:  ст. гр. 820603  Лаходанов А. Е. | Проверил:  А. Л. Гончаревич |

Минск 2021

# Цель работы

Изучение основных возможностей объектно-ориентированного языка *PHP* в работе с ветвлениями и функциями*.*

# Проектирование

Для каждого задания из лабораторной работы №2 создам программы, содержащие их выполнение.

# Теоретические сведения

*PHP –* это язык программирования, специально разработанный для написания веб-приложений (сценариев) исполняющихся на стороне сервера.

Важным преимуществом языка *PHP* перед языками *Perl* и *C* является возможность создания *HTML*-документов с внедренными командами *PHP*. Конструкции *PHP*, вставленные в *HTML*-текст, выполняются сервером при каждом посещении страницы. Результат их обработки вместе с обычным *HTML*-текстом передается браузеру.

Значительным отличием *PHP* от других скриптов, выполняющихся на стороне клиента (например, *JavaScript*), является то, что скрипты *PHP* выполняются на стороне сервера. Можно сконфигурировать сервер так, чтобы *HTML*-документы обрабатывались процессором *PHP* таким образом, что пользователи даже не смогут определить, получают они обычную *HTML*-страницу либо результат выполнения кода *PHP*.

В *PHP* используется четыре основных (простейших) типа данных (литералов) и соответствующих им переменных: целые числа, числа с плавающей точкой, строчки, булевские значения.

Переменная – это имя (или идентификатор), присваиваемое ячейке памяти компьютера, которая хранит определенные литералы во время исполнения сценария *PHP*.

Если необходимо отобразить значение, присвоенное переменной, достаточно сослаться на нее c помощью оператора*echo*.

Строка – это набор любых символов. В *PHP* существует два способа задания строк в двойных, либо одинарных кавычках. Разница между ними заключается в том, что если внутри строки первого типа написать имя переменной, то интерпретатор *PHP* подставит значение этой переменной при результирующем выводе. В случае же со вторым типом строк (одинарные кавычки) подстановки не происходит.

Константы представляют собой контейнеры для данных, как и переменные, но после присваивания константе значения его уже нельзя изменить. Константы создаются в *РНР* с помощью функции *define().*

Группу операторов составляют операции сравнения: «меньше» (<), «меньше или равно» (<=), «больше» (>) и «больше или равно» (>=). Все они сравнивают два заданных значения и возвращают логическое значение *true* или *false*.

Логические операторы проверяют булевы условия. Существует четыре главных булевых условия: «И» (*and* или &&), «ИЛИ» (*or* или ||), «НЕ» (!) и «исключающее ИЛИ» (*хоr или ^*).

Операция «И» возвращает истину только в том случае, если истинны оба выражения.

Операция «**ИЛИ»** возвращает истину в том случае, если истинно хотя бы одно выражение.

Операция «НЕ» переворачивает значение выражения.

Операция исключающее «ИЛИ» обозначается как *xor*. Он возвращает истину, если один и только один из операндов имеет истинное значение. Если оба операнда имеют истинное значение, оператор вернет ложное значение.

Ветвление – это выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения. По сути, ветвление – это разделение программы на варианты в зависимости от определённых условий.

Оператор *if* в *PHP* (как и *elseif*) используется для выполнения блока кода только в том случае, если указанное условие имеет значение *true*.

Оператор *if … else* позволяет выполнить один блок кода, если указанное условие оценивается как *true*, а другой блок кода, если *false*.

Оператор *switch* сравнивает значение условного выражения с несколькими значениями. Как правило, в качестве выражения используется переменная, в зависимости от значения которой должен быть исполнен тот или иной блок кода. Для сравнения в *switch* используется оператор равенства (==). Оператор *switch* берет значение, возвращаемое условным выражением, и начинает сравнивать его со всеми значениями, стоящими рядом с ключевым словом case (метка), в порядке их следования. Как только совпадение обнаружено, выполняется соответствующий блок кода. Если соответствие не найдено, не исполняется ни один из блоков. Если должен быть выполнен только один блок кода, соответствующий определенному значению, то в конце этого блока следует вставить ключевое слово *break*.

Функция – это блок кода, к которому можно обращаться из разных частей скрипта. Функции могут иметь входные и выходные параметры. Входные параметры могут использоваться в операциях, которые содержит функция. Выходные параметры устанавливаются функцией, а их значения используются после выполнения функции. Функция в *PHP* объявляется с помощью ключевого слова *function*. После указания параметров в фигурных скобках следует тело функции.

Функция может возвращать некоторое значение – число, строку и т.д., то есть некоторый результат. Для возвращения значения в функции применяется оператор *return*, после которого указывается возвращаемое значение.

Внутри блока кода функций можно пользоваться абсолютно любыми средствами языка *PHP* также, как и вне функций. В том числе, существует возможность вызова функцией самой себя. Эта операция называется рекурсией.

Функция *date()* выводит текущие дату и время в заданном формате.

# Ход работы

## Использование оператора *if…elseif…else*

Программа, в которой в зависимости от значений переменных (положительные, отрицательные, разных знаков), будет выводиться их разность, произведение или сумма:

*<?php*

*$a = 10;*

*$b = -8;*

*if ($a >= 0 && $b >= 0)*

*echo $a - $b;*

*elseif ($a < 0 && $b < 0)*

*echo $a \* $b;*

*else*

*echo $a + $b;*

*?>*

При значениях *$a = 10* и *$b = -8* получен результат как на  
рисунке 1.



Рисунок 1 – Вывод результата вычисления

## Работа с оператором *switch*

Вывод на экран чисел от переменной *$a* до 15:

*<?php*

*$a = 10;*

*switch ($a < 16 && $a >= 0){*

*case true:*

*test($a);*

*break;*

*default:*

*echo "error";*

*}*

*function test($a)*

*{*

*if($a > 15)*

*return;*

*echo "$a, ";*

*$a += 1;*

*return test($a);*

*}*

*?>*

В случае *$a = 10,* получена последовательность как на рисунке 2.

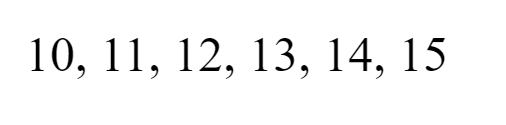


Рисунок 2 – Последовательность чисел

## Использование функций для вычисления арифметических операций

Код страницы, в которой реализованы основные арифметические операции в виде функций с двумя параметрами:

*<?php*

*function sum($x, $y)*

*{*

*return $x + $y;*

*}*

*function razn($x, $y)*

*{*

*return $x - $y;*

*}*

*function pr($x, $y)*

*{*

*return $x \* $y;*

*}*

*function del($x, $y)*

*{*

*return $x % $y;*

*}*

*echo $s = sum(3,6);*

*echo "</br>";*

*echo $r = razn(5,3);*

*echo "</br>";*

*echo $p = pr(2,2);*

*echo "</br>";*

*echo $d = del(8,3);*

*?>*

Результат вычислений четырех арифметических операций показан на рисунке 3.

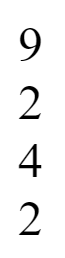


Рисунок 3 – Реализация арифметических операций

Cкрипт, который при помощи функции с тремя параметрами выполняет арифметическую операцию и возвращает полученное значение:

*<?php*

*function mathOperation($arg1, $arg2, $operation)*

*{*

*switch ($operation) {*

*case "+":*

*return sum($arg1, $arg2);*

*case "-":*

*return razn($arg1, $arg2);*

*case "\*":*

*return pr($arg1, $arg2);*

*case "%":*

*return del($arg1, $arg2);*

*default:*

*echo "error";*

*}*

*}*

*function sum($x, $y)*

*{*

*return $x + $y;*

*}*

*function razn($x, $y)*

*{*

*return $x - $y;*

*}*

*function pr($x, $y)*

*{*

*return $x \* $y;*

*}*

*function del($x, $y)*

*{*

*return $x % $y;*

*}*

*echo $res = mathOperation(6, 4, "\*");*

*?>*

При передачи в функцию пармаетров: 6, 4 и «\*» получен результат какна рисунке 4.



Рисунок 4 – Реализация функции с тремя параметрами

## Работа с рекурсией

Программа, которая организует функцию возведения числа в степень:

*<?php*

*function power($val, $pow)*

*{*

*if ($pow == 0)*

*return 1;*

*elseif ($pow > 0 )*

*return $val \* power($val, $pow-1);*

*else*

*return 1/($val \* power($val, abs($pow)-1));*

*}*

*echo power(0.5, 5);*

*?>*

При передаче в функцию параметров 0.5 и 5, на страницу выводится результат как на рисунке 5.

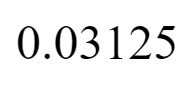


Рисунок 5 – Возведение числа в степень

## Использование функции для работы со временем

Код программы которая вычисляет время и выводит его в формате с правильными склонениями:

*<?php*

*function num\_word($value, $words, $show = true)*

*{*

*$num = $value % 100;*

*if ($num > 19) {*

*$num = $num % 10;*

*}*

*$out = ($show) ? $value . ' ' : '' ;*

*switch ($num) {*

*case 1: $out .= $words[0]; break;*

*case 2:*

*case 3:*

*case 4: $out .= $words[1]; break;*

*default: $out .= $words[2]; break;*

*}*

*return $out;*

*}*

*echo num\_word(date("H"), ['час', 'часа', 'часов'] );*

*echo "&nbsp";*

*echo num\_word( date("i"), ['минута','минуты','минут'] );*

*?>*

Результат вычисления времени продемонстрирован на рисунке 6.

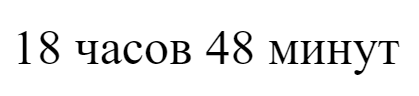


Рисунок 6 – Время в формате с правильными склонениями

Заключение

В результате выполнения лабораторной работы №2 мною разработаны программы, иллюстрирующие выполнение заданий, с использованием такого языка программирования как *PHP*.