**Лабораторна робота №1.**

**Програмування з боку сервера. Введення в PHP. Методи GET та POST.**

Синтаксис мови: змінні, рядки, масиви, функції, область видимості змінних, статичні змінні, цикли, константи, вирази.

Тривалість: 2 акад. години.

Максимальна оцінка: 5.

Мета: набути практичних навичок з використання базових операторів PHP.

Завдання: ознайомитися із теоретичними відомостями та виконати завдання згідно ходу роботи.

Довідковий матеріал: <https://www.php.net/manual/en/index.php>

Синтаксис PHP: <https://www.php.net/manual/en/langref.php>

1.1 Теоретичні відомості.

1.1.1 Стилі PHP- дескрипторів

Існують чотири стилі PHP- дескрипторів (наведені нижче фрагменти коду еквівалентні):

– XML- стиль

**<?php echo ‘<p>Hello world!</p>‘; ?>.**

Даний стиль найбільш поширений (у даних методичних вказівках використовується саме такий стиль), адміністратори серверів не мають можливості вимкнути його, тому він гарантовано доступний у всіх випадках, що важливо у випадку розробки програмного коду, виконання якого передбачається у різних середовищах.

– Скорочений стиль

**<? echo ‘<p>Hello world!</p>‘; ?>.**

Це найпростіший стиль, що відповідає стилю інструкцій обробки мови SGML(StandardGeneralizedMarkupLanguage). Використання його не бажане, оскільки системні адміністратори часто його вимикають для запобігання конфліктів з XML-документами.

– SCRIPT-стиль

**<script language=’php’>echo ‘<p> Hello world!</p>‘; </script>.**

SCRIPT- стиль часом застосовують у випадку виникнення проблем з іншими стилями у HTML-редакторах.

– ASP-стиль

**<% echo ‘<p> Hello world!</p>‘; %>**

Аналогічний стиль дескриптора використовується у технологіях ASP (ActiveServerPages). Такому стилеві надають перевагу при роботі з редактором, орієнтованим на ASP.NET.

1.1.2 Оператори PHP

Дії, котрі повинен виконати інтерпретатор PHP, задаються операторами PHP, розташованими між відкриваючими та закриваючими дескрипторами:

**echo ‘<p>Hello world!</p>‘;**

Оператор echo виконує виведення у вікно браузера переданого йому рядка. Для розділення операторів використовується символ-крапка з комою: “;”.

1.1.3 Пробіли

Порожні символи, такі як порожній рядок (повернення каретки), пробіли між словами та символи табуляції, утворюють категорію пробільних. Браузери ігнорують пробільні символи у HTML-коді. Аналогічно діє і механізм PHP. Пробіли між PHP- операторами не є необхідними, проте вони підвищують читабельність коду. Фрагменти

echo ‘Вітаємо’;

echo ‘на нашому сайті’;

echo ‘Вітаємо’; echo‘ на нашому сайті’;

еквівалентні, але перша версія візуально сприймається легше.

1.1.4 Коментарі

Зазвичай коментарями супроводжуються більшість PHP-сценаріїв, за виключенням найпростіших.

Інтерпретатор PHP ігнорує текст, розміщений у коментарі. Для синтаксичного аналізатора коментар рівнозначний пробілам.

PHP підтримує коментарі у стилі С, С++ і сценаріїв оболонки.

/\*Приклад

багаторядкового коментаря С-стилю\*/

Багаторядкові коментарі починаються з символів /\*і закінчуються символами \*/.Коментарі не можуть бути вкладеними.

Також використовують однорядкові коментарі у стилі C++:

echo ‘<p>Замовлення виконано</p>’; // початок виведення замовлення

або у стилі сценаріїв оболонки

echo ‘<p>Замовлення виконано</p>’; # початок виведення замовлення.

1.1.5 Доступ до змінних форми

Сенс використання форми полягає в отриманні скриптом інформації, введеної користувачем у текстові поля форми. Розглянемо приклад:

<html>

<head>

<title>Приклад форми</title>

</head>

<body>

<form action="form.php" method="post">

<input type="text" name="formvariable">

<input type="submit" value="Додати">

</form>

</body>

</html>

Даний HTML-скрипт створює форму для введення даних і кнопку “Додати”, яка передає введене скриптові form.php

Доступ до вмісту поля PHP-скриптом отримують наступним чином (файл form.php):

<?php

$variable=$\_POST['formvariable'];

echo $variable;

?>

У даному випадкові передача даних реалізована за допомогою методу POST. У цьому випадкові відбувається неявна передача: користувач не помічає ніяких зовнішніх ознак процесу і не може здійснити припущення про імена змінних. Така передача зручна з точки зору безпеки.

Існує інший спосіб передачі даних – GET. При цьому дані передаються за допомогою сформованого додатку до назви файлу, що отримує дані. Зовні виглядає як лінк, при цьому користувач отримує доступ до імен змінних, що передаються, та їхніх значень.

<html>

<head>

<title>Приклад форми</title>

</head>

<body>

<form action="form.php" method="get">

<input type="text" name="formvariable">

<input type="submit" value="Додати">

</form>

</body>

</html>

Даний приклад відрізняється від попереднього методом передачі: замість POST використано GET.Нехай у поле введено значення “test”. Після натискання кнопки “Додати” буде особливим чином активізовано скрипт form.php: до адреси файлу буде додано символ “?”, назву змінної, переданої з форми, знак “=” та значення змінної:

http://localhost/form.php?formvariable=test

PHP-скрипт при цьому змінюють наступним чином:

<?php

$variable=$\_GET['formvariable'];

echo $variable;

?>

Масиви $\_POST та $\_GET належать до категорії суперглобальних. Як у першому, так і у другому випадках, доступ до даних форми можна отримати через масив $\_REQUEST.

<?php

$variable=$\_REQUEST['formvariable'];

echo $variable;

?>

Детально використання масивів буде розглянуто у лабораторній роботі №2.

Зазвичай блоки отримання даних з форми розташовують на початкові скрипта.

1.1.6 Конкатенація рядків

Операція конкатенації рядків використовується для об’єднання рядків (фрагментів тексту) у єдиний текст. Вона часто застосовується при виведенні даних у браузер за допомогою оператора echo. Реалізується за домового символу оператора конкатенації – крапки “.”.

**<?php**

**$text1="Hello, ";**

**$text2="world!";**

**echo $text1.$text2;**

**?>**

**<?php**

**$text1="world!";**

**echo "Hello, ".$text1;**

**?>**

Довільну змінну, відмінну від змінної масиву, можна помістити у подвійні лапки і застосувати до неї оператор echo. Зверніть увагу на результат виконання двох наступних скриптів:

<?php

$text1="world!";

echo "Hello, $text1";

?>

Буде виведено у вікно браузера: Hello, world!

<?php

$text1="world!";

echo 'Hello, $text1';

?>

Результат: Hello, $text1.

Якщо ім’я змінної знаходиться між подвійними лапками, то ім’я замінюється значенням змінної, якщо ім’я змінної чи довільний текст обмежені одинарними лапками, то вони передаються без змін.

1.1.7 Ідентифікатори

Ідентифікатори – це імена змінних, функцій та класів. Використання ідентифікаторів регламентується наступними правилами:

– ідентифікатори можуть мати довільну довжину і складатися з букв, цифр та символів підкреслювання “\_”;

– ідентифікатори не можуть починатися з цифри;

– у PHP ідентифікатори чутливі до регістру символів. Змінні $formvariable $FormVariable– це різні змінні. Виключення складають вбудовані PHP-функції – їхні імена можуть бути представлені довільним регістром;

– змінні можуть мати імена, що співпадають з іменами вбудованих функції. Однак, це може призвести до плутанини, тому подібних ситуацій слід уникати. Не можна також створювати функції, чиї імена співпадають з іменами вбудованих функцій.

Імена змінних у PHP починаються знаком долара ($). Пропуск цього знаку – поширена помилка.

1.1.8 Типи змінних

Тип змінної характеризується видом даних, котрі вона зберігає. PHP підтримує наступні базові типи даних:

– Integer (цілий) – використовується для представлення цілих чисел;

– Float (ще їх називають double – подвійної точності) – використовується для представлення дійсних чисел;

– String (рядковий) – використовується для представлення символів;

– Boolean (булевий) – використовується для зберігання значень true та false;

– Array(масив) – використовується для зберігання декількох елементів даних (буде розглянутий у лабораторній роботі №2);

– Object (об’єкт) – використовується для зберігання екземплярів класу (буде розглянутий у лабораторній роботі №5).

Доступні також два спеціальних типи – NULL та resource (ресурс). Змінні, котрим не присвоєно конкретного значення, котрі не визначені або набувають значення NULL, належать до типу NULL. Деякі вбудовані функції (такі, як для роботи з базами даних) повертають змінну ресурсного типу (наприклад, з’єднання з базами даних).

Також PHP підтримує типи pdfdoc та pdfinfo, якщо встановлена підтримка обробки PDF-документів.

Мова PHP – слабо типізована. На відміну від мови С, тип змінної визначається типом присвоєного їй значення.

1.1.9 Перетворення типів

За допомогою механізму перетворення типів можна переводити змінну або конкретне значення до іншого типу. Перетворення відбувається так само, як і на мові С.

<?php

$a=2.34; // Змінна $a має тип float

$a=(int)$a; // Тепер змінна $a має тип integer

echo $a;

?>

1.1.10 Змінні змінних

Усі мови програмування дозволяють змінювати значення змінної, деякі – тип змінної і зовсім небагато мов дозволяють змінювати ім’я змінної.

В основі цієї можливості – ідея використання значення однієї змінної як імені іншої.

Нехай існує змінна $variable. Тоді:

<?php

$variable=NULL;

$varname="variable";

$$varname="Hello, world!";

echo $variable;

?>

Запис $$varname="Hello, world!" еквівалентний наступному: $variable="Hello, world!".

1.1.11 Константи

Разом із використанням змінних, PHP надає можливість оголошення констант. Як і змінна, константа містить значення, але її значення встановлюється одноразово і не може бути зміненим у сценарії.

Константи оголошуються за допомогою функції define

<?php

define('CONSVALUE',10);

echo CONSVALUE;

?>

Зверніть увагу на те, що імена констант записують символами у верхньому регістрі. Дана особливість перейшла з мови С. Дотримуватися її не обов’язково, проте вона значно спрощує читання і супровід коду.

Важливо: при звертанні до констант не використовується знак долара ($).

1.1.12 Область дії змінних

Термін “область дії” належить тим розділам сценаріїв, в яких можливий доступ до деякої конкретної змінної, іншими словами, це – область, з довільного місця якої видно дану змінну. У PHP використовують шість базових правил визначення області дії:

– вбудовані суперглобальні змінні видно з довільного місця сценарію;

– константи, щойно вони оголошені, видно глобально, тобто можуть використовуватися як зовні так і всередині функцій;

– глобальні змінні, оголошені у сценарії, видно у довільному місці сценарію, але не всередині функцій;

– змінні, що використовуються всередині функцій, що оголошені як глобальні, посилаються на глобальні змінні з тими самими іменами;

– змінні, що використовуються всередині функцій, що оголошені як статичні, невидимі поза межами функцій, проте вони зберігають свої значення поміж двома викликами цих функцій;

– змінні, створені всередині функцій, є локальними стосовно своєї функції і припиняють існування після завершення функції.

Вбудовані суперглобальні змінні у PHP версій від 4.1 і вище наступні:

– $\_GLOBALS – масив глобальних змінних. Дає можливість доступу до глобальних змінних всередині функції, наприклад: $\_GLOBALS[‘myvariable’];

– $\_SERVER – масив змінних середовища сервера;

– $\_GET – масив змінних, переданих у сценарій за допомогою методу GET;

– $\_POST – масив змінних, переданих у сценарій за допомогою методу POST;

– $\_COOKIE – масив cookie- змінних;

– $\_FILES – масив змінних завантаження файлів;

– $\_ENV – масив змінних оточення;

– $\_REQUEST – масив, пов’язаний із введенням даних користувачем, включаючи $\_GET, $\_POST, $\_COOKIE;

– $\_SESSION – масив змінних сеансу.

1.1.13 Посилання

Операція посилання позначається як “&” і може використовуватися у поєднанні з операцією присвоювання.

Зазвичай, коли значення змінної $a присвоюється змінній $b, створюється копія змінної $a, яка зберігається у пам’яті. Якщо у майбутньому значення $a буде змінене, то $b зміни не торкнуться.

$a=5;

$b=$a; // $b=5;

$a=6; // як і раніше, $b=5

Створення копії можна уникнути, використавши операцію посилання &:

$a=5;

$b=&$a; // $b дорівнює 5;

$a=6; // зверніть увагу: $b дорівнює 6

$a та $b вказують на одну і ту ж ділянку пам’яті. Цей зв’язок можна розірвати, “скинувши” одну зі змінних:

unset($a);

При цьому значення $b продовжуватиме існувати.

1.1.14 Операції порівняння

Операції порівняння наведено у табл.1.1. Результатом виконання операції порівняння є true або false.

Таблиця 1.1 – Операції порівняння PHP

|  |  |  |
| --- | --- | --- |
| Операція | Назва | Використання |
| ==  ===  !=  !==  <>  <  >  <=  >= | дорівнює  тотожно  не дорівнює  не тотожно  не дорівнює  менше  більше  менше або дорівнює  більше або дорівнює | $a == $b  $a === $b  $a != $b  $a !== $b  $a <> $b  $a < $b  $a > $b  $a <= $b  $a >= $b |

Операція перевірки тотожності повертає значення true тільки у тому випадкові, якщо обидва операнди рівні і мають однаковий тип. Наприклад:

$a=0;

$b='0';

echo($a==$b); // результат: true

echo($a===$b); // результат: false

Логічні операції слугують для комбінування результатів логічних умов. Перелік логічних операцій разом з описом їхнього застосування наведено у табл. 1.2.

Таблиця 1.2 – Логічні операції PHP

|  |  |  |  |
| --- | --- | --- | --- |
| Операція | Назва | Використання | Результат |
| !  &&  ||  and  or | НЕ  І  АБО  І  АБО | !$b  $a && $b  $a || $b  $a and $b  $a or $b | Повертається true, якщо значення $a дорівнює false та навпаки  Повертається true, якщо обидві змінні $a та $b мають значення true, в іншому випадкові повертається значення false  Повертається true, якщо довільна зі змінних $a або $b має значення true, інакше повертається false  Те саме, що і &&, але з меншим пріоритетом  Те саме, що і ||, але з меншим пріоритетом |

1.1.15 Оператори if

Для прийняття рішень використовується оператор if. Операторові необхідно задати умову. Якщо умова рівна true, то виконується блок коду, розташований після оператора. Умова в операторі if записується поміж круглими дужками “(…)”. Приклад:

<?php

$a=0;

$b=3;

if($a<$b) echo '$a < $b';

?>

1.1.16. Блоки коду

Часто всередині такого умовного оператора, як if, необхідно виконати більше одного оператора. У такому випадку відповідна послідовність операторів записується у вигляді блоку. Для оголошення блоку оператори необхідно оточити фігурними дужками:

if($a<$b)

{

echo '$a < $b';

$a=$b;

}

Таким чином при виконанні умови ($a<$b) буде виконано обидва оператори

echo '$a < $b';

$a=$b;

Якщо ж умова не виконуватиметься, то обидва оператори будуть проігнорованими.

1.1.17. Оператори else

Оператор else дозволяє визначити альтернативну дію, котра повинна виконатися, якщо значення умови в операторі if виявиться false. Наприклад:

<?php

$a=0;

$b=3;

echo '$a='.$a.'; $b='.$b.'<br />';

if($a<$b)

echo '$a < $b';

else

echo '$a < $b';

?>

Результат виконання скрипта:

$a=0; $b=3

$a< $b

Вкладання операторів if один в одного дозволяє будувати складні ланцюжки.

1.1.18 Оператори elseif

У багатьох випадках прийняття рішення передбачає вибір відповідного варіанту з деякої множини можливих варіантів. Послідовність цієї множити можна створити за допомогою оператора комбінації операторів if – else. За наявності послідовності умов програма може перевіряти кожну з них до тих пір, поки не знайде таку, значення якої буде true. Наприклад:

<?php

$a=23;

if($a>0 && $a<10) echo 'Результат - у першому інтервалі';

elseif($a>=10 && $a<20) echo 'Результат - у другому інтервалі';

elseif($a>=20 && $a<30) echo 'Результат - у третьому інтервалі';

?>

Після виконання скрипта отримують: “Результат - у третьому інтервалі”

1.1.19 Оператори switch

Оператор switch працює аналогічно до оператора if, але надає можливість умовному виразові мати як результат більше двох значень. В операторі if умова набуває значення true або false. Натомість в операторі switch умова може набувати довільну кількість значень у тих випадках, коли результат його обчислення – простий тип (integer, string чи float). Для забезпечення реагування на кожне таке значення слід передбачити для нього відповідний оператор case, а також (не обов’язково) визначити дії, що виконуватимуться по замовчуванню, якщо виникне випадок, не передбачений оператором case. Наприклад:

<?php

$a=1;

switch ($a)

{

case 1:

echo '$a=1';

break;

case 2:

echo '$a=2';

break;

default:

echo '$a='.$a;

break;

}

?>

Результат виконання: $a=1.

1.1.20 Цикли while

Найпростішим циклом у PHP є цикл while. Різниця між оператором if та оператором while полягає у тому, що у випадку виконання умови оператор if виконує блок коду тільки один раз, а оператор while повторює його до тих пір, поки умова виконується.

Наприклад:

<?php

$var=1;

while($var<=5)

{

echo $var.'<br />';

$var++;

}

?>

1.1.21 Цикли for

Цикл типу while можна записати і у більш компактній формі за допомогою оператора for. Базова структура оператора for має вигляд:

for(вираз 1; умова; вираз 2)

вираз 3;

Вираз 1 виконується один раз на початкові циклу. Як правило, він встановлює початкове значення змінної циклу.

Вираз умова перевіряється перед кожною ітерацією. Якщо цей вираз повертає значення false – цикл зупиняється.

Вираз 2 виконується у кінці кожної ітерації. Зазвичай у ньому змінюється значення лічильника.

Вираз 3 виконується один раз під час кожної ітерації. Це, як правило, – блок коду, який містить тіло циклу. Наступний приклад виводить табличку з двох стовпчиків і п’яти рядків.

<?php

echo "<table>";

for($distance=50; $distance<=250; $distance+=50)

{

echo "<tr>\n <td align='right'>$distance</td>\n";

echo " <td align=right'>".$distance."</td>\n</tr>\n";

}

echo "</table>";

?>

1.1.22 Цикли do…while

Загальні структура оператора do…while має вигляд:

do

вираз;

while(умова 1);

Цикл do…while відрізняється від циклу while тим, що в ньому умова перевіряється у кінці. Отже, у циклі do…while оператор або блок операторів всередині циклу завжди виконується, принаймі, один раз.

<?php

$a=10;

do

{

echo $a.'<br />';

$a-=1;

}

while($a>=0);

?>

* 1. Хід роботи

1. Зареєструватися на веб-хостингу <https://www.zzz.com.ua>. (чи на будь-якому іншому). Задати пароль для FTP-з’єднання. Якщо Ви вже є зареєстровані і термін дії закінчується не швидше 30 червня 2021 року, то наново реєструватися не потрібно.
2. Налаштувати редактор текстів (Notepad++, VS code, Sublime тощо) для роботи з FTP-сервером. (див. файл «Безкоштовний текстовий редактор Notepad.docx» для налаштування Notepad++).
3. Скопіювати на FTP-сервер папку Web. Розібратися із запропонованими викладачем файлами. Переглянути їх через браузер.

**Увага! Усі роботи мають знаходитися у відповідних папках labN, де N – номер роботи. Усі назви файлів, в тому числі і з розширеннями .jpg, .txt і т.д. у папках labN мають починатися із вашого прізвища на латині, якщо ні, то буду знижувати оцінки! Якщо у завдання задана конкретна назва файлу, наприклад *example1\_1\_5\_1.php* ви його перейменовуєте на *your\_last\_name\_example1\_1\_5\_1.php****.*

1. У файл config.php в межах блоку php додати наступний код:

$LastModified\_unix = strtotime(date("D, d M Y H:i:s", filectime($\_SERVER['SCRIPT\_FILENAME'])));

$LastModified = gmdate("D, d M Y H:i:s \G\M\T", $LastModified\_unix);

echo "Last modified: $LastModified"."<br>";

Файл config.php повинен бути приєднаним до всіх файлів всіх лаб. робіт і бути єдиним (цієї вимоги також потрібно дотримуватися щодо інших файлів і папок, які знаходяться безпосередньо в папці Web, окрім папок labN), тоді даний код буде виводити повідомлення, коли будь-який із файлів був останній раз змінений.

1. Реалізувати і дослідити функціонування операторів, дія яких розглядається у теоретичних відомостях. Посилання (назва повинна містити номер підпункту прикладу) на файл із реалізацією кожного із прикладів розмістити у файлі lab1.php.
2. Модифікувати приклад в файлі example1\_1\_5\_1.php наступним чином:

* вбудувати html-код в php-блок;
* додати ще одну форму із текстовим полем і кнопкою, яка б методом GET передавала в example1\_1\_5\_2.php введене число. Одержане число у файлі example1\_1\_5\_2.php помножити на 2, вивести із вказівкою, що вхідні дані одержані із файлу example1\_1\_5\_1.php;

1. Прислати на адресу: [mariia.dutchak@pnu.edu.ua](mailto:mariia.dutchak@pnu.edu.ua) посилання на стартову сторінку Вашого сайту, в темі листа вказати ПІБ і групу. Посилання на всі наступні роботи теж розміщувати на стартовій сторінці.
2. Виконану роботу архівуєте, назва архіву має починатися із Вашого прізвища на кирилиці, даний архів здаєте у класрум. Якщо робота здана у класрум вчасно (а також дата останньої зміни будь-якого із файлів лаб. роботи на хостингу не пізніша терміну здачі у класрум), то вона оцінюється максимум із 5 балів, якщо невчасно, то із 4. А якщо пари закінчаться, а Ви тоді згадаєте, що потрібно здавати роботи, то без захисту максимум 3. Якщо хочете підвищити оцінку - потрібно захищати роботу.

Контрольні запитання:

1. Які типи PHP- дескрипторів існують на даний час?

2. Поясніть суть оператора у PHP.

3. Які особливості використання пробілів?

4. Які типи коментарів використовують при програмуванні на PHP?

5. Яким чином реалізують доступ до змінних форми?

6. Що таке конкатенація рядків та як вона реалізується?

7. Поясніть зміст поняття “ідентифікатор” на PHP.

8. Які типи змінних підтримує PHP?

9. Поясніть суть поняття “змінна змінних”?

10. Яким чином оголошуються константи на PHP?

11. Поясніть суть поняття “область дії змінних”?

12. Яке призначення посилань?

13. Операції порівняння на PHP.

14. Яким чином реалізують розгалуження у програмі?

15. Які оператори для створення циклів надає PHP?