**Moduł 1: Wprowadzenie do PHP**

**Cel:** Zrozumienie czym jest PHP i jak działa w kontekście aplikacji webowych.

**Tematy:**

* Czym jest PHP i do czego służy
* Instalacja środowiska (XAMPP, MAMP, Laragon lub Docker)
* Podstawowa składnia
* Pliki .php, echo, komentarze

**Zadanie praktyczne:**

* Utwórz plik index.php i wypisz "Witaj świecie!".
* Stwórz prostą stronę HTML z osadzonym kodem PHP.

**🔤 Moduł 2: Zmienne i typy danych**

**Cel:** Praca z danymi w PHP.

**Tematy:**

* Zmienne i zasady ich nazywania
* Typy danych (string, integer, float, boolean, null, array, object)
* Operatory matematyczne i logiczne

**Zadanie:**

* Napisać kalkulator dodający dwie liczby wpisane przez użytkownika.

**🔁 Moduł 3: Instrukcje warunkowe i pętle**

**Cel:** Podejmowanie decyzji i powtarzanie czynności w kodzie.

**Tematy:**

* if, else, elseif, switch
* Pętle: for, while, do...while, foreach

**Zadanie:**

* Napisz program wypisujący liczby od 1 do 100 i zaznaczający liczby parzyste.

**📚 Moduł 4: Funkcje i zasięg zmiennych**

**Cel:** Organizowanie kodu w funkcje.

**Tematy:**

* Tworzenie funkcji
* Parametry i zwracanie wartości
* Zasięg zmiennych (global, local, static)

**Zadanie:**

* Zbuduj funkcję obliczającą silnię liczby.

**📦 Moduł 5: Tablice i operacje na nich**

**Cel:** Praca z wieloma danymi naraz.

**Tematy:**

* Tablice indeksowane i asocjacyjne
* Iteracja po tablicach
* Funkcje tablicowe (array\_push, count, array\_merge, in\_array itd.)

**Zadanie:**

* Stwórz tablicę z listą produktów i wypisz ich nazwy i ceny.

**🧩 Moduł 6: Formularze i dane od użytkownika**

**Cel:** Interakcja z użytkownikiem strony.

**Tematy:**

* Praca z formularzami HTML (POST i GET)
* Walidacja danych
* isset(), empty(), htmlspecialchars()

**Zadanie:**

* Stwórz formularz kontaktowy z walidacją imienia i adresu e-mail.

**🗃️ Moduł 7: Praca z plikami i sesjami**

**Cel:** Trwałe przechowywanie danych.

**Tematy:**

* Odczyt/zapis plików (fopen, fwrite, fread, file\_get\_contents)
* Praca z sesjami (session\_start(), $\_SESSION)
* Ciasteczka (setcookie())

**Zadanie:**

* Zapamiętaj imię użytkownika po zalogowaniu (sesja).

**💾 Moduł 8: Wstęp do pracy z bazami danych (MySQL)**

**Cel:** Przechowywanie danych w bazie.

**Tematy:**

* Połączenie z bazą danych (mysqli, PDO)
* Tworzenie, odczyt, aktualizacja i usuwanie danych (CRUD)
* Podstawy SQL

**Zadanie:**

* Zbuduj prosty system notatek zapisujący dane w MySQL.

## ****Ćwiczenie: Kalkulator dodający dwie liczby wpisane przez użytkownika****

### 🎯 ****Cel ćwiczenia:****

* Utrwalenie pracy ze **zmiennymi**
* Poznanie **typów danych** i konwersji
* Praktyka użycia **formularzy**, **operatorów matematycznych**, i funkcji $\_POST

### 🧠 ****Krok po kroku – jak to zrobić****

1. **Utwórz formularz HTML**, w którym użytkownik może wpisać dwie liczby.
2. **Odbierz dane w PHP** za pomocą $\_POST.
3. **Sprawdź, czy dane zostały przesłane**.
4. **Zamień dane na liczby** i dodaj je do siebie.
5. **Wyświetl wynik**.

### 📄 ****Kod przykładowy (plik:**** kalkulator.php****)****

php

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Kalkulator Dodawania</title>

</head>

<body>

<h2>Prosty Kalkulator Dodawania</h2>

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

<label for="liczba1">Liczba 1:</label>

<input type="number" name="liczba1" required><br><br>

<label for="liczba2">Liczba 2:</label>

<input type="number" name="liczba2" required><br><br>

<input type="submit" value="Dodaj">

</form>

<?php

// Sprawdź, czy formularz został wysłany

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Pobierz wartości z formularza

$liczba1 = $\_POST["liczba1"];

$liczba2 = $\_POST["liczba2"];

// Dodaj liczby

$suma = $liczba1 + $liczba2;

// Wyświetl wynik

echo "<h3>Wynik dodawania: $suma</h3>";

}

?>

</body>

</html>

### ****Co ćwiczymy tutaj?****

* **Zmienne**: $liczba1, $liczba2, $suma
* **Typy danych**: liczby całkowite (int), ale pole input type="number" może też przyjąć liczby zmiennoprzecinkowe (float)
* **Operatory matematyczne**: + (dodawanie)
* **Formularz HTML + PHP**: przesyłanie danych metodą POST
* **Warunek**: sprawdzenie, czy dane zostały przesłane

### 💡 ****Dodatkowe zadania dla chętnych:****

1. Dodaj możliwość odejmowania, mnożenia i dzielenia (wybór operacji z listy rozwijanej).
2. Obsłuż przypadek dzielenia przez 0.
3. Zrób wersję z walidacją – np. sprawdź, czy wpisano liczby.

## ****Ćwiczenie: Wypisywanie liczb od 1 do 100 z zaznaczeniem liczb parzystych****

### 🎯 ****Cel ćwiczenia:****

* Ćwiczenie **pętli** (for)
* Poznanie instrukcji warunkowych (if, else)
* Utrwalenie podstawowej logiki i składni

### 🧠 ****Krok po kroku – co robimy:****

1. Tworzymy pętlę for, która przechodzi od 1 do 100.
2. Dla każdej liczby sprawdzamy, czy jest **parzysta**.
3. Jeśli tak – wypisujemy ją z informacją, że jest parzysta.
4. Jeśli nie – wypisujemy ją zwyczajnie.

### 📄 ****Przykładowy kod (plik:**** petla.php****)****

php

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Liczby od 1 do 100</title>

</head>

<body>

<h2>Liczby od 1 do 100</h2>

<?php

for ($i = 1; $i <= 100; $i++) {

// Sprawdzenie, czy liczba jest parzysta

if ($i % 2 == 0) {

echo "<strong>$i (parzysta)</strong><br>";

} else {

echo "$i<br>";

}

}

?>

</body>

</html>

### 🔍 ****Wyjaśnienie kodu:****

| **Element** | **Co robi** |
| --- | --- |
| for ($i = 1; $i <= 100; $i++) | Pętla liczy od 1 do 100 |
| $i % 2 == 0 | Sprawdza, czy liczba dzieli się bez reszty (czyli jest parzysta) |
| echo | Wypisuje dane na ekran |
| <strong> | HTML – pogrubia liczby parzyste |

### 🧠 ****Warianty i rozszerzenia (dla chętnych):****

1. **Zmienna stylizacja** – parzyste na zielono, nieparzyste na czerwono:

php

if ($i % 2 == 0) {

echo "<span style='color:green;'>$i (parzysta)</span><br>";

} else {

echo "<span style='color:red;'>$i</span><br>";

}

1. **Użyj pętli while** zamiast for:

php

$i = 1;

while ($i <= 100) {

if ($i % 2 == 0) {

echo "<strong>$i (parzysta)</strong><br>";

} else {

echo "$i<br>";

}

$i++;

}

1. **Użyj switch**, choć nie jest konieczny w tym przypadku – ale można dodać np. opis dla konkretnych liczb:

php

switch ($i) {

case 50:

echo "Połowa drogi!<br>";

break;

}

### ✅ ****Czego się uczysz:****

* Logiki: sprawdzania warunku parzystości (% 2 == 0)
* Pętli i kontroli przepływu (for, if, else)
* Formatowania wyników HTML z poziomu PHP

### Co to jest silnia?

Silnia liczby n (oznaczana jako n!) to iloczyn wszystkich liczb całkowitych dodatnich ≤ n.  
**Przykład:**  
5! = 5 × 4 × 3 × 2 × 1 = 120

## 🧠 ****Krok po kroku – jak to napisać:****

1. Tworzymy funkcję silnia($n).
2. Sprawdzamy, czy liczba jest większa od 0.
3. Jeśli n = 0, to silnia wynosi 1 (bo 0! = 1).
4. W pętli mnożymy wszystkie liczby od 1 do n.
5. Zwracamy wynik (return).

## 📄 ****Przykładowy kod (plik:**** silnia.php****)****

php

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Obliczanie silni</title>

</head>

<body>

<h2>Obliczanie silni liczby</h2>

<?php

// Funkcja obliczająca silnię liczby

function silnia($n) {

if ($n < 0) {

return "Silnia nie istnieje dla liczb ujemnych!";

}

$wynik = 1; // zmienna lokalna

for ($i = 1; $i <= $n; $i++) {

$wynik \*= $i;

}

return $wynik;

}

// Przykładowe użycie funkcji

$liczba = 5;

echo "Silnia liczby $liczba wynosi: " . silnia($liczba);

?>

</body>

</html>

### 🔍 ****Wyjaśnienie kodu:****

| **Linia** | **Co się dzieje** |
| --- | --- |
| function silnia($n) | Definicja funkcji z parametrem $n |
| $wynik = 1 | Zmienna lokalna (tylko wewnątrz funkcji) |
| for ($i = 1; $i <= $n; $i++) | Pętla obliczająca iloczyn |
| return $wynik | Funkcja zwraca wynik |
| $liczba = 5 | Zmienna zewnętrzna (globalna dla tego pliku) |
| silnia($liczba) | Wywołanie funkcji z argumentem |

## 🌐 ****Zasięg zmiennych – wyjaśnienie:****

* **Lokalny:** zmienna dostępna tylko wewnątrz funkcji ($wynik, $i)
* **Globalny:** zmienna zdefiniowana poza funkcją ($liczba)
* **Static:** zachowuje swoją wartość pomiędzy wywołaniami

### 🧪 Przykład zmiennej static:

php

function licznik() {

static $liczba = 0;

$liczba++;

echo "Funkcja została wywołana $liczba razy.<br>";

}

licznik(); // 1

licznik(); // 2

licznik(); // 3

### 💡 ****Dodatkowe wyzwania:****

1. Zrób wersję **rekurencyjną** funkcji silnia()
2. Umożliw użytkownikowi wpisanie liczby przez formularz
3. Obsłuż błędne dane (np. tekst, liczby ujemne)

## ****Krok po kroku – jak to zrobić:****

1. Stworzymy **tablicę asocjacyjną** z nazwami produktów jako kluczami i ich cenami jako wartościami.
2. Użyjemy pętli foreach, aby przejść przez wszystkie elementy tablicy.
3. Wyświetlimy każdy produkt w ładnym formacie.

## 📄 ****Przykładowy kod (plik:**** produkty.php****)****

php

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Lista Produktów</title>

</head>

<body>

<h2>Nasze produkty:</h2>

<?php

// Tablica asocjacyjna z nazwami i cenami produktów

$produkty = [

"Chleb" => 4.50,

"Masło" => 6.20,

"Mleko" => 3.80,

"Ser" => 8.50,

"Jajka (10 szt.)" => 9.00

];

// Iteracja po tablicy i wypisanie produktów

foreach ($produkty as $nazwa => $cena) {

echo "Produkt: <strong>$nazwa</strong> – Cena: <em>$cena zł</em><br>";

}

?>

</body>

</html>

### 🔍 ****Wyjaśnienie kodu:****

| **Fragment** | **Co robi** |
| --- | --- |
| $produkty = [...] | Tworzy **tablicę asocjacyjną** – klucze to nazwy produktów, wartości to ceny |
| foreach ($produkty as $nazwa => $cena) | Przechodzi przez każdy produkt, oddzielając nazwę i cenę |
| echo ... | Wypisuje dane w formacie HTML z prostym formatowaniem |

## 🔧 ****Przykładowe użycie funkcji tablicowych:****

### 📌 count() – liczba produktów:

php

echo "Liczba produktów: " . count($produkty);

### 📌 in\_array() – sprawdzenie, czy w tablicy jest jakaś cena:

php

if (in\_array(3.80, $produkty)) {

echo "Produkt w cenie 3.80 zł jest dostępny.";

}

### 📌 array\_push() – dodanie nowego produktu (lepiej do tablic indeksowanych):

php

// Do indeksowanej tablicy, np. $lista = [];

array\_push($produkty, "Nowy produkt"); // niezalecane dla asocjacyjnych

Dla tablic asocjacyjnych lepiej:

php

$produkty["Kawa"] = 15.00;

### 💡 ****Rozszerzenia (dla chętnych):****

1. Dodaj możliwość **sortowania** produktów według ceny (asort) lub nazwy (ksort).
2. Podlicz **łączną wartość zakupów**:

php

$suma = array\_sum($produkty);

echo "Łączna wartość: $suma zł";

1. Użyj HTML-owej **tabeli** zamiast br do wyświetlania listy produktów.

## ****Ćwiczenie: Formularz kontaktowy z walidacją imienia i adresu e-mail****

### 🎯 ****Cel ćwiczenia:****

* Tworzenie prostych **formularzy HTML**
* Odbieranie danych za pomocą $\_POST
* **Walidacja danych** (czy pola są wypełnione i poprawne)
* Użycie funkcji PHP: isset(), empty(), htmlspecialchars()

### 🧠 ****Krok po kroku – co zrobimy:****

1. Stworzymy formularz HTML z dwoma polami: imię i e-mail.
2. Po przesłaniu formularza dane będą sprawdzane po stronie PHP:
   * Czy pola nie są puste?
   * Czy adres e-mail ma poprawny format?
3. Dane zostaną wyczyszczone z potencjalnych niebezpiecznych znaków.
4. Wyświetlimy potwierdzenie albo komunikaty o błędach.

## 📄 ****Przykładowy kod (plik:**** formularz.php****)****

php

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Formularz kontaktowy</title>

</head>

<body>

<h2>Formularz kontaktowy</h2>

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

Imię: <input type="text" name="imie"><br><br>

E-mail: <input type="text" name="email"><br><br>

<input type="submit" value="Wyślij">

</form>

<?php

// Sprawdzenie, czy dane zostały przesłane

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Czy pola zostały ustawione?

if (isset($\_POST["imie"]) && isset($\_POST["email"])) {

// Oczyszczenie danych

$imie = htmlspecialchars(trim($\_POST["imie"]));

$email = htmlspecialchars(trim($\_POST["email"]));

// Sprawdzenie, czy pola nie są puste

if (empty($imie) || empty($email)) {

echo "<p style='color:red;'>Proszę wypełnić oba pola.</p>";

}

// Sprawdzenie poprawności e-maila

elseif (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

echo "<p style='color:red;'>Adres e-mail jest niepoprawny.</p>";

}

// Jeśli wszystko jest ok

else {

echo "<p style='color:green;'>Dziękujemy, $imie! Otrzymaliśmy Twój kontakt: $email</p>";

}

} else {

echo "<p style='color:red;'>Niepoprawne dane formularza.</p>";

}

}

?>

</body>

</html>

### 🔍 ****Wyjaśnienie ważnych funkcji i pojęć:****

| **Funkcja** | **Co robi** |
| --- | --- |
| isset() | Sprawdza, czy zmienna została ustawiona |
| empty() | Sprawdza, czy zmienna jest pusta (np. puste pole) |
| htmlspecialchars() | Zabezpiecza dane przed XSS – zamienia np. < na &lt; |
| trim() | Usuwa białe znaki z początku i końca ciągu |
| filter\_var(..., FILTER\_VALIDATE\_EMAIL) | Sprawdza, czy e-mail jest w poprawnym formacie |

## ****Ćwiczenie: Zapamiętanie imienia użytkownika po zalogowaniu (sesja)****

### 🎯 ****Cel ćwiczenia:****

* Zrozumienie jak działa **sesja użytkownika**
* Praktyczne użycie session\_start() oraz zmiennej $\_SESSION
* Umożliwienie „zapamiętania” użytkownika podczas przeglądania strony

### 🧠 ****Opis działania:****

1. Użytkownik wpisuje imię w formularzu logowania.
2. Dane są przesyłane do skryptu PHP.
3. PHP zapisuje imię użytkownika do **zmiennej sesyjnej**.
4. Podczas kolejnych wizyt (w tej samej sesji) – witamy użytkownika z imienia.

## 📄 ****Przykład – 2 pliki:****

### ✅ ****Plik 1:**** login.php ****– logowanie użytkownika****

php

<?php

session\_start(); // Uruchamiamy sesję

// Jeśli formularz został przesłany

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

if (!empty($\_POST["imie"])) {

$\_SESSION["imie"] = htmlspecialchars(trim($\_POST["imie"])); // Zapisujemy imię do sesji

header("Location: welcome.php"); // Przekierowanie po zalogowaniu

exit();

} else {

$blad = "Proszę podać imię!";

}

}

?>

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Logowanie</title>

</head>

<body>

<h2>Logowanie</h2>

<?php if (isset($blad)) echo "<p style='color:red;'>$blad</p>"; ?>

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

Imię: <input type="text" name="imie"><br><br>

<input type="submit" value="Zaloguj">

</form>

</body>

</html>

### ✅ ****Plik 2:**** welcome.php ****– strona powitalna****

php

<?php

session\_start(); // Ponownie uruchamiamy sesję

// Sprawdzamy, czy imię zostało zapisane w sesji

if (isset($\_SESSION["imie"])) {

$imie = $\_SESSION["imie"];

} else {

header("Location: login.php"); // Jeśli nie ma sesji – wracamy do logowania

exit();

}

?>

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Witaj</title>

</head>

<body>

<h2>Witaj, <?php echo $imie; ?>!</h2>

<p>Miło Cię znowu widzieć.</p>

<!-- Dodaj przycisk wylogowania -->

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

<input type="submit" value="Wyloguj">

</form>

</body>

</html>

### ✅ ****Plik 3 (opcjonalny):**** logout.php ****– wylogowanie użytkownika****

php

<?php

session\_start();

session\_unset(); // Czyści wszystkie dane sesji

session\_destroy(); // Zniszczenie sesji

header("Location: login.php"); // Powrót do logowania

exit();

## 🔍 ****Wyjaśnienie ważnych elementów:****

| **Funkcja** | **Opis** |
| --- | --- |
| session\_start() | Startuje sesję (musi być na początku pliku!) |
| $\_SESSION["imie"] | Zmienna sesyjna – zapamiętuje dane użytkownika |
| header("Location: ...") | Przekierowanie użytkownika na inną stronę |
| htmlspecialchars() | Zabezpieczenie przed wstrzyknięciem kodu HTML/JS |
| session\_unset() / session\_destroy() | Czyści sesję przy wylogowaniu |

## 💡 ****Pomysły na rozwinięcie:****

1. Dodaj **czas trwania sesji** (limit ważności).
2. Dodaj ciasteczko setcookie("ostatnie\_logowanie", ...) przy zalogowaniu.
3. Zapisz imię użytkownika do pliku (fwrite()) – np. do logów:

php

file\_put\_contents("logi.txt", "$imie zalogował się o " . date("Y-m-d H:i:s") . "\n", FILE\_APPEND);

## ****Ćwiczenie: System notatek zapisujący dane w MySQL****

### 🎯 ****Cel ćwiczenia:****

* Poznać, jak łączyć się z bazą danych w PHP (mysqli lub PDO)
* Nauczyć się wykonywać operacje **CRUD**: Create, Read, Update, Delete
* Stosować **podstawowe polecenia SQL**
* Zapisywać i odczytywać notatki użytkownika z bazy

## 🧱 ****Struktura ćwiczenia****

Składa się z 4 głównych plików:

1. config.php – połączenie z bazą
2. index.php – formularz + wyświetlenie notatek
3. dodaj.php – zapis nowej notatki
4. usun.php – usunięcie notatki

## 🧰 ****1. Baza danych i tabela – SQL****

Wykonaj to w **phpMyAdmin** lub MySQL CLI:

sql

CREATE DATABASE notatki\_db;

USE notatki\_db;

CREATE TABLE notatki (

id INT AUTO\_INCREMENT PRIMARY KEY,

tresc TEXT NOT NULL,

data\_dodania TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

## 🧩 ****2. Połączenie z bazą –**** config.php

php

<?php

$host = "localhost";

$user = "root";

$pass = "";

$db = "notatki\_db";

$conn = new mysqli($host, $user, $pass, $db);

if ($conn->connect\_error) {

die("Błąd połączenia: " . $conn->connect\_error);

}

?>

## 📝 ****3. Formularz + wyświetlanie notatek –**** index.php

php

<?php include "config.php"; ?>

<!DOCTYPE html>

<html lang="pl">

<head>

<meta charset="UTF-8">

<title>Notatki</title>

</head>

<body>

<h2>Dodaj notatkę</h2>

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

<textarea name="tresc" rows="4" cols="50" required></textarea><br><br>

<input type="submit" value="Zapisz">

</form>

<hr>

<h2>Twoje notatki:</h2>

<?php

$sql = "SELECT \* FROM notatki ORDER BY data\_dodania DESC";

$wynik = $conn->query($sql);

if ($wynik->num\_rows > 0) {

while ($row = $wynik->fetch\_assoc()) {

echo "<div style='border:1px solid #ccc; margin:10px 0; padding:10px;'>";

echo "<p>" . htmlspecialchars($row["tresc"]) . "</p>";

echo "<small>Dodano: " . $row["data\_dodania"] . "</small><br>";

echo "<a href='usun.php?id=" . $row["id"] . "'>Usuń</a>";

echo "</div>";

}

} else {

echo "<p>Brak notatek.</p>";

}

?>

</body>

</html>

## ➕ ****4. Zapis notatki do bazy –**** dodaj.php

php

<?php

include "config.php";

if ($\_SERVER["REQUEST\_METHOD"] == "POST" && !empty($\_POST["tresc"])) {

$tresc = $conn->real\_escape\_string($\_POST["tresc"]);

$sql = "INSERT INTO notatki (tresc) VALUES ('$tresc')";

$conn->query($sql);

}

header("Location: index.php");

exit();

## ❌ ****5. Usuwanie notatki –**** usun.php

php

<?php

include "config.php";

if (isset($\_GET["id"])) {

$id = (int) $\_GET["id"];

$sql = "DELETE FROM notatki WHERE id = $id";

$conn->query($sql);

}

header("Location: index.php");

exit();

## 🔍 ****Wyjaśnienie ważnych elementów:****

| **Część** | **Opis** |
| --- | --- |
| mysqli | Używamy do połączenia i zapytań |
| real\_escape\_string() | Zabezpiecza dane użytkownika przed SQL Injection |
| htmlspecialchars() | Zapobiega XSS – wyświetla dane w bezpieczny sposób |
| header("Location: ...") | Przekierowuje po zapisie/usunięciu |