INSTRUKCJA LABORATORYJNA

PHP: OPERATORY LOGICZNE, FORMULARZE I SWITCH

Klasa 4TI | 2 x 45 minut

> **Założenie zajęć**
> Materiał zawiera całą potrzebną teorię oraz jedno zadanie końcowe. Nie wykonujesz już ćwiczeń 1, 2 ani 3. Czytaj kolejno sekcje, a następnie wykonaj zadanie końcowe krok po kroku.

Imię i nazwisko: .....

Data: ..... Stanowisko: .....

Cel: połączyć formularz HTML z logiką programu PHP.

1. Co będziemy robić?

Na tych zajęciach nie wracamy do podstaw zmiennych ani polecenia echo. Skupiamy się na tym, jak program internetowy podejmuje decyzje na podstawie danych wpisanych lub wybranych przez użytkownika w formularzu.

łączyć warunki za pomocą && (AND), || (OR) oraz ! (NOT),

wysyłać dane z formularza metodą POST i odbierać je przez $_POST,

sprawdzać checkbox za pomocą isset(),

wykorzystywać pola text, number, radio, checkbox i select,

stosować if / elseif / else do warunków złożonych,

stosować switch dla kilku konkretnych wartości.

Najważniejsza myśl
HTML zbiera dane. PHP odbiera dane i podejmuje decyzję. Operator logiczny pozwala połączyć kilka warunków w jedną decyzję.

2. Plan pracy - 2 x 45 minut

Czas	Etap	Co robimy
0-15 min	Operatory logiczne	&&, ||, ! oraz nawiasy
15-30 min	HTML -> POST -> PHP	form, name, value, $_POST
30-45 min	Pola wyboru	radio i checkbox
45-55 min	select + switch	obsługa konkretnego wyboru
55-60 min	Bezpieczne dane	htmlspecialchars() i typowe błędy
60-90 min	Zadanie końcowe	jedna aplikacja wykonywana krok po kroku

3. Jak dane przechodzą z formularza do PHP?

Schemat przepływu
1. Użytkownik wypełnia formularz HTML -> 2. klika przycisk Wyślij -> 3. przeglądarka wysyła dane metodą POST -> 4. PHP odczytuje dane z $_POST -> 5. if / switch sprawdza wartości -> 6. PHP generuje odpowiedź.

Przykład: pole HTML ma name="wiek". PHP odczytuje je jako $_POST["wiek"]. Atrybut name jest mostem między formularzem a PHP.

```
<input type="number" name="wiek">
<?php
$wiek = $_POST["wiek"] ?? 0;
?>
```

Dlaczego używamy ??
Zapis $_POST["wiek"] ?? 0 oznacza: pobierz wartość pola, a jeśli nie została przesłana - użyj 0. Dzięki temu nie otrzymujemy ostrzeżenia o brakującym kluczu.

4. Operatory logiczne - łączenie warunków

4.1. Operator && - AND - „i jednocześnie”

Operator && zwraca prawdę tylko wtedy, gdy wszystkie połączone nim warunki są prawdziwe.

```
$wiek = 19;
$regulamin = true;

if ($wiek >= 18 && $regulamin == true) {
    echo "Można kontynuować";
} else {
    echo "Brak dostępu";
}
```

Warunek A	Warunek B	A && B
prawda	prawda	prawda
prawda	fałsz	fałsz
fałsz	prawda	fałsz
fałsz	fałsz	fałsz

Czytaj warunek jak zdanie

$wiek >= 18 && $regulamin == true -> „wiek wynosi co najmniej 18 lat I regulamin został zaakceptowany”.

4.2. Operator || - OR - „lub”

Operator || wymaga, aby przynajmniej jeden z warunków był prawdziwy.

```
$rola = "moderator";
if ($rola == "admin" || $rola == "moderator") {
    echo "Dostęp do panelu";
} else {
    echo "Brak uprawnień";
}
```

Warunek A	Warunek B	A || B
prawda	prawda	prawda
prawda	fałsz	prawda
fałsz	prawda	prawda
fałsz	fałsz	fałsz

4.3. Operator ! - NOT - „nie”

Operator ! odwraca wartość logiczną. Prawda staje się fałszem, a fałsz prawdą.

```
$zablokowany = false;
if (!$zablokowany) {
    echo "Konto nie jest zablokowane";
}
```

Jak czytać !$zablokowany?
Jeżeli $zablokowany ma wartość false, to !$zablokowany ma wartość true. Czytamy: „jeżeli użytkownik NIE jest zablokowany”.

4.4. Nawiasy - gdy warunek robi się dłuższy

Nawiasy pomagają jednoznacznie określić kolejność sprawdzania warunków. Stosuj je w dłuższych warunkach - kod jest wtedy czytelniejszy.

```
if (($wiek >= 18 && $regulamin) || $rola == "admin") {
    echo "Dostęp przyznany";
}
```

Odczytaj warunek
Dostęp otrzyma osoba pełnoletnia, która zaakceptowała regulamin, LUB użytkownik o roli admin. Nawias grupuje dwa pierwsze warunki.

5. Formularz HTML - elementy, które musisz rozumieć

```
<form method="POST">
    <label for="wiek">Wiek:</label>
    <input type="number" id="wiek" name="wiek" min="0" required>
    <button type="submit">Sprawdź</button>
</form>
```

Element	Co oznacza?
...	Obszar formularza. Pola znajdujące się wewnątrz mogą zostać wysłane do serwera.
method="POST"	Dane są wysyłane metodą POST i w PHP odbieramy je z tablicy $_POST.
label / for	Etykieta i powiązanie jej z id pola.
type="number"	Pole przeznaczone do wpisywania liczby.
id="wiek"	Identyfikator elementu w HTML.
name="wiek"	Najważniejsza nazwa dla PHP - pod tym kluczem wartość trafia do $_POST.
min / required	Walidacja po stronie przeglądarki.
button type="submit"	Wysyła formularz.

name i id to nie to samo
id służy głównie po stronie HTML/CSS/JavaScript. name decyduje, pod jaką nazwą wartość zostanie wysłana do PHP. Jeśli zabraknie name, PHP nie otrzyma wartości tego pola.

6. Jak poprawnie odebrać formularz w PHP?

```
<?php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $wiek = (int)($_POST["wiek"] ?? 0);

    if ($wiek >= 18) {
        echo "Użytkownik pełnoletni";
    } else {
        echo "Użytkownik niepełnoletni";
    }
}
?>
```

Wyjaśnienie: REQUEST_METHOD sprawdza sposób wysłania strony; $_POST["wiek"] pobiera pole name="wiek"; ?? 0 ustawia wartość domyślną; (int) zamienia tekst na liczbę całkowitą.

> **Ważne**
> required i min pomagają użytkownikowi, ale logika PHP nadal powinna samodzielnie sprawdzać najważniejsze warunki.

7. Pole radio - wybór jednej opcji

Pola radio stosujemy, gdy użytkownik ma wybrać dokładnie jedną opcję. Wszystkie przyciski w jednej grupie mają taki sam name, ale różne value.

```
<fieldset>
  <legend>Rodzaj klienta</legend>
  <label><input type="radio" name="typ" value="standard" required> Standard</label>
  <label><input type="radio" name="typ" value="vip"> VIP</label>
</fieldset>

<?php
$typ = $_POST["typ"] ?? "";
?>
```

> **Co naprawdę wysyła radio?**
> PHP otrzymuje wartość z atrybutu value. Jeśli zaznaczono value="vip", to $_POST["typ"] zawiera tekst "vip".

8. Checkbox - pole, którego może nie być w POST

```
<label><input type="checkbox" name="regulamin"> Akceptuję regulamin</label>

<?php
$regulamin = isset($_POST["regulamin"]);
?>
```

Jeżeli checkbox nie jest zaznaczony, jego nazwa może w ogóle nie pojawić się w $_POST. Dlatego używamy isset().

9. Select + switch - wybór konkretnej wartości

```
<label for="usluga">Wybierz usługę:</label>
<select id="usluga" name="usluga" required>
  <option value="diagnostyka">Diagnostyka</option>
  <option value="czyszczenie">Czyszczenie</option>
  <option value="system">Instalacja systemu</option>
</select>
```

Jeżeli jedna zmienna może przyjmować kilka konkretnych wartości, switch jest czytelniejszy niż długi łańcuch porównań.

```
$usluga = $_POST["usluga"] ?? "";

switch ($usluga) {
  case "diagnostyka":
    $cena = 50;
    break;
  case "czyszczenie":
    $cena = 80;
    break;
  case "system":
    $cena = 150;
    break;
  default:
    $cena = 0;
}
```

Dlaczego break?

Po znalezieniu pasującego case PHP ma zakończyć bieżący przypadek. Bez break kod może przejść do następnego case.

10. Pełny przykład - formularz i logika PHP

Najpierw HTML zbiera dane, a następnie PHP je odbiera i podejmuje decyzję.

```
<form method="POST">
  <input type="number" name="wiek" min="0" required>
  <label><input type="radio" name="typ" value="uczen" required> Uczeń</label>
  <label><input type="radio" name="typ" value="nauczyciel"> Nauczyciel</label>
  <label><input type="checkbox" name="regulamin"> Akceptuję regulamin</label>
  <button type="submit">Sprawdź</button>
</form>
```

```php
<?php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $wiek = (int)($_POST["wiek"] ?? 0);
    $typ = $_POST["typ"] ?? "";
    $regulamin = isset($_POST["regulamin"]);

    if ($wiek >= 18 && $regulamin) {
        echo "Warunki podstawowe spełnione.<br>";
        switch ($typ) {
          case "uczen": echo "Utworzono konto ucznia."; break;
          case "nauczyciel": echo "Utworzono konto nauczyciela."; break;
          default: echo "Nieznany typ konta.";
        }
    } else {
        echo "Rejestracja niemożliwa.";
    }
}
?>
```

11. Wyświetlanie tekstu wpisanego przez użytkownika

Jeżeli wyświetlasz tekst wpisany przez użytkownika, zastosuj htmlspecialchars(). Znaki HTML zostaną potraktowane jako tekst, a nie kod strony.

```php
$imie = $_POST["imie"] ?? "";
echo "Witaj " . htmlspecialchars($imie, ENT_QUOTES, "UTF-8");
```

Do zapamiętania

Danych z formularza nie należy bezmyślnie wstawiać do HTML. htmlspecialchars() jest prostym zabezpieczeniem przy wyświetlaniu tekstu podanego przez użytkownika.

12. Najczęstsze błędy - sprawdź zanim poprosisz o pomoc

Problem	Przyczyna	Co sprawdzić?
PHP nie widzi pola	Brak name lub inna nazwa w $_POST	Porównaj name="..." z kluczem $_POST["..."]
Radio wybiera kilka opcji	Różne wartości name	Cała grupa radio musi mieć ten sam name
Checkbox daje błąd	Niezaznaczone pole nie zostało wysłane	Użyj isset($_POST["nazwa"])
switch wykonuje kolejny case	Brak break	Dodaj break po obsłudze przypadku
Warunek działa inaczej	Zły && / || lub brak nawiasów	Przeczytaj warunek po polsku i pogrupuj nawiasami
Formularz nic nie robi	Brak submit lub reakcji na POST	Sprawdź button type="submit" i REQUEST_METHOD

13. ZADANIE KOŃCOWE - krok po kroku

> **Zapis na warsztaty informatyczne**
> Wykonaj plik warsztaty.php. To jedyne zadanie do samodzielnego wykonania. Rób kroki po kolei i uruchamiaj stronę po każdym większym etapie.

Cel zadania: formularz ma zapisać uczestnika na wybrany warsztat. Użyjesz: POST, $_POST, text, number, radio, checkbox, select, isset(), &&, ||, !, switch oraz htmlspecialchars().

KROK 1. Utwórz formularz

Przepisz formularz. Na tym etapie sprawdź tylko, czy wyświetla się w przeglądarce.

```
<form method="POST">
  <label>Imię:</label>
  <input type="text" name="imie" required>

  <label>Wiek:</label>
  <input type="number" name="wiek" min="0" required>

  <label><input type="radio" name="typ" value="uczen" required> Uczeń</label>
  <label><input type="radio" name="typ" value="nauczyciel"> Nauczyciel</label>

  <label>Warsztat:</label>
  <select name="warsztat" required>
    <option value="html">HTML i CSS</option>
    <option value="php">PHP</option>
    <option value="bazy">Bazy danych</option>
  </select>

  <label><input type="checkbox" name="regulamin"> Akceptuję regulamin</label>
  <button type="submit">Zapisz</button>
</form>
```

KROK 2. Otwórz część PHP

```
<?php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    // tutaj będą kolejne kroki
}
?>
```

Zadanie końcowe - dalsze kroki

KROK 3. Odbierz dane

Wewnątrz bloku POST utwórz zmienne: $imie, $wiek, $typ, $warsztat i $regulamin.

```
$imie = $_POST["imie"] ?? "";
$wiek = (int)($_POST["wiek"] ?? 0);
$typ = $_POST["typ"] ?? "";
$warsztat = $_POST["warsztat"] ?? "";
$regulamin = isset($_POST["regulamin"]);
```

KROK 4. Sprawdź regulamin - operator !

Jeżeli regulamin NIE został zaznaczony, wyświetl: Musisz zaakceptować regulamin. Użyj if (!$regulamin).

KROK 5. Zapisz warunek udziału

Jeżeli regulamin jest zaakceptowany, warunek brzmi: (uczeń ma co najmniej 16 lat) LUB (użytkownik jest nauczycielem). Użyj &&, || i nawiasów. Podpowiedź: część ucznia to $wiek >= 16 && $typ == "uczen".

KROK 6. Rozpoznaj warsztat przez switch

Dla html ustaw „Sala 101”, dla php - „Sala 102”, dla bazy - „Sala 103”. Każdy case zakończ break. Dodaj default.

```
switch ($warsztat) {
  case "html":
    $sala = "Sala 101";
    break;
  case "php":
    // ustaw Sala 102
    break;
  case "bazy":
    // ustaw Sala 103
    break;
  default:
    $sala = "Nieznana sala";
}
```

KROK 7. Wyświetl wynik

Dla poprawnego zapisu wyświetl: imię, typ uczestnika, wybrany warsztat i salę. Imię wyświetl przez htmlspecialchars($imie, ENT_QUOTES, "UTF-8"). Gdy warunek udziału nie jest spełniony, wyświetl Rejestracja niemożliwa.

KROK 8. Przetestuj

- uczeń 17 lat + regulamin
- uczeń 14 lat + regulamin
- nauczyciel + regulamin
- odznaczony regulamin - sprawdź operator !

> **Szybka kontrola przed oddaniem**
> POST i name są poprawne; dane są pobierane z $_POST; checkbox używa isset(); w kodzie występują &&, || i !; switch ma case, break i default; imię przechodzi przez htmlspecialchars().

14. Ściąga - jedna strona do powtórki

Zapis	Znaczenie
A && B	A i B muszą być prawdziwe
A || B	wystarczy, że A lub B jest prawdziwe
!A	negacja - „nie A”
$_POST["wiek"]	wartość pola z name="wiek"
$_POST["wiek"] ?? 0	pobierz wiek, a jeśli go brak - użyj 0
isset($_POST["regulamin"])	sprawdź, czy checkbox został wysłany
$_SERVER["REQUEST_METHOD"] === "POST"	sprawdź, czy formularz został wysłany metodą POST
name="typ"	nazwa klucza widocznego dla PHP
value="vip"	wartość przesyłana do PHP
switch ($x)	wybierz kod zależnie od konkretnej wartości $x
case "vip":	przypadek, gdy wartość to "vip"
break;	zakończ bieżący przypadek switch
default:	obsłuż wartość, która nie pasowała do żadnego case
htmlspecialchars()	bezpieczne wyświetlenie tekstu podanego przez użytkownika

Szkielet, który warto zapamiętać

```
<form method="POST">
  <!-- pola z atrybutami name -->
  <button type="submit">Wyślij</button>
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
  $wartosc = $_POST["pole"] ?? "";
  $checkbox = isset($_POST["checkbox"]);

  if (warunek1 && warunek2) {
    // wykonaj kod
  }

  switch ($wartosc) {
    case "opcja1":
      // wykonaj kod
      break;
    default:
      // przypadek domyślny
  }
}
?>
```

Na koniec odpowiedz sobie

Czy potrafisz wskazać, co w formularzu decyduje o nazwie klucza w $_POST? Czy potrafisz zapisać słowami warunek zawierający && i ||? Czy wiesz, dlaczego checkbox sprawdzamy przez isset()? Jeżeli tak - osiągnąłeś główne cele tej instrukcji.