Temat: Zmienne, stałe, typy danych i operatory

Zespół Szkół Komunikacji ; Gabriela Ledachowicz

Kryteria:

1. Podaję sposoby i zasady definiowania zmiennych w php. Wyjaśniam różnice w ich wykorzystaniu w zależności od użytej składni

PHP – Zmienne

- ✓Przed nazwą zmiennej należy umieścić znak **$**
- ✓Zmienna jest tworzona przy pierwszym jej użyciu
- ✓Nie trzeba wcześniej deklarować zmiennej ani określać jej typu
- ✓Musi zaczynać się od litery lub znaku podkreślenia
- ✓Może składać się jedynie z liter, cyfr i znaku podkreślenia
- ✓W nazwach rozróżniane są małe i duże litery
- ✓W nazwach można stosować polskie litery

PHP – Zmienne

✓**Wartość przypisujemy zmiennej za pomocą znaku = (operator przypisania)**.

✓Wartość zmiennej umieszczamy w **cudzysłowie lub apostrofie**. Każde przypisanie kończymy średnikiem!

```
$x=”naleśniki”;
```

✓Przypisana wartość może być wyrażeniem, które zawiera inne zmienne:

```
$oMnie = ”Nazywam się Jan i bardzo lubię $x”;
```

✓Można przypisać wartości pojedynczym zmiennym:

```
$a=20; $b=6;
```

lub kilku, hurtowo:

```
$d=$e=$f=$g=20;
```

PHP – Użycie zmiennych w ciągach znaków z wykorzystaniem cudzysłowu i apostrofu

✓PHP nie dokonuje interpretacji ciągu znaków ujętego w **apostrofach**, jeśli np. wyświetlimy go na ekranie, pojawi się w większości przypadków w niezmienionej postaci, dotyczy to szczególnie zmiennych

✓ łańcuch znakowy ujęty w znaki **cudzysłowu** – jeśli w tego typu sekwencji znajdzie się określenie zmiennej, zostanie ono zmienione na wartość przez nią reprezentowaną

PHP – Użycie zmiennych w ciągach znaków z wykorzystaniem cudzysłowu i apostrofu

```php
<?php
    $x="pierogi";
    $oMnie="Nazywam się Gabriela i
    bardzo lubię $x";
    $oMnie2='Nazywam się Piotr i też
    bardzo lubię $x';

    $X='sushi';
    $oMnie3="Jestem Rafał. Uwielbiam $X!";
    $oMnie4='Jestem Anka i też uwielbiam
    $X!!!';
    echo $oMnie,"<br>",$oMnie2,"<br>",
    $oMnie3,"<br>",$oMnie4;
?>
```

Nazywam się Gabriela i bardzo lubię pierogi
Nazywam się Piotr i też bardzo lubię $x
Jestem Rafał. Uwielbiam sushi!
Jestem Anka i też uwielbiam $X!!!

http://localhost/zsk/l4p9.php

PHP – Wykorzystanie składni heredoc i nowdoc

```php
<?php
    $name = "Gabriela";
    $food = "pierogi";

    $here_doc = <<<HERE
    Nazywam się $name <br>
    i uwielbiam programować<br>
HERE;
echo $here_doc;

$now_doc = <<<'NOW'
    Nazywam się $name <br>
    i uwielbiam $food
NOW;
echo $now_doc;
?>
```

Nazywam się Gabriela
i uwielbiam programować
Nazywam się $name
i uwielbiam $food

http://localhost/zsk/l4p10.php

PHP – Wykorzystanie składni heredoc i nowdoc

- ✓Zmienne zawarte w ciągu znaków korzystającym z **heredoc**, podobnie jak w przypadku składni z **cudzysłowami**, zostaną zamienione na odpowiadające im wartości.
- ✓w **nowdoc** ciąg nie jest interpretowany, a zatem nazwy zmiennych nie są zamieniane na odpowiadające im wartości.
- ✓różnica między nowdoc a heredoc jest taka jak między użyciem znaków apostrofu i cudzysłowu.

Kryteria:

1. Podaję sposoby i zasady definiowania zmiennych w php. Wyjaśniam różnice w ich wykorzystaniu w zależności od użytej składni

---

Pokaż światłami/kciukami jak oceniasz, na ile spełniasz kryterium

Kryteria:

2. Wyjaśniam czym są stałe i jak mogę je zastosować

PHP – Stałe i ich wykorzystanie

- ✓Stałe często wykorzystuje się w skryptach, które kilkukrotnie wykorzystują tę samą wartość. W momencie, gdy wartość ta ulega zmianie, zachodzi konieczność zmiany jej we wszystkich miejscach. Może to rodzić błędy
- ✓Można wartość zdefiniować raz, a następnie odwoływać się do niej wielokrotnie w programie poprzez nazwę.
- ✓Jeżeli zajdzie konieczność zmiany wartości, wystarczy to zrobić jednokrotnie, a odniesie to skutek wszędzie, gdzie jest ona wykorzystywana.

PHP – Stałe i ich wykorzystanie

Stałe mogą być definiowane przez użytkownika za pomocą funkcji

`define(),`

która przyjmuje 2 parametry: **nazwę stałej i wartość do niej przypisaną**.

```
define(”NAZWA_STALEJ”, ”WARTOSC_STALEJ”);
```

PHP – Stałe i ich wykorzystanie

Definiowana wartość może być :

- ✓ liczbą całkowitą:

```
define('LICZBA_STRON', 4);
```

- ✓ napisem:

```
define('TYTUL', 'Lorem ipsum');
```

- ✓ wartością logiczną:

```
define('CZY_DRUKOWAC_TYTUL', true);
```

- ✓ liczbą zmiennopozycyjną:

```
define('LICZBA_PI', 3.1415);
```

PHP – Stałe i ich wykorzystanie

```php
<?php
    define("STALA", "Witaj świecie.");
    echo STALA;
    echo "<br>";
    define("NR_TEL","666258147");
    echo NR_TEL;
?>
```

Witaj świecie.
666258147

http://localhost/zsk/l4p11.php

PHP – Różnice pomiędzy stałymi i zmiennymi:

- ✓nazwy stałych nie są poprzedzone znakiem dolara (*$*);
- ✓stałe mogą zostać zdefiniowane jedynie przy użyciu funkcji **define()** a nie poprzez zwykłe przypisanie;
- ✓mogą być definiowane i używane wszędzie, niezależnie od zasięgu zmiennych;
- ✓nie mogą być zmieniane ani usuwane jeśli raz zostały ustawione ;
- ✓mogą zawierać jedynie wartości skalarne.

Kryteria:

2. Wyjaśniam czym są stałe i jak mogę je zastosować

Pokaż światłami/kciukami jak oceniasz, na ile spełniasz kryterium

Kryteria:

3. Podaję stosowane w PHP typy danych i krótko je opisuję

PHP – typy danych

W PHP występuje **osiem** podstawowych typów danych. Typy te określają m.in. rodzaje danych, jakie mogą być przechowywane przez zmienne. Typy danych można podzielić na trzy różne rodzaje:

- ✓typy skalarne
- ✓typy specjalne
- ✓typy złożone

PHP – Typy skalarne

Typy skalarne (ang. scalar types) to inaczej **typy proste** (ang. primitive types). Dzielą się one na cztery rodzaje. Są to:

- ✓typ boolean
- ✓typ integer
- ✓typ float
- ✓typ string

PHP – Typ boolean

Jest to typ logiczny, który może przyjmować tylko dwie wartości: **true (prawda) oraz false (fałsz**). Ten typ wykorzystywany jest przy konstruowaniu wyrażeń logicznych oraz sprawdzaniu warunków.

PHP – Typ integer

✓Jest to typ całkowitoliczbowy, dzięki któremu można reprezentować zarówno dodatnie, jak i ujemne **liczby całkowite**.

✓Liczby mogą być zapisane w trzech różnych formatach: dziesiętnym, ósemkowym (oktalnym) lub szesnastkowym (heksadecymalnym). **Domyślnie stosowany jest format dziesiętny.**

✓Maksymalny zakres typu całkowitego zależy od platformy sprzętowo-systemowej, na której uruchamiane jest PHP. W przypadku przekroczenia zakresu wartość jest konwertowana na typ float.

PHP – Typ float

✓ reprezentuje zarówno dodatnie, jak i ujemne liczby rzeczywiste (zmiennopozycyjne, zmiennoprzecinkowe).
✓zakres, podobnie jak dla typu integer, jest zależny od platformy sprzętowo-systemowej; z reguły są to wartości od –1,8×10308 do 1,8×10308.
✓Typową reprezentacją jest **zapis z kropką dziesiętną**, czyli np. **1.5**.
✓Można również używać notacji wykładniczej w postaci X.YeZ, gdzie X to część całkowita, Y część dziesiętna, natomiast Z to wykładnik potęgi liczby 10.

PHP – Typ string

✓ typ łańcuchowy, który służy do zapamiętywania sekwencji znaków.

✓Nie ma ograniczenia długości tego ciągu.

✓Łańcuch znaków można utworzyć na poznane wcześniej cztery sposoby używając:

- znaków apostrofu,
- znaków cudzysłowu,
- składni heredoc,
- składni nowdoc.

PHP – Typ string

Kodowanie wybranych znaków specjalnych

Sekwencja znaków	Znaczenie
\n	Nowa linia (ang. *new line*).
\\	Lewy ukośnik (ang. *backslash*).
\$	Znak dolara.
\"	Znak cudzysłowu

Począwszy od PHP7 można stosować sekwencje specjalne \u{kod}, gdzie kod oznacza kod znaku w formacie Unicode

Np. sekwencja \u{62}oznacza małą literę b, a \u{017c}— małą literę ż.

PHP – typy danych

Funkcja PHP **`var_dump ()`** zwraca typ i wartość danych

```php
<?php
$x = "ZSK";
echo "Typ tekstowy: ",$x,"<br>";
var_dump($x);
echo "<br>";
$y = 5985;
echo "Liczba całkowita: ",$y, "<br>";
var_dump($y);
echo "<br>";
$z = 12.34;
echo "Liczba zmiennoprzecinkowa: ",$z, "<br>";
var_dump($z);
echo "<br>";
$a = true;
echo "Wartość logiczna: ",$a, "<br>";
var_dump($a);
?>
```

Typ tekstowy: ZSK
string(3) "ZSK"
Liczba całkowita: 5985
int(5985)
Liczba zmiennoprzecinkowa: 12.34
float(12.34)
Wartość logiczna: 1
bool(true)

http://localhost/zsk/l4p12.php

PHP – Typy specjalne

Typy specjalne dzielą się na dwa rodzaje. Są to:

- ✓**Typ resource** jest typem specjalnym wskazującym, że zmienna przechowuje odwołanie do zasobu zewnętrznego utworzonego za pomocą specjalnych funkcji.
- ✓**Typ null** jest typem informującym, że dana zmienna nie przechowuje żadnej wartości. Jeżeli chcemy ustawić zmienną na null, piszemy:

```
$zmienna = null;
```

Wielkość liter nie ma przy tym znaczenia, prawidłowe są zatem zapisy: null, NULL, Null, czy nUlL.

PHP – Typy złożone

Dzielą się na dwa rodzaje. Są to:

- ✓typ array (tablicowy),
- ✓typ object (obiektowy) – na kolejnych lekcjach.

PHP – typ array (tablicowy),

Tablice to występujące w większości języków programowania struktury, pozwalające na przechowywanie zbioru danych określonego typu. Zawartością pojedynczej komórki tablicy może być wartość dowolnego typu danych.
W PHP dostępne są dwa rodzaje tablic:

- ✓ klasyczne (indeksowane numerycznie)
- ✓ asocjacyjne.

Dostęp do poszczególnych danych zawartych w tablicy uzyskuje się poprzez podanie indeksu (inaczej klucza), pod którym dana wartość została zapisana.

PHP – typ array (tablicowy),

Tablica indeksowana numerycznie:

✓ 1 sposób

używając słowa kluczowego array:

```
$tablica = array(wartość1, wartość2,..., wartośćN);
```

można również zastosować zapis w postaci:

```
$tablica = array
(
   wartość1,
   wartość2,
   ...,
   wartośćN
);
```

PHP – typ array (tablicowy),

Tablica indeksowana numerycznie:

Do wyświetlenia zawartości tablicy używamy funkcji **print_r**

```php
<?php
// tworzenie tablicy indeksowanej numerycznie 1sp.:
$frameworks = array("Symfony", "Zend", "Laravel");
/*
echo $frameworks;
// echo nie pozwala na wyświetlenie całej tablicy
*/
// wyświetlenie zawartości tablicy:
print_r ($frameworks);
// wyświetlenie w wielu liniach:
print "<br><pre>";
print_r ($frameworks);
print "</pre>";
echo "<br>Wyświetlenie wartości
zapisanej w danej komórce:";
echo "<br>".$frameworks[1];
?>
```

```
Array ( [0] => Symfony [1] => Zend [2] => Laravel )

Array
(
    [0] => Symfony
    [1] => Zend
    [2] => Laravel
)


Wyświetlenie wartości zapisanej w danej komórce:
Zend
```

http://localhost/zsk/l4p13.php

PHP – typ array (tablicowy),

Tablica indeksowaną numerycznie:

✓ 2 sposób (nowszy)

używając nawiasów kwadratowych:

```
$tablica = [wartość1, wartość2,..., wartośćN];
```

można również zastosować zapis w postaci:

```
$tablica =[
   wartość1,
   wartość2,
   ...,
   wartośćN
];
```

PHP – typ array (tablicowy),

Tablica indeksowana numerycznie:

```php
<?php
// tworzenie tablicy indeksowanej numerycznie 2sp.:
$auto = [
"Toyota",
2000,
"gaz"];
print_r ($auto);
// wyświetlenie w wielu liniach:
echo "<br><pre>";
print_r ($auto);
echo "</pre>";
echo "<br>Wyświetlenie wartości
zapisanej w danej komórce:";
echo "<br>".$auto[2];
?>
```

Array ([0] => Toyota [1] => 2000 [2] => gaz)

```
Array
(
    [0] => Toyota
    [1] => 2000
    [2] => gaz
)
```

Wyświetlenie wartości zapisanej w danej komórce
gaz

http://localhost/zsk/l4p13a.php

PHP – typ array (tablicowy),

Tablica indeksowaną numerycznie:

✓ 3 sposób

poprzez bezpośrednie przypisywanie wartości jej komórkom.

Zamiast pisać:

```
$kolory = array("czerwony", "zielony",
"niebieski");
```

można wykorzystać serię instrukcji w postaci:

```
$kolory[0] = "czerwony";
$kolory[1] = "zielony";
$kolory[2] = "niebieski";
```

PHP – typ array (tablicowy),

Dodawanie i zmiana zawartości tablicy:

```php
<?php
// tworzenie tablicy indeksowanej numerycznie:
$kolory = array("czerwony", "zielony");
print_r ($kolory);
// dodanie kolejnej wartości:
$kolory[]="fioletowy";
// wyświetlenie w wielu liniach:
echo "<br>Po dodaniu wartości:<pre>";
print_r ($kolory);
echo "</pre>";
// zmiana zawartości:
$kolory[1]="pomarańczowy";
echo "<br>Po zamianie zawartości:<pre>";
print_r ($kolory);
echo "</pre>";
?>
```

Array ([0] => czerwony [1] => zielony)
Po dodaniu wartości:

```
Array
(
    [0] => czerwony
    [1] => zielony
    [2] => fioletowy
)
```

Po zamianie zawartości:

```
Array
(
    [0] => czerwony
    [1] => pomarańczowy
    [2] => fioletowy
)
```

http://localhost/zsk/l4p13b.php

PHP – typ array (tablicowy),

Tablica asocjacyjna:

✓każdemu indeksowi można nadać unikalną nazwę, zamiast indeksów 0, 1, 2 itd. mogą występować indeksy: kolor, autor, procesor itp.

✓zamiast terminu indeks stosuje się inny termin - **klucz**.

✓Mówimy zatem, że w tablicy asocjacyjnej występują pary **klucz – wartość**, w których każdy klucz jednoznacznie identyfikuje przypisaną mu wartość

PHP – typ array (tablicowy),

Tablica asocjacyjna:

```php
<?php
// tworzenie tablicy asocjacyjnej:
$person = array(
"imie" => "Zenon",
"nazwisko"=> "Małolepszy",
"waga"=>"lekka" );
echo "<pre>";
print_r ($person);
echo "</pre>";
// zmiana zawartości:
$person["nazwisko"]="Ciutlepszy";
echo "Po zamianie zawartości:<pre>";
print_r ($person);
echo "</pre>";
// odwołanie do pojedynczych wartości:
echo $person["imie"]." ".$person["nazwisko"].
" zmienił nazwisko z Małolepszego";
?>
```

```
Array
(
    [imie] => Zenon
    [nazwisko] => Małolepszy
    [waga] => lekka
)

Po zamianie zawartości:

Array
(
    [imie] => Zenon
    [nazwisko] => Ciutlepszy
    [waga] => lekka
)

Zenon Ciutlepszy zmienił nazwisko z Małolepszego
```

http://localhost/zsk/l4p14.php

Kryteria:

3. Podaję stosowane w PHP typy danych i krótko je opisuję

Pokaż światłami/kciukami jak oceniasz, na ile spełniasz kryterium

Kryteria:

4. Określam rodzaje operatorów stosowanych w PHP, analizuję działanie wybranych operatorów

PHP – Operatory

Wykorzystywane do wykonywania operacji na zmiennych i wartościach.

PHP dzieli operatory na różne grupy, m.in.:

- ✓ operatory arytmetyczne
- ✓ operatory przypisania
- ✓ operatory inkrementacji i dekrementacji
- ✓ operatory porównania
- ✓ operatory logiczne
- ✓ operatory łączące napisy

PHP – operatory arytmetyczne

- ✓są używane z wartościami liczbowymi
- ✓służą do wykonywania typowych operacji arytmetycznych, takich jak dodawanie, odejmowanie, mnożenie itp.

Operator	Nazwa	Przykład
+	dodawanie	$x + $y
-	odejmowanie	$x - $y
*	mnożenie	$x * $y
/	dzielenie	$x / $y
%	dzielenie modulo/reszta z dzielenia	$x % $y
**	potęgowanie	$x ** $y

PHP – skrócone operatory arytmetyczne/operatory przypisania

oznaczenie	rozumiany jako	opis
$x = $y	$x = $y	Lewy argument zostaje ustawiony na wartości wyrażenia z prawej
$x += $y	$x = $x + $y	dodanie
$x -= $y	$x = $x - $y	odejmowanie
$x *= $y	$x = $x * $y	mnożenie
$x /= $y	$x = $x / $y	dzielenie
$x %=$y	$x = $x % $y	dzielenie modulo (reszta z dzielenia)

PHP – operatory inkrementacji i dekrementacji

Służą do zwiększania lub zmniejszania wartości zmiennej o 1.

++$a zwiększa $a o 1 a następnie zwraca wartość $a

$a++ zwraca wartość $a, a następnie zwiększa $a o 1

--$a zmniejsza $a o 1 a następnie zwraca wartość $a

$a-- zwraca wartość $a, a następnie zwiększa $a o 1

PHP – operatory inkrementacji i dekrementacji

Jaki ciąg liczb otrzymamy?

```php
<?php
/*1*/ $x = 1;
/*2*/ echo $x++,"<br>";
/*3*/ echo ++$x,"<br>";
/*4*/ echo $x,"<br>";
/*5*/ $y = $x++;
/*6*/ echo $y,"<br>";
/*7*/ $y = ++$x;
/*8*/ echo $y,"<br>";
/*9*/ echo ++$y;
?>
```

http://localhost/zsk/l4p15.php

PHP – operatory inkrementacji i dekrementacji

```php
<?php
/*1*/ $x = 1;
/*2*/ echo $x++,"<br>";
/*3*/ echo ++$x,"<br>";
/*4*/ echo $x,"<br>";
/*5*/ $y = $x++;
/*6*/ echo $y,"<br>";
/*7*/ $y = ++$x;
/*8*/ echo $y,"<br>";
/*9*/ echo ++$y;
?>
```

1
3
3
3
5
6

PHP – operatory porównania

Służą do porównywania dwóch wartości

operator	nazwa	przykład	rezultat
==	Równy	$x == $y	Zwraca true jeśli $x jest równe $y
===	Identyczny	$x === $y	Zwraca true jeśli $x jest równe $y, i są one tego samego typu
!=	Nie równe	$x != $y	Zwraca true jeśli $x nie jest równe $y
<>	Nie równe	$x <> $y	Zwraca true jeśli $x nie jest równe $y
!==	Nie identyczny	$x !== $y	Zwraca true jeśli $x nie jest równe $y, lub nie są tego samego typu
>	Większy niż	$x > $y	Zwraca true jeśli $x jest większe niż $y
<	Mniejszy niż	$x < $y	Zwraca true jeśli $x jest mniejsze niż $y
>=	Większe bądź równe	$x >= $y	Zwraca true jeśli $x jest większe lub równe $y
<=	Mniejsze lub równe	$x <= $y	Zwraca true jeśli $x jest mniejsze lub równe $y

PHP – operatory logiczne

Używane do łączenia warunków w instrukcjach warunkowych

operator	nazwa	przykład	rezultat
and	i	$x and $y	Prawda jeśli $x i $y jest prawdziwe
or	lub	$x or $y	Prawda jeśli $x lub $y jest prawdziwe
xor	xor	$x xor $y	Prawda jeśli $x lub $y jest prawdziwe ale nie oba
&&	i (wyższy priorytet)	$x && $y	Prawda jeśli $x i $y jest prawdziwe
||	lub(wyższy priorytet)	$x || $y	Prawda jeśli $x lub $y jest prawdziwe
!	nie	!$x	Prawda jeśli $x nie jest prawdą

PHP – Operatory dotyczące napisów/konkatenacji

Służy do łączenia dwóch łańcuchów znaków

operator	nazwa	przykład
.	Łączenie napisów	$txt1 . $txt2
.=	Dołączanie napisu	$txt1 .= $txt2

```php
<?php
    $skrot="PHP";
    $nazwa="Hypertext Preprocessor";
    echo  $skrot." to rekurencyjny skrót,
    pochodzący od ".$nazwa."!";
?>
```

http://localhost/zsk/l4p16.php

Kryteria:

4. Określam rodzaje operatorów stosowanych w PHP, analizuję działanie wybranych operatorów

Pokaż światłami/kciukami jak oceniasz, na ile spełniasz kryterium