

2. Tablice

Tablice są bardzo specyficznym typem zmiennych – są to, najprościej mówiąc, zmienne zawierające w sobie uporządkowany zbiór zmiennych. Do zmiennych tych uzyskuje się dostęp przez liczbę w nawiasie kwadratowym podane bezpośrednio po nazwie zmiennej – tablicy. Liczba ta to tak zwany indeks – numer kolejny zmiennej w tablicy. Tak samo przypisuje się wartość do tablicy.

Przykład 2.1. Tworzenie tablicy

```
<?php
```

```
$tablica[0] = "Wpis numer 0";  
$tablica[1] = "Wpis numer 1";  
$tablica[2] = "Wpis numer 2";
```

```
echo $tablica[2]; // Wyświetlony zostanie napis "Wpis numer 2";
```

```
?>
```

Aby po prostu dodać kolejny wpis na końcu tabeli wystarczy przy przypisywaniu wartości nie wpisywać indeksu do nawiasów kwadratowych. Jeśli w ten sposób dodawane są wpisy do nowej tablicy, to pierwszy wpis ma indeks 0.

Indeks można też podawać ze zmiennej, z innej tablicy czy funkcji – z dowolnego wyrażenia zwracającego wartość.

Przykład 2.2. Indeksy tablic

```
<?php
```

```
$tab1[] = 1;  
$tab1[] = 0;  
$tab1[] = 3;  
$tab1[] = 2;  
  
$tab2[] = "Pierwszy";  
$tab2[] = "Drugi";  
$tab2[] = "Trzeci";  
$tab2[] = "Czwarty";
```

```
echo $tab2[$tab1[2]];
```

```
?>
```

Elementem tablicy może być każdy typ zmiennej (z innymi tablicami i obiektami włącznie).

Innym ze sposobów deklaracji tablic jest użycie słowa kluczowego Array:

Przykład 2.3.

```
<?php  
  
$owoce = array („mango", „ papaja", „banan", "aronia");  
  
var_dump( $owoce );  
  
?>
```

Przykład 2.4.

```
<?php  
  
$owoce = array ("d"=>"mango", "a"=>"papaja", "b"=>"banan", "c"=>"aronia");  
  
var_dump( $owoce );  
  
?>
```

Przykład 2.5.

```
<?php  
  
$imiona = Array('Marcin', 100 => 'Daniel', 200 => 'Magda', 'Paulina');  
  
echo $imiona[0]; // wyświetli Marcin  
  
echo $imiona[200]; // wyświetli Magda  
  
$imiona[201] = 'Katarzyna'; // zmiana wartości  
  
echo $imiona[201]; // wyświetli Katarzyna  
  
?>
```

2.2 Tablica asocjacyjna

W PHP występuje też inny rodzaj tablic, tak zwane tablice asocjacyjne (zwane też czasem haszami – hash table). Są to tablice, w których zamiast indeksów liczbowych używa się identyfikatorów znakowych (kluczy):

Przykład 2.6. Tablice asocjacyjne

```
<?php  
$tablica["imie"] = "Jan";  
$tablica["nazwisko"] = "Kowalski";  
$tablica["adres"] = "Polna 1";  
echo $tablica["imie"]." ".$tablica["nazwisko"].", ul. ".$tablica["adres"]."\n";  
?>
```

Przykład 2.7. Tablice asocjacyjne

```
<?php

// tworzymy tablicę asocjacyjną

$tablica = array("imie" => "Jan", "nazwisko" => "Kowalski", "email" => "jankowal@gmail.com");

// Wyświetlamy

foreach ($tablica as $klucz => $dana)
{
    echo 'Klucz to <b>'. $klucz. '</b> a jego wartość to <b>'. $dana. '</b>';
}
?>
```

2.3 Tablice wielowymiarowe

PHP umożliwia także deklaracje tzw. tablic wielowymiarowych. Polega to na tworzeniu kolejnych węzłów elementu dając przy tym wrażenie drzewa elementów. *Tablice* wielowymiarowe można deklarować podając kolejne indeksy w nawiasach kwadratowych:

```
<?

$dane[0]['imie'] = 'Jan';

$dane[0]['nazwisko'] = 'Kowalski';

$dane[0]['ulica'] = 'Kowalowska';

$dane[1]['imie'] = 'Maciej';

$dane[1]['nazwisko'] = 'Nowak';

$dane[1]['ulica'] = 'Nowakowska';

print_r($dane);

?>
```

2.4 Tworzenie ciągów z tablic i odwrotnie

PHP umożliwia zamianę ciągów na tablice i odwrotnie. Zamiana ciągu na tablicę jest bardzo przydatna jeśli zachodzi potrzeba wyciągnięcia jakiegoś fragmentu danych z ciągu. Załóżmy że w odczytaliśmy z pliku z danymi (o odczycie z plików w jednym z kolejnych rozdziałów) linię z logu zapisanego przez licznik WWW: "12/11/2000;19:23:33;Netscape Navigator;192.168.1.1". Jak widać dane rozdzielone są średnikami. Do rozdzielania ciągów na tablicę służy funkcja `explode()`. Jako pierwszy parametr trzeba do niej podać znak lub dłuższy ciąg który oddziela kolejne pola, jako drugi ciąg do rozdzielania. Opcjonalnie można podać trzeci argument, który oznacza maksymalną liczbę pól – jeśli jest ich więcej niż ta liczba, to ostatnie pole będzie zawierało wszystkie pozostałe pola. Funkcja zwraca tablicę zawierającą kolejne pola.

Przykład 2.8.

```
<?php
```

```
$dane = "12/11/2000;19:23:33;Netscape Navigator;192.168.1.1";  
$tablica = explode(";", $dane);
```

```
?>
```

Jest także rozszerzona wersja funkcji `explode`: `split()`. Różni się ona tym, że zamiast prostego ciągu znaków rozdzielających pola, akceptuje ona wyrażenia regularne.

Czasem potrzebne jest działanie w drugą stronę: złącznie pól tablicy w jeden ciąg, w którym pola oddzielone są jakimś znakiem (lub kilkoma). Do tego służy funkcja `implode()`. Jako pierwszy parametr podawany jest ciąg za pomocą którego “sklejane” są elementy tablicy, a jako drugi właśnie tablica do posklejania. Zwracany jest ciąg zawierający posklejane elementy. Jako przykład zastosowania może posłużyć właśnie zapisywanie danych o użytkowniku w aplikacji licznika odwiedzin – tablica zawiera dane o odwiedzającym, a potrzebny jest ciąg pooddzielany średnikami.

Przykład 2.9.

```
<?php
```

```
$dane = implode(";", $tablica);
```

```
?>
```

2.5 Funkcje tablic - dodawanie nowych elementów**array_push**

Wstawia jeden lub więcej elementów na koniec tablicy

Opis

`int array_push (array &tablica, mixed wartosc [, mixed ...])`

array_push() traktuje zmienną *tablica* jako stos i wstawia przekazane parametry na koniec podanej tablicy. Długość parametru *tablica* zwiększa się o liczbę przekazanych wartości.

Przykład 2.10

```
<?php
```

```
$stos = array("pomarańcza", "banan");  
array_push($stos, "jabłko", "malina");  
print_r($stos);
```

```
?>
```

Po wykonaniu powyższego przykładu zmienna *\$stos* będzie miała następujące elementy:

```
Array
(
    [0] => pomarańcza
    [1] => banan
    [2] => jabłko
    [3] => malina
)
```

array_unshift

Wstawia jeden lub więcej elementów na początek tablicy

Opis

int array_unshift (array &tablica, mixed wartość [, mixed ...])

array_unshift() wstawia jeden lub więcej przekazanych jako parametry elementów na początek tablicy *tablica*. Zauważ, że lista elementów wstawiana jako całość, więc elementy zostają w takim samym porządku. Wszystkie klucze liczbowe zostaną zmodyfikowane tak, aby ich wartości zaczynały się od zera, podczas gdy klucze znakowe nie zostaną zmienione.

Funkcja zwraca nową liczbę elementów w tablicy *tablica*.

Przykład 2.11

```
<?php
$kolejka = array ("pomarańcza", "banan");
array_unshift ($kolejka, "jabłko", "malina");
?>
```

Zmienna *\$kolejka* będzie miała następujące elementy:

```
Array
(
    [0] => jabłko
    [1] => malina
    [2] => pomarańcza
    [3] => banan
)
```

2.6 Funkcje tablic - usuwanie elementów

array_pop

Zdejmuje element z końca tablicy

Opis

mixed **array_pop** (array &tablica)

array_pop() zdejmuję i zwraca ostatnią wartość tablicy *tablica*, skracając tą tablicę o jeden element. Jeśli *tablica* jest pusta (lub nie jest tablicą), zwracana jest wartość **NULL**.

Przykład 2.12

```
<?php
$stos = array("pomarańcza", "banan", "jabłko", "malina");
$owoc = array_pop($stos);
print_r($stos);
?>
```

Po wykonaniu powyższego kodu *\$stos* będzie miał tylko trzy elementy:

```
Array
(
    [0] => pomarańcza
    [1] => banan
    [2] => jabłko
)
```

a *malina* będzie przypisana do zmiennej *\$owoc*.

array_shift

Usuwa element z początku tablicy

Opis

mixed **array_shift** (array &tablica)

array_shift() usuwa pierwszą wartość parametru *tablica* i zwraca go skracając tą tablicę o jeden element przesuwając wszystkie pozostałe elementy w dół. Wszystkie klucze liczbowe zostaną zmodyfikowane tak, aby ich wartości zaczynały się od zera, podczas gdy klucze znakowe nie zostaną zmienione. Jeśli *tablica* jest pusta (lub nie jest tablicą), zwracana jest wartość **NULL**.

Notatka: Ta funkcja **zresetuje()** wskaźnik tablicy po swoim wykonaniu.

Przykład 2.13

```
<?php
$stos = array ("pomarańcza", "banan", "jabłko", "malina");
$owoc = array_shift ($stos);
?>
```

Zmienna *\$stos* będzie miała 3 elementy:

```
Array  
(  
    [0] => banan  
    [1] => jabłko  
    [2] => malina)
```

a *pomarańcza* będzie przypisana do zmiennej *\$owoc*

Funkcja **unset()**;
pozwala usunąć pojedynczą, lub więcej zmiennych, lub element tablicy.

```
<?php  
unset($owoc[1]); // usunięcie pojedynczej zmiennej  
unset($owoc[0], $owoc[3], $owoc[5]); // usunięcie wielu zmiennych  
?>
```

2.7 Funkcje tablic - losowanie

rand

Generuje losową liczbę stałoprzecinkową

int **rand** ([int \$min [, int \$max]])

Jeśli wywołana bez opcjonalnych argumentów *min* i *max*, funkcja **rand()** zwraca pseudolosową liczbę stałoprzecinkową z przedziału pomiędzy 0 a **RAND_MAX**. Dla uzyskania liczby losowej z przedziału np. od 5 do 15 (włącznie), należy wywołać *rand(5,15)*.

Przykład 2.14

```
<?php  
echo rand() . "\n";  
echo rand() . "\n";  
  
echo rand(5, 15);  
?>
```

Powyższy przykład wyświetli coś podobnego do:

```
7771  
22264  
11
```

2.8 Sortowanie tablic

PHP oferuje cały zestaw funkcji służących do sortowania tablic. Są to:

sort() - sortuje zwykłe tablice (nie asocjacyjne) w kolejności alfabetycznej

rsort() - sortuje zwykłe tablice (nie asocjacyjne) w odwróconej kolejności

asort() - sortuje tablice asocjacyjne zachowując przypisanie kluczy do wartości

arsort() - sortuje w odwrotnej kolejności tablice asocjacyjne zachowując przypisanie kluczy do wartości.

ksort() - sortuje tablice asocjacyjne według kluczy

Przykład 2.15

```
<?php

$owoce = array ("d"=>"mango", "a"=>"papaja", "b"=>"banan", "c"=>"aronia");

asort ($owoce);

echo "<pre>";

print_r ($owoce);

echo "</pre>";

?>
```

Powyższy przykład wyświetli coś podobnego do:

```
Array
(
    [c] => aronia
    [b] => banan
    [d] => mango
    [a] => papaja
)
```

2.9 Spacer po tablicy.

next

Przesuń do przodu wewnętrzny wskaźnik tablicy

Przesuwa wewnętrzny wskaźnik tablicy i jedną pozycję do przodu i zwraca element tablicy aktualnie wskazywany przez wskaźnik, lub **FALSE** jeśli nie ma już więcej elementów.

current

Zwraca bieżący element tablicy

Funkcja **current()** po prostu zwraca element tablicy, na który aktualnie wskazuje wewnętrzny wskaźnik. Nie przesuwa ona wskaźnika. Jeśli wewnętrzny wskaźnik jest poza końcem listy elementów, **current()** zwraca **FALSE**.

prev

Cofnij wewnętrzny wskaźnik tablicy

Zwraca wartość z tablicy z miejsca poprzedniego od tego na które wskazywał wewnętrzny wskaźnik pliku, lub **FALSE** jeśli nie ma już więcej elementów.

reset

Ustaw wewnętrzny wskaźnik tablicy na jej pierwszy element

reset() przewija wewnętrzny wskaźnik tablicy parametru *tablica* na jego pierwszy element i zwraca jego wartość, lub **FALSE** jeśli tablica jest pusta.

Przykład 2.16

```
<?php
```

```
$tablica = array('krok pierwszy', 'krok drugi', 'krok trzeci', 'krok czwarty');
```

```
echo current($tablica) . "<br />\n"; // "krok pierwszy"
```

```
next($tablica);
```

```
next($tablica);
```

```
prev($tablica);
```

```
echo current($tablica) . "<br />\n"; // "krok drugi"
```

```
reset($tablica);
```

```
echo current($tablica) . "<br />\n"; // "krok pierwszy"
```

```
?>
```

2.10 Przeszukiwanie tablicy.**in_array**

Sprawdza czy wartość istnieje w tablicy

`bool in_array (mixed $igła , array $stóg_siana [, bool $ściśle])`

Przeszukuje *stóg_siana* w poszukiwaniu parametru *igła* i zwraca **TRUE** jeśli wartość została znaleziona lub **FALSE** w przeciwnym przypadku.

Przykład 2.17

```
<?php
$a = array(0,1,2,3,4,5);
in_array("0", $a) ;//return true
in_array("3", $a) ;//return true
in_array("8", $a);// return false
?>
```

array_search

Przeszukuje tablicę pod kątem podanej wartości i w przypadku sukcesu zwraca odpowiedni klucz

`mixed array_search (mixed $igła , array $stóg_siana [, bool $ściśle])`

Przeszukuje *stóg_siana* w poszukiwaniu parametru *igła* i zwraca odpowiedni klucz jeśli został on znaleziony lub **FALSE** w przeciwnym przypadku.

Przykład 2.18

```
<?php
$tablica = array(0 => 'niebieski', 1 => 'czerwony', 2 => 'zielony', 3 => 'czerwony');

$klucz = array_search('zielony', $tablica); // $klucz = 2;
$klucz = array_search('czerwony', $tablica); // $klucz = 1;
?>
```

2.10 Transformacje tablic.

array_intersect

Zwraca przecięcie tablic

`array array_intersect (array $tablica1 , array $tablica2 [, array $...])`

array_intersect() zwraca tablicę zawierającą wszystkie wartości tablicy *tablica1* które istnieją we wszystkich argumentach. Zauważ, że zachowywane są przypisania kluczy.

Przykład 2.19

```
<?php
$tablica1 = array ("a" => "zielony", "czerwony", "niebieski");
$tablica2 = array ("b" => "zielony", "żółty", "czerwony");
$wynik = array_intersect ($tablica1, $tablica2);
print_r($wynik);
?>
```

Powyższy przykład wyświetli:

```
Array
(
    [a] => zielony
    [0] => czerwony
)
```

array_diff

Zwraca różnice pomiędzy tablicami

array **array_diff** (array \$tablica1 , array \$tablica2 [, array \$...])

array_diff() zwraca tablicę zawierającą wszystkie wartości tablicy *tablica1* które nie są obecne w innych tablicach-argumentach. Zauważ, że zachowywane są klucze.

Przykład 2.20

```
<?php
$tablica1 = array ("a" => "zielony", "czerwony", "niebieski", "czerwony");
$tablica2 = array ("b" => "zielony", "żółty", "czerwony");
$wynik = array_diff ($tablica1, $tablica2);

print_r($wynik);
?>
```

Wielokrotne wystąpienia w \$tablica1 są traktowane w ten sam sposób. Powyższy przykład wyświetli:

```
Array
(
    [1] => niebieski
)
```

Ćwiczenie 2.1

Przed dodaniem nowych produktów:

```
Array
(  
    [0] => Telewizor  
    [1] => Książka kucharska  
    [2] => Słuchawki  
    [3] => Stół kuchenny  
    [4] => Kawa inka  
)
```

Po modyfikacji (dodanie nowych elementów):

```
Array
(  
    [0] => Telewizor  
    [1] => Książka kucharska  
    [2] => Słuchawki  
    [3] => Stół kuchenny  
    [4] => Kawa inka  
    [5] => Telefon  
    [6] => Plecak  
)
```

Po modyfikacji (usunięcie elementów):

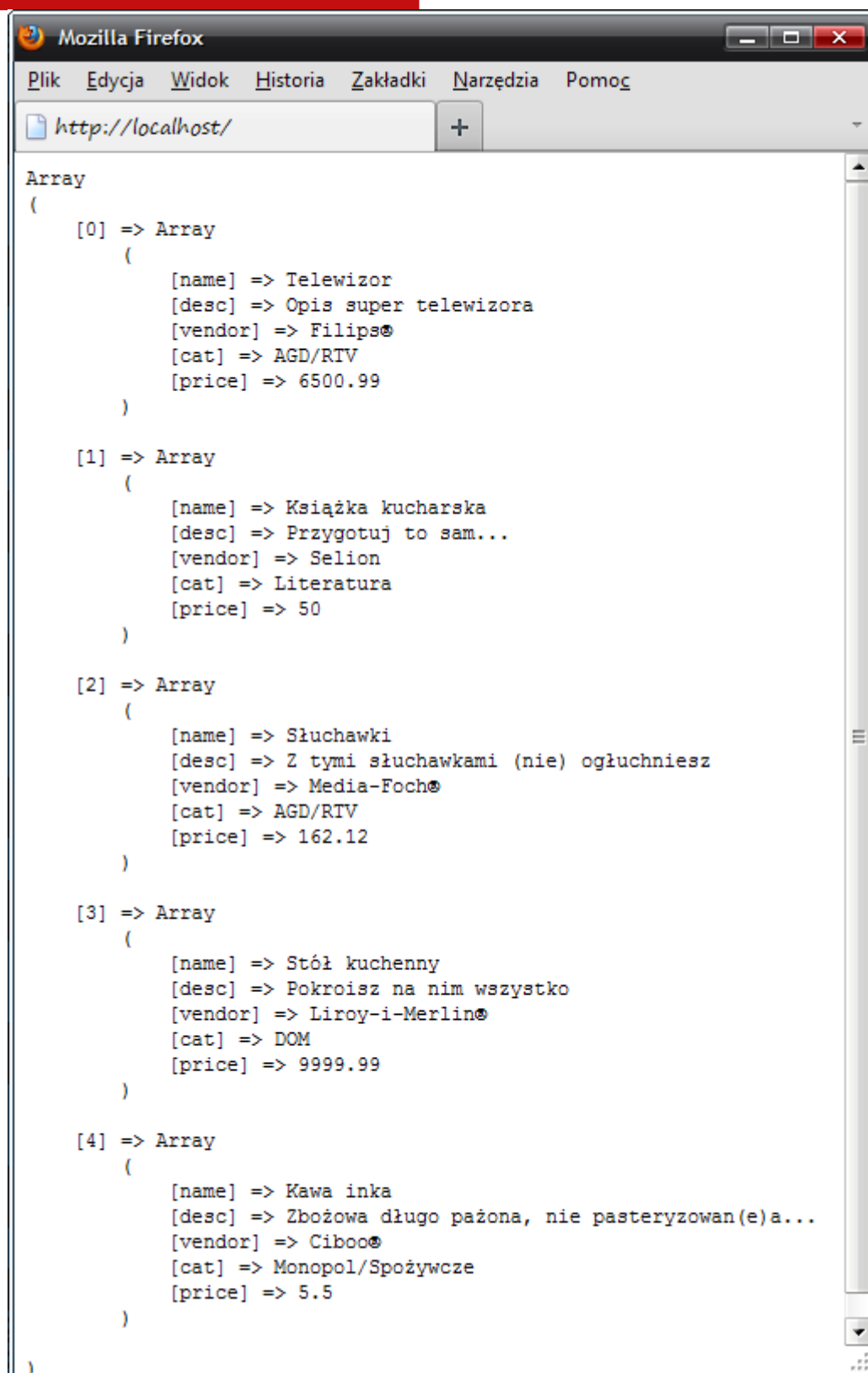
```
Array
(  
    [0] => Telewizor  
    [2] => Słuchawki  
    [4] => Kawa inka  
    [5] => Telefon  
    [6] => Plecak  
)
```

Dynamiczne tworzenie nowej tablicy:

```
To jest@przykładowy@ciąg znakówArray
(  
    [0] => To jest  
    [1] => przykładowy  
    [2] => ciąg znaków  
)
```

Telewizor[#]Słuchawki[#]Kawa inka[#]Telefon[#]Plecak

Ćwiczenie 2.2



The screenshot shows a Mozilla Firefox browser window with the address bar set to `http://localhost/`. The main content area displays a JSON array of five objects, each representing a product. The objects are indexed from 0 to 4. Each object contains the following fields: `name`, `desc`, `vendor`, `cat`, and `price`.

```
Array
(
    [0] => Array
        (
            [name] => Telewizor
            [desc] => Opis super telewizora
            [vendor] => Philips®
            [cat] => AGD/RTV
            [price] => 6500.99
        )

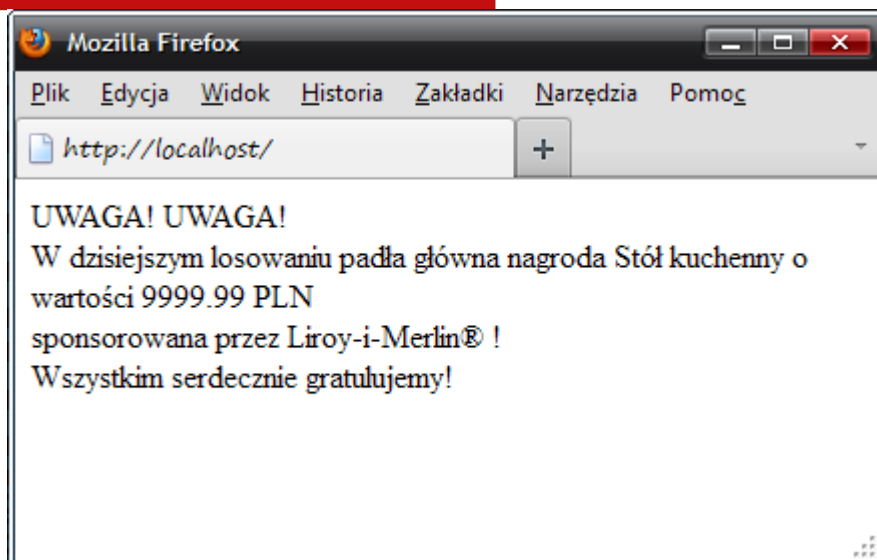
    [1] => Array
        (
            [name] => Książka kucharska
            [desc] => Przygotuj to sam...
            [vendor] => Selion
            [cat] => Literatura
            [price] => 50
        )

    [2] => Array
        (
            [name] => Słuchawki
            [desc] => Z tymi słuchawkami (nie) ogłuchniesz
            [vendor] => Media-Foch®
            [cat] => AGD/RTV
            [price] => 162.12
        )

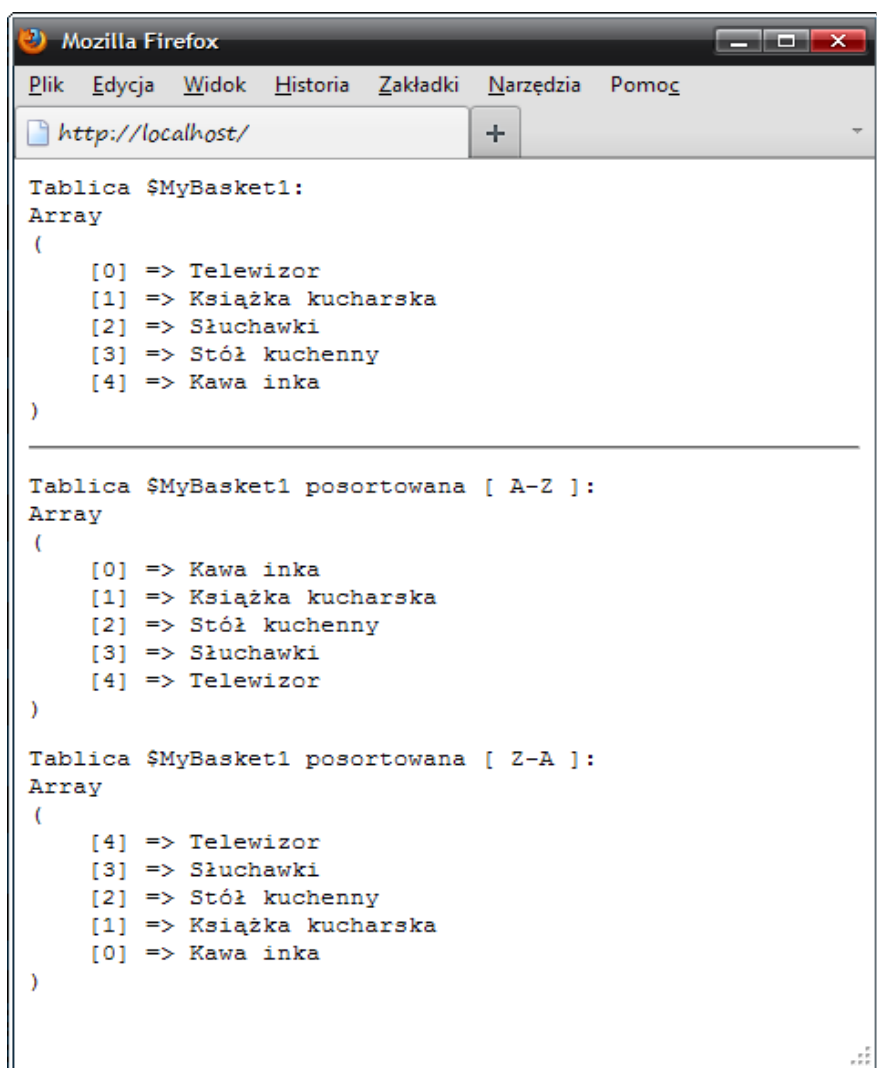
    [3] => Array
        (
            [name] => Stół kuchenny
            [desc] => Pokroisz na nim wszystko
            [vendor] => Liroy-i-Merlin®
            [cat] => DOM
            [price] => 9999.99
        )

    [4] => Array
        (
            [name] => Kawa inka
            [desc] => Zbożowa długo pażona, nie pasteryzowan(e)a...
            [vendor] => Ciboo®
            [cat] => Monopol/Spożywcze
            [price] => 5.5
        )
)
```

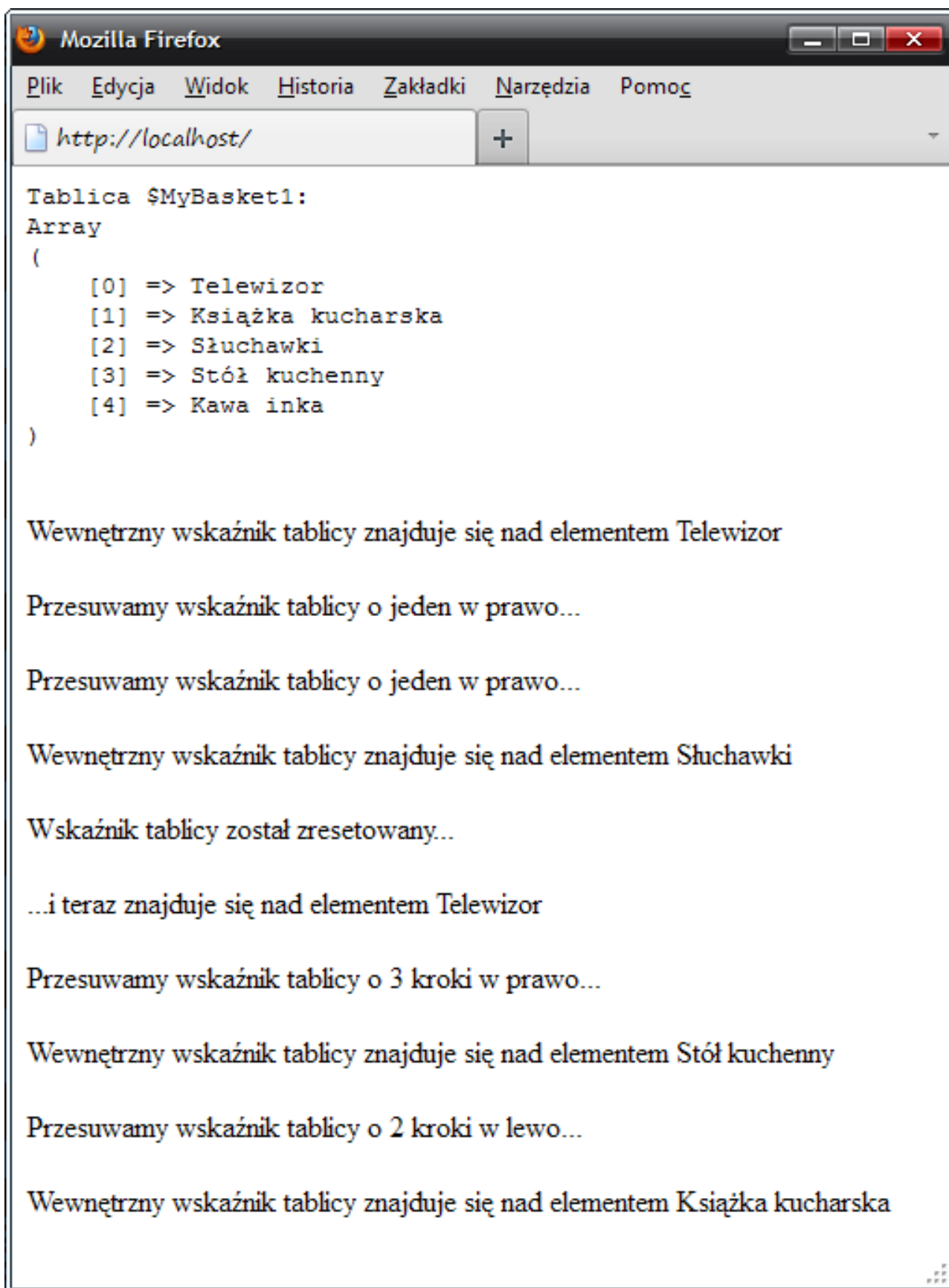
Ćwiczenie 2.3



Ćwiczenie 2.4



Ćwiczenie 2.5



Mozilla Firefox

Plik Edycja Widok Historia Zakładki Narzędzia Pomoc

<http://localhost/>

```
Tablica $MyBasket1:  
Array  
(  
    [0] => Telewizor  
    [1] => Książka kucharska  
    [2] => Słuchawki  
    [3] => Stół kuchenny  
    [4] => Kawa inka  
)
```

Wewnętrzny wskaźnik tablicy znajduje się nad elementem Telewizor

Przesuwamy wskaźnik tablicy o jeden w prawo...

Przesuwamy wskaźnik tablicy o jeden w prawo...

Wewnętrzny wskaźnik tablicy znajduje się nad elementem Słuchawki

Wskaźnik tablicy został zresetowany...

...i teraz znajduje się nad elementem Telewizor

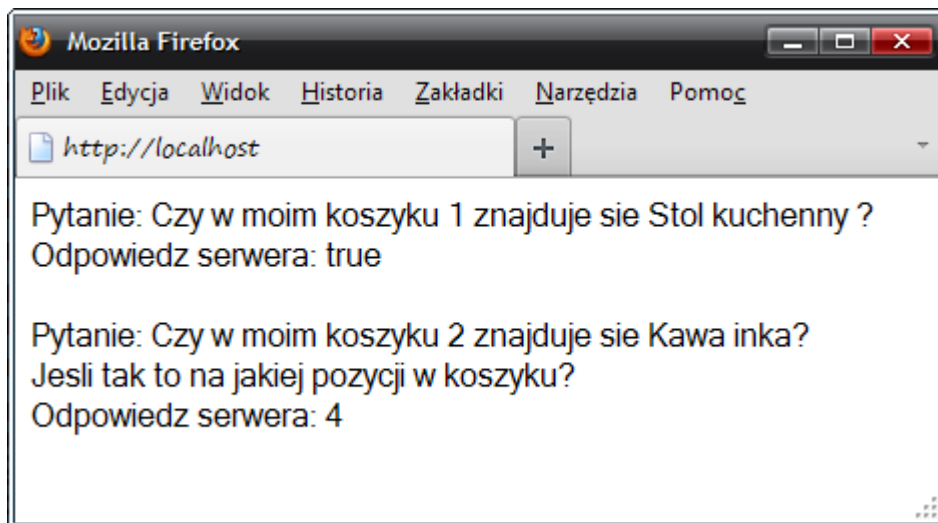
Przesuwamy wskaźnik tablicy o 3 kroki w prawo...

Wewnętrzny wskaźnik tablicy znajduje się nad elementem Stół kuchenny

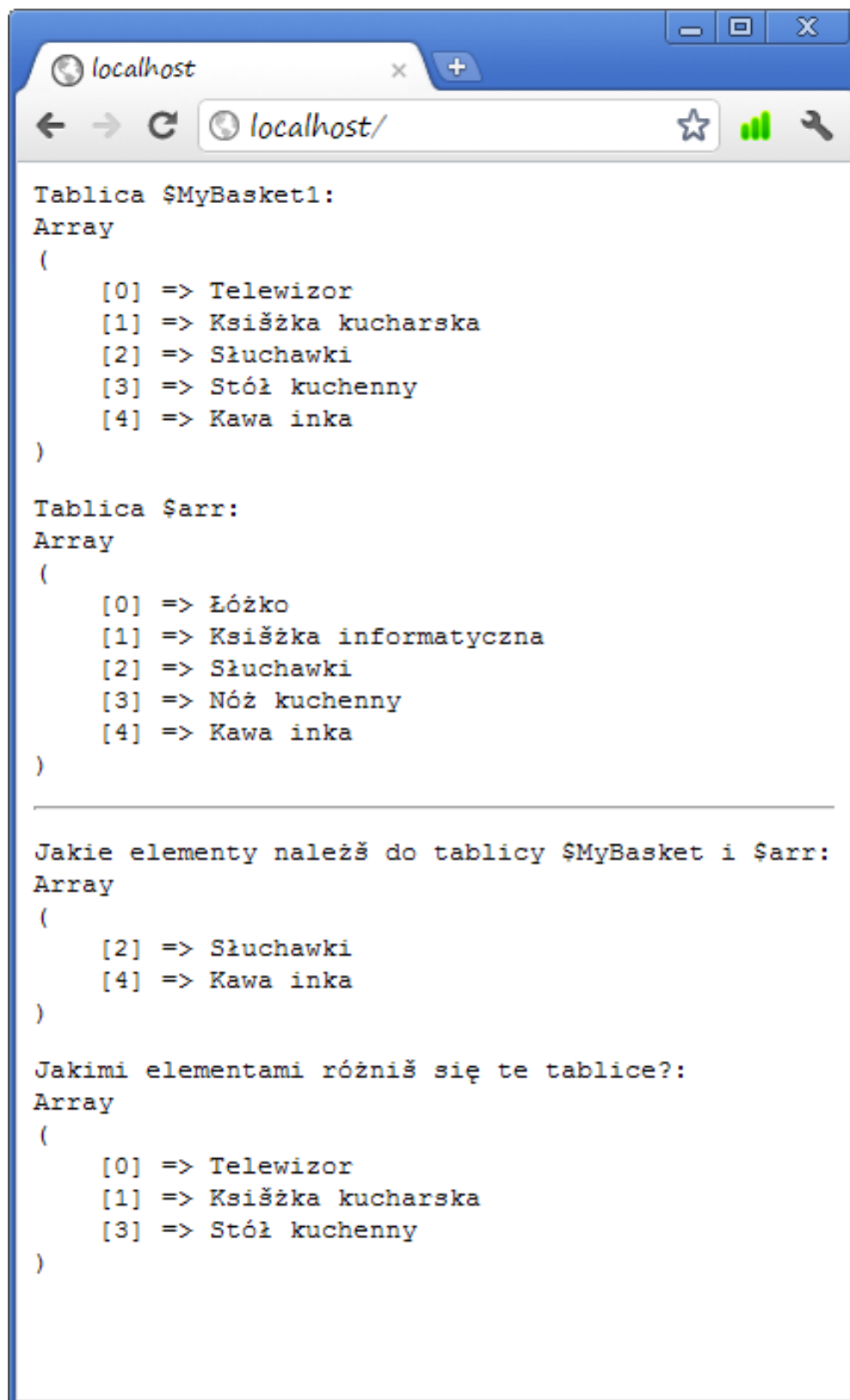
Przesuwamy wskaźnik tablicy o 2 kroki w lewo...

Wewnętrzny wskaźnik tablicy znajduje się nad elementem Książka kucharska

Ćwiczenie 2.6



Ćwiczenie 2.7



The screenshot shows a web browser window with the address bar set to 'localhost/'. The page content displays two PHP arrays and a comparison task. The first array, \$MyBasket1, contains 'Telewizor', 'Książka kucharska', 'Słuchawki', 'Stół kuchenny', and 'Kawa inka'. The second array, \$arr, contains 'Łóżko', 'Książka informatyczna', 'Słuchawki', 'Nóż kuchenny', and 'Kawa inka'. A horizontal line separates these from the task instructions. The instructions ask for elements common to both arrays and for elements that differ between them.

```
Tablica $MyBasket1:  
Array  
(  
    [0] => Telewizor  
    [1] => Książka kucharska  
    [2] => Słuchawki  
    [3] => Stół kuchenny  
    [4] => Kawa inka  
)  
  
Tablica $arr:  
Array  
(  
    [0] => Łóżko  
    [1] => Książka informatyczna  
    [2] => Słuchawki  
    [3] => Nóż kuchenny  
    [4] => Kawa inka  
)  
  
-----  
  
Jakie elementy należą do tablicy $MyBasket i $arr:  
Array  
(  
    [2] => Słuchawki  
    [4] => Kawa inka  
)  
  
Jakimi elementami różni się te tablice?:  
Array  
(  
    [0] => Telewizor  
    [1] => Książka kucharska  
    [3] => Stół kuchenny  
)
```

Zadanie 1

Napisz skrypt obliczający liczbę elementów w tablicy podzielnych i niepodzielnych przez dwa.

Zadanie 2

Napisz kod, który z jednej tablicy zawierającej wartości liczbowe oraz ciągi znaków utworzy dwie oddzielne tablice – jedną zawierającą wyłącznie ciągi znaków i drugą zawierającą wyłącznie wartości liczbowe.

Zadanie 3

Napisz program, który do tablicy dwuwymiarowej o wielkości 3x3 wpisze wartości losowe z zakresu $-5 \dots 5$ wyświetli ją w tabeli HTML i policzy wyznacznik macierzy.