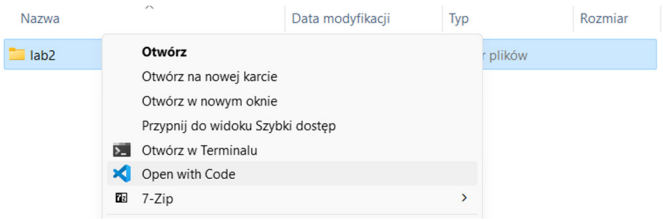
Aplikacje internetowe 1 (AI1) – laboratorium nr 2

Podstawy języka PHP 8

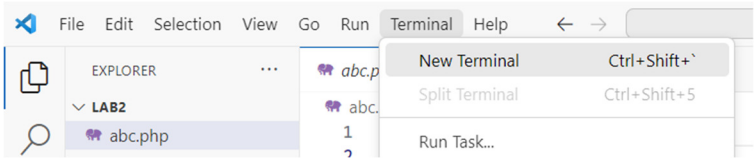
Początek laboratorium:

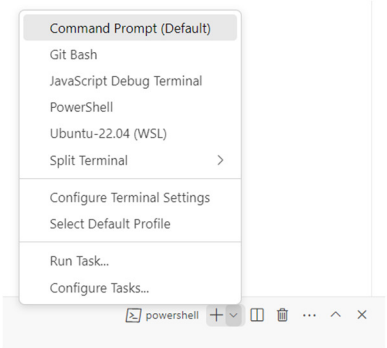
• pobrać na pulpit archiwum `Lab002_AI1_start.zip` w którym umieszczone są pliki potrzebne do wykonania zadań oraz rozpakować to archiwum,

• przejść do rozpakowanego folderu oraz otworzyć folder `lab2` w *VSCode*,

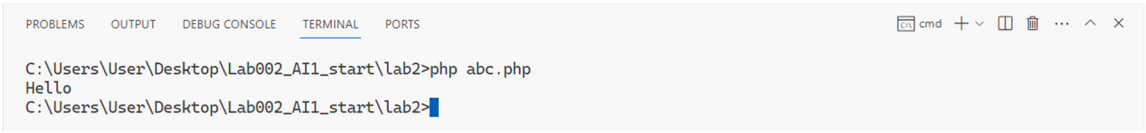


• otworzyć terminal, wybrać *CMD* (nie *PowerShell*),





• wykonać skrypt z pliku *abc.php*, oczekiwany rezultat to wyświetlenie `Hello`,



• każde z poniższych zadań rozwiązywać w plikach *zadX.php*, gdzie *X* to numer zadania (*2.X*) oraz uruchamiać skrypty według powyższego sposobu,

• wspomagać się dokumentacją *PHP*: https://www.php.net/manual/en/index.php,

• zadania/fragmenty zadań oznaczone * należy pominąć, będą do samodzielnego dokończenia w przypadku braku wystarczającej ilości czasu na laboratorium,

Zadania (*PHP 8*):

Zadanie 2.1:
Używając jednej instrukcji `print` wyświetlić na konsoli następujący napis:

```
'Hello world!' znaczy "Witaj świecie!".
       Jest to napis pojawiający się w pierwszych programach.
```

```
C:\Users\User\Desktop\Lab002_AI1_start\lab2>php zad1.php
'Hello world!' znaczy "Witaj świecie!".
       Jest to napis pojawiający się w pierwszych programach.
```

Zadanie 2.2:
Utworzyć:
• zmienną `a` i przypisać do niej wartość `4`,
• stałą `B` z wartością `10`,
• zmienną `c` z `4.0`,
• `d` z `5.667`.
Wypisać na konsoli zdanie z informacją o wykonywanej operacji oraz/lub jej wynik:
• dodawania `a` i `B`,
• dzielenia a przez B *,
• `a` do potęgi `B` *,
• resztę z dzielenia `B` przez `a` *,
• czy `a` ma taką samą wartość jak `B` *,
• czy wartość `a` jest większa od `B`,
• czy wartość `a` jest większa od `B` (używając operatora trójargumentowego) *,
• czy `a` i `c` mają taką samą wartość,
• czy `a` i `c` mają taką samą wartość (z uwzględnieniem typu),
• `d` bez części po przecinku,
• `d` zaokrąglone do 2 miejsca po przecinku *.

```
Suma … i … wynosi …
…
Są różne./Są równe.
Jest większa./Jest mniejsza./Są równe.
```

`define, PHP_EOL, round`

Zadanie 2.3: *
Utworzyć `a` oraz `B` z wartościami jak w poprzednim zadaniu.
Spróbować przypisać do nich nowe wartości: `7` do `a` oraz `22` do `B`.
Wyjaśnić zaistniałą sytuację w komentarzu.

–

Zadanie 2.4:
Obecne są dwie zmienne przechowujące napisy, zawierające nadmiarowe spacje:

```
$text1 = "   Programuję dobrze  ";
$text2 = "dobrze w PHP.  ";
```

Następnie wyświetlić na konsoli:
• długość zmiennej `text1`,

• zmienną `text2` w odwrotnej kolejności (od tyłu) *,
• która zmienna zawiera dłuższy ciąg znaków? *,
• czy zmienna `text1` zawiera słowo `Programuję`?,
• czy zmienna `text2` zaczyna się słowem `dobrze`? *,
• połączone zmienne `text1` oraz `text2` z wcześniej usuniętymi nadmiarowymi spacjami,
• rezultat z powyższej pauzy umieścić w zmiennej `text3` podzielić według separatora będącym spacją, oraz wyświetlić powstałą tablicę *,
• zmienną `text1` ze zmienionym słowem `dobrze` na `źle` *,
• na którym indeksie (pozycji) zaczyna się słowo *PHP* w `text2`?,
• zmienną `text1` z wszystkimi literami zmienionymi na duże,
• zmienną `text2` z pierwszą literą zmienioną na dużą *,
• zmienną `text2` w zakresie od 9 do 11 pozycji włącznie *.

```
mb_strlen, strrev, str_contains, str_starts_with, trim, explode, str_replace,
                mb_strpos, mb_strtoupper, ucfirst, mb_substr
```

Zadanie 2.5: *
Obecna jest zmienna `n` (ma wartość `3.5`) oraz zmienna `note` (nie ma żadnej wartości). Zastosować konstrukcję warunkową `switch`, która na podstawie przyjętej wartości (w zakresie `2.0`, …, `5.0`) przypisze do zmiennej `note` słowną nazwę oceny.
W przypadku podania wartości spoza zakresu przypisać do zmiennej `note` pusty ciąg znaków.
Wykonać to zadanie jeszcze raz, używając konstrukcji `match`.

```
niedostateczny
...
bardzo dobry
```

Zadanie 2.6:
Utworzyć własną funkcję o nazwie `ctf`, której zadaniem będzie przeliczać stopnie *Celsjusza* na *Fahrenheita*, przyjmującą jeden opcjonalny parametr typu *float* o nazwie `c`, z domyślną wartością `null`.
W przypadku wywołania funkcji:
• bez parametru ma być wyświetlony komunikat „*Nie podano wartości*” i zwrócony `null`,
• podania wartości parametru ma być zwracana przeliczona wartość w *Fahrenheitach* (także *float*), uwzględnić poprawne działanie funkcji dla 0°C.

```
°F = (°C × 9/5) + 32
```

Zadanie 2.7: *
Zmienna `l` ma wartość `10`.
Utworzyć własną funkcję `rd`, która zmieni wartość tej zmiennej na liczbę wylosowaną z przedziału 1-50.

```
random_int
```

Zadanie 2.8: *
Wypisać liczby od 0 do 100 z krokiem co 5 (0, 5, 10, …, 95, 100), z wyłączeniem tych które są podzielne przez 7.

–

Zadanie 2.9:
Utworzyć tablicę o nazwie `fruits` z kluczami numerycznymi zawierającą wymienione poniżej nazwy owoców. Wykonać poniższe pauzy:
• wyświetlić liczbę elementów w tablicy,
• wyświetlić wszystkie elementy tablicy, każdy w osobnej linijce,
• dodać cytrynę na koniec tablicy,
• usunąć ostatni element tablicy,
• wyświetlić tablicę posortowaną malejąco *.

```
"banana", "apple", "strawberry", "grape", "orange", "watermelon", "blueberry"

count, array_push, print_r, array_pop, sort
```

Zadanie 2.10:
Utworzyć tablicę o nazwie `people` z kluczami, które są ciągami znaków, będącymi imionami osób, a wartościami będzie aktualny wiek danej osoby. Utworzyć tablicę z przykładowymi osobami. Następnie:
• wyświetlić wszystkie elementy, każdy w osobnej linijce, w postaci jak w poniżej,
• wyświetlić liczbę elementów w liście,
• wyświetlić wiek pana *Bartosza*,
• dodać pana *Witolda* mającego 28 lat,
• usunąć pana *Piotra*,
• wyświetlić tablicę posortowaną malejąco według wieku osób.

```
Jan ma 45 lat, Bartosz ma 38 lat, Piotr ma 40 lat

count, unset, arsort, print_r
```

Zadanie 2.11: *
Utworzyć własną funkcję o nazwie `division`, której zadaniem będzie zwracać wynik dzielenia dwóch liczb całkowitych (przyjmowanych jako parametry `x`, `y`). Przetestować funkcję dla kilku zestawów różnych parametrów.
Następnie:
• wewnątrz funkcji zgłosić problem, gdy drugi z parametrów jest wartością 0 (jest to nieprawidłowa wartość dla dalszego działania funkcji, prowadząca do dzielenia przez 0),
• w miejscu wywołania funkcji obsłużyć sytuację, gdyby drugi z parametrów był wartością 0,
• wewnątrz funkcji zgłosić problem, gdy któryś z parametrów ma typ inny niż `int` (jest to nieprawidłowa wartość dla dalszego działania funkcji, funkcja ma tylko przeprowadzać dzielenie liczb całkowitych),
• w miejscu wywołania funkcji obsłużyć sytuację, gdyby któryś z parametrów był wartością inną niż całkowita.

```
InvalidArgumentException, TypeError, getMessage
```

Zadanie 2.12: *
Wyświetlić:
• obecną datę w formacie: `Thursday, 06-03-2025`,
• obecną datę i godzinę w formacie: `2024-March-06 10:10`,
• liczbę dni pomiędzy dniem dzisiejszym a *12 marca 2022 roku*,
• liczbę godzin i minut pomiędzy aktualną godziną a 7:00 dnia *dzisiejszego*,
• która data jest wcześniejsza: data dzisiejsza czy *1 kwietnia 2024 roku*.

```
date, mktime, strtotime, new DateTime(…), getTimestamp, date_diff, setTime, format
```

Zadanie 2.13:
Zaimplementować klasę **Point** pozwalającą na reprezentowanie punktów w płaszczyźnie 2D (współrzędna x i y). Klasa powinna dostarczyć funkcjonalności:
• możliwość wypisania informacji o punkcie w postaci **Point(…,…)**, gdzie … to wartości współrzędnych,
• możliwość zaktualizowania punktowi danej współrzędnej,
• przesunięcia punktu o podany dystans.
Następnie utworzyć kilka przykładowych punktów oraz przetestować działanie wszystkich składowych klasy.

```
class, __construct, set…, get…, __toString, new …(…)
```

Zadanie 2.14: *
Zaimplementować klasę **Dog** reprezentującą psy ze schroniska, każdy z nich ma *uniwersalny identyfikator* (*v4*), *nazwę*, *wiek*, *datę przyjęcia*. Klasa ma dostarczyć funkcjonalność wypisania informacji o obiekcie (w postaci przedstawionej poniżej).
Do tworzenia losowych identyfikatorów doinstalować *Composerem* pakiet *ramsey/uuid* (wykonać poniższą komendę).
Następnie utworzyć 5 przykładowych psów, które dodać do listy oraz przetestować ich wyświetlanie.
Uwaga: *VSCode* może nietrafnie podkreślać niektóre linijki.

```
composer require ramsey/uuid
```

```
Burek (9 l.) przyjęty w dn. 10-03-2025
Clifford (9 l.) przyjęty w dn. 05-02-2025
Azor (12 l.) przyjęty w dn. 15-02-2025
Szarik (8 l.) przyjęty w dn. 22-02-2025
Idefix (15 l.) przyjęty w dn. 26-01-2025
```

```
Uuid::uuid4()
```

* – zadania/podpunkty do samodzielnego dokończenia/wykonania,

* – zadania/podpunkty dla zainteresowanych.

Po zakończonym laboratorium należy skasować wszystkie pobrane oraz utworzone przez siebie pliki z komputera w sali laboratoryjnej.