

## ПРАКТИЧНА РОБОТА №2

**Тема:** Побудова складних логічних умов та конструкцій match/switch.

**Мета:** Застування конструкцій розгалуження для управління обчисленнями.

Освоїти зчитування даних з консолі терміналу.

**Програмне забезпечення:** Локальний сервер Open Server 6.0 та вище, текстовий редактор VS Code, Notepad++.

### ТЕОРЕТИЧНІ ВІДОМОСТИ

#### Умовні оператори if, else, elseif, switch

##### Конструкція if

Вказані дії виконуються тільки тоді, коли умова є істинною

```
if (умова) {           if ($index > 0) {  
    Дія;             echo 'Index > 0';  
}                   }
```

##### Конструкція if...else

Якщо умова істинна, виконується дія із блоку if, в протилежному випадку — з блоку else.

```
if (умова) {           if ($index > 0) {  
    Дія;             echo 'Так';  
} else {               } else {  
    Дія;             echo 'Ні';  
}
```

##### Конструкція elseif

Якщо умова блоку if істина, виконаються дії блоку if. В іншому випадку, якщо умова блоку elseif істина, виконаються дії блоку elseif. У всіх інших випадках виконається дії з блоку else.

```
if (умова) {           if ($numb= 5 && $numb = 10)  
    Дія;             $discount = 5;  
} else {                } else {  
    Дія;             $discount =10;  
}
```

Конструкція switch Якщо значення змінної відповідає значенню одного з блоків case, виконаються дії з цього блоку. В іншому випадку - з блоку default.

```
switch (Змінна) {  
    case Значення 1:  
        Дія 1;  
        break;
```

```
case Значення 2:
    Дія 2;
    break;
[default: Дія;]
}
switch ($day) {
    case 1:
        echo 'Понеділок'; break;
    case 2:
        echo 'Вівторок'; break;
    case 3:
        echo 'Середа'; break;
    case 4:
        echo 'Четвер'; break;
    case 5:
        echo 'П'ятниця'; break;
    case 6:
        echo 'Субота'; break;
    case 7:
        echo 'Неділя'; break;
default:
    echo 'Немає такого дня';
```

Приклад. Приклад використання match з операторами порівняння

```
<?php
$age = 18;

$output = match (true) {
    $age < 2 => "Baby",
    $age < 13 => "Child",
    $age <= 19 => "Teenager",
    $age >= 40 => "Old adult",
    $age > 19 => "Young adult",
};

var_dump($output);
?>
```

## Зчитування даних для скрипту

### *fgets(STDIN) в PHP*

Опис:

Функція `fgets(STDIN)` використовується для зчитування введених користувачем даних з консолі (терміналу).

```
$input = fgets (STDIN) ;
```

`STDIN` (Standard Input) — стандартний потік введення, який читає дані з клавіатури.

`fgets()` — функція, яка читає рядок тексту (до символу нового рядка `\n` або кінця файлу EOF).

*Особливості:*

- Читає один рядок тексту.
- Включає символ нового рядка `\n` в кінці введення (якщо натиснута Enter).
- Використовується в CLI-скриптах (у командному рядку чи терміналі), але не працює у веб-браузері.
- Щоб прибрати зайві пробіли та символи нового рядка, часто використовують `trim(fgets(STDIN))`.

*Приклад використання:*

1. Зчитування введеного рядка

```
<?php  
echo "Введіть ваше ім'я: ";  
$name = trim(fgets(STDIN)); // Видаляємо зайві пробіли  
та новий рядок  
echo "Привіт, $name! \n";  
?>
```

Вхідні дані:

Введіть ваше ім'я: Олександр

Вивід:

Привіт, Олександр!

### **Зчитування числа**

Для зчитування чисел проводять перетворення даних (`int`) або (`float`)

```
<?php  
echo "Введіть число: ";  
$number = (int) fgets(STDIN); // Перетворюємо введені  
дані на число  
echo "Ваше число: $number\n";  
?>
```

Вхідні дані:

Введіть число: 42

Вивід:

Ваше число: 42

### **Коли fgets(STDIN) корисний?**

При написанні консольних PHP-скриптів.

Коли потрібно взаємодіяти з користувачем у терміналі.

Обмеження та альтернативи

✗ Не працює у веб-браузері!

### **Висновок:**

fgets(STDIN) — це простий і зручний спосіб отримати ввід від користувача в терміналі під час виконання PHP-скриптів.

## **ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВИКОНАННЯ**

### **1. Перевірка парності числа**

Напишіть програму, яка запитує у користувача ціле число та перевіряє, чи воно парне чи непарне.

- Якщо число парне, виведіть "Число парне"
- Якщо непарне — "Число непарне"

**Підказка:** Використовуйте оператор % для перевірки залишку від ділення на 2.

### **2. Визначення оцінки за балами**

Користувач вводить число від 0 до 100 (бал за тест). Напишіть програму, яка визначає оцінку за такими критеріями:

- **90-100** – "Відмінно"
- **70-89** – "Добре"
- **50-69** – "Задовільно"
- **0-49** – "Незадовільно"

**Підказка:** Використовуйте if-elseif-else.

### **3. Перевірка високосного року**

Напишіть програму, яка запитує у користувача рік і визначає, чи є він високосним.

Рік є високосним, якщо:

- Він кратний 400
- Або кратний 4, але не кратний 100

Якщо рік високосний, виведіть "Рік високосний", інакше "Рік не високосний".

**Підказка:** Використовуйте && та || для логічних перевірок.

#### **4. Калькулятор простих операцій**

Користувач вводить два числа та оператор (+, -, \*, /). Програма повинна виконати відповідну операцію.

- Якщо користувач вводить / і друге число **0**, вивести "Ділення на нуль неможливе"

- Інакше обчислити результат та вивести його

**Підказка:** Використовуйте switch.

#### **5. Перевірка довжини рядка**

Користувач вводить текст. Програма повинна перевірити довжину рядка:

- Якщо довжина менше 5 символів – вивести "Короткий текст"
- Якщо від 5 до 20 символів – "Нормальний текст"
- Якщо більше 20 – "Дуже довгий текст"

**Підказка:** Використовуйте strlen().

#### **6. Створити скрипт для обчислення значення у за формулою.**

$$1) y = \begin{cases} \sqrt{|x+1|} & \text{при } x < 0 \\ \sqrt[3]{x^2 + 3} & \text{при } x \geq 0 \end{cases} \quad 2) y = \begin{cases} \sin^2 x & \text{при } x < 0 \\ \sqrt{x} & \text{при } 0 \leq x < 4 \\ \cos x^3 & \text{при } x \geq 4 \end{cases}$$

#### **Додаткове завдання.**

Дано чотири цілих числа. Визначити скільки з них парні. (Запропонувати спосіб без використання оператора розгалуження).