PHP - Sortowanie bąbelkowe

© Copyright by 3bird Projects 2026, http://edukacja.3bird.pl

Ogólne

Aplikacja prezentuje sortowanie liczb metodą bąbelkową.

Czas realizacji: 2 tygodnie.

Zasady oceniania:

- **dostateczny** - wersja podstawowa oddana po wyznaczonym czasie;
- **dobry** - wersja podstawowa + krótki opis algorytmu swoimi słowami (2-3 zdania); wersja rozbudowana oddana po wyznaczonym czasie;
- **bardzo dobry** - wersja rozbudowana oddana w wyznaczonym czasie;
- **celujący** - wersja rozbudowana (oddana w pierwszym tygodniu) + przycisk z walidacją HTML5.

Uwaga: Nigdy nie wolno kopiować kodu z PDF-a, gdyż zawiera on niewidoczne znaki końca linii, twarde odstępy, odmienne cudzysłowy, odmienne apostrofy, odmienne myślniki. To godziny dodatkowej pracy na wykrywanie błędów i ich poprawianie. Kod należy przepisać ze zrozumieniem.

Zasada działania

Jeśli chcemy posortować od najmniejszej do największej liczby, porównujemy najpierw dwie sąsiadujące liczby (3 i 1) i ewentualnie zamieniamy je miejscami. Na przykład:

8

12

5

1

3

W ten sposób, po kilku przebiegach, najmniejsza liczba przesunięta zostanie na samą górę (uniesie się jak bąbelek w szampanie).

Przedstawienie procedury w ujęciu poziomym (pierwszy przebieg pętli FOR):

8, 12, 5, 1, 3

8 ? 12, 5, 1, 3

8, **12 ? 5**, 1, 3

8, 5, **12 ? 1**, 3

8, 5, 1, **12 ? 3**

8, 5, 1, 3, 12

Drugi przebieg pętli FOR:

8, 5, 1, 3, 12

8 ? 5, 1, 3, 12

5, **8 ? 1**, 3, 12

5, 1, **8 ? 3**, 12

5, 1, 3, **8 ? 12**

5, 1, 3, 8, 12

Trzeci przebieg pętli FOR:

5, 1, 3, 8, 12

5 ? 1, 3, 8, 12

1, **5 ? 3**, 8, 12

1, 3, **5 ? 8**, 12

1, 3, 5, **8 ? 12**

1, 3, 5, 8, 12

Kod - wersja podstawowa

Plik „formularz-sortowanie.php”

```
<!DOCTYPE html>
<html lang="pl">
<head>
        <meta charset="utf-8">
        <title>Sortowanie bąbelkowe</title>
</head>
<body>
<h1>PHP - sortowanie babelkowe</h1>
<p>Wprowadź liczby oddzielone spacją:</p>
<form method="post">
        <input id="liczby" type="text" name="liczby">
        <input type="submit" value="Sortuj">
</form>

<p id="wynikSortowania">
```

```php
<?php
if (!empty($_POST['liczby'])) {
        $liczbyJakoTablica = explode(" ", $_POST['liczby']);
        // Proces sortowania
        $czyKolejnyPrzebieg = true;
        while ($czyKolejnyPrzebieg) {
                $czyKolejnyPrzebieg = false;
                $licznik = 0;
                $dlugoscTablicy = count($liczbyJakoTablica) - 1;
                for ($licznik = 0; $licznik < $dlugoscTablicy; $licznik++) {
                        if ($liczbyJakoTablica[$licznik] > $liczbyJakoTablica[$licznik + 1]) {
                                [$liczbyJakoTablica[$licznik], $liczbyJakoTablica[$licznik + 1]] =
                                [$liczbyJakoTablica[$licznik + 1], $liczbyJakoTablica[$licznik]];
                                $czyKolejnyPrzebieg = true;
                        }
                }
        }
        echo implode(" ", $liczbyJakoTablica);
}
?>
</p>
</body>
</html>
```

Kod - wersja rozbudowana

Plik „formularz-sortowanie.php”

```php
<?php
session_start(); // Rozpoczynamy sesję
// Poniżej, podwójny znak zapytania ?? oznacza: Jeśli $_SESSION['error'] istnieje i nie jest null,
// przypisz jego wartość do $error. W przeciwnym razie przypisz pusty string "".
$pierwotne = $_SESSION['pierwotne'] ?? ""; // Przechwytujemy zmienne sesyjne z drugiego pliku
$posortowane = $_SESSION['posortowane'] ?? "";
$error  = $_SESSION['error'] ?? "";
```

```
// Usunięcie danych z sesji po ich odczytaniu:
unset($_SESSION['pierwotne']);
unset($_SESSION['posortowane']);
unset($_SESSION['error']);
?>
<!DOCTYPE html>
<html lang="pl">
<head>
        <meta charset="utf-8">
        <title>Sortowanie bąbelkowe</title>
</head>
<body>
<h1>PHP - sortowanie babelkowe</h1>
<p>Wprowadź liczby oddzielone spacją:</p>
<form action="php-sortowanie-babelkowe-backend.php" method="post" style="margin-bottom: 1cm; width: 450px; margin-left: auto; margin-right: auto; display: block; text-align: center;">
<fieldset style="background-color: #dcdcdc; border-style: none; border-color: #000000; border-width: 2px; padding: 0px;">
<legend style="font-weight: bold; font-size: xx-large; width: 99%; background-color: #000000; color: #ffffff; text-align: center; border-style: solid; border-width: 1px; border-color: #000000; padding-top: 5px; padding-bottom: 5px;">Liczby:</legend>
<input id="liczby" style="display: inline-block; vertical-align: middle; font-size: x-large; text-align: center; margin-top: 1cm; margin-left: 0.5cm; margin-right: 0.5cm;" type="text" name="liczby" placeholder="Wpisz liczby..." required="required" autofocus="autofocus" title="Tu wpisać liczby..." maxlength="24" autocomplete="off" size="24">
<div style="text-align: center; margin-top: 1cm; margin-bottom: 2cm;">
  <input type="submit" value="Sortuj" style="font-size: larger; font-weight: bold;">
</div>
</fieldset>
</form>

<p style="text-align: center;" id="wprowadzone"></p>
<p style="text-align: center;" id="wynikSortowania"></p>
<p style="text-align: center; color: red;" id="blad"></p>
```

```
<script>
// Czeka aż całe drzewo DOM zostanie załadowane:
document.addEventListener("DOMContentLoaded", function() {
// Użyto json_encode, aby wyeliminować problemy z potencjalnymi cudzysłowami w danych:
var liczbyWprowadzone = <?php echo json_encode($pierwotne); ?>;
var liczbyPosortowane = <?php echo json_encode($posortowane); ?>;
var blad  = <?php echo json_encode($error); ?>;

if(liczbyWprowadzone !== "") {
        let miejsceNaLiczbyWprowadzone = document.getElementById("wprowadzone");
        let spanNaTekstLiczbyWprowadzone = document.createElement("span");
        spanNaTekstLiczbyWprowadzone.style.textDecorationLine = "underline";
        spanNaTekstLiczbyWprowadzone.textContent = "Liczby wprowadzone";
        miejsceNaLiczbyWprowadzone.appendChild(spanNaTekstLiczbyWprowadzone);

        let dwukropekDoLiczbyWprowadzone = document.createTextNode(": ");
        miejsceNaLiczbyWprowadzone.appendChild(dwukropekDoLiczbyWprowadzone);

        let nowaLiniaDoLiczbyWprowadzone = document.createElement("br");
        miejsceNaLiczbyWprowadzone.appendChild(nowaLiniaDoLiczbyWprowadzone);

        let liczbyWprowadzoneJakoObiekt = document.createTextNode(liczbyWprowadzone +
        ".");
        miejsceNaLiczbyWprowadzone.appendChild(liczbyWprowadzoneJakoObiekt);
}

if(liczbyPosortowane !== "") {
        let miejsceNaLiczbyPosortowane = document.getElementById("wynikSortowania");
        let spanNaTekstLiczbyPosortowane = document.createElement("span");
        spanNaTekstLiczbyPosortowane.style.textDecorationLine = "underline";
        spanNaTekstLiczbyPosortowane.textContent = "Liczby posortowane";
        miejsceNaLiczbyPosortowane.appendChild(spanNaTekstLiczbyPosortowane);

        let dwukropekLiczbyPosortowane = document.createTextNode(": ");
```

```
        miejsceNaLiczbyPosortowane.appendChild(dwukropekLiczbyPosortowane);

        let nowaLiniaDoLiczbyPosortowane = document.createElement("br");
        miejsceNaLiczbyPosortowane.appendChild(nowaLiniaDoLiczbyPosortowane);

        let spanNaLiczbyPosortowane = document.createElement("span");
        spanNaLiczbyPosortowane.style.color = "red";
        spanNaLiczbyPosortowane.style.fontSize = "x-large";
        spanNaLiczbyPosortowane.style.fontWeight = "900";
        spanNaLiczbyPosortowane.textContent = liczbyPosortowane;
        miejsceNaLiczbyPosortowane.appendChild(spanNaLiczbyPosortowane);

        let kropkaLiczbyPosortowane = document.createTextNode(".");
        miejsceNaLiczbyPosortowane.appendChild(kropkaLiczbyPosortowane);
}

if(blad !== "") {
        let bladKomunikat = document.createTextNode("Błąd: " + blad);
        miejsceNaBlad = document.getElementById("blad");
        miejsceNaBlad.appendChild(bladKomunikat);
}
});
</script>
</body>
</html>
```

Plik „php-sortowanie-babelkowe-backend.php”

```
<?php
// Włączamy się do sesji:
session_start();
// Poniżej, zapasowe zabezpieczenie, gdyż puste pole wykrywa już sam formularz w HTML5.
// Funkcja "trim" usuwa białe znaki z początku i z końca stringa (spacje, tabulatory, znaki nowej
// linii):
if(!isset($_POST['liczby']) || trim($_POST['liczby']) === ""){
```

```
    $_SESSION['error'] = "Nie wprowadzono żadnych danych.";
    header("Location: formularz-sortowanie.php");
    exit(); // Zatrzymujemy przetwarzanie dalszej części kodu
}

// Funkcja "trim" usuwa białe znaki z początku i końca stringa (spacje, tabulatory, znaki nowej
// linii):
$oczyszczonyString = trim($_POST['liczby']);
// Poniżej, funkcja "preg_split(wzorzec, tekst)" usuwa spacje wewnątrz stringa, wydobywa stringi
// liczbowe i wkłada je do tablicy. Wyrażenie \s oznacza spację, tab, nowąLinię, zaś znak "+"
// oznacza "jeden lub więcej". Znak "/" definiuje początek i koniec wzorca.
// Można zrobić to tak:
// $liczbyJakoTablica = preg_split('/\s+/', $oczyszczonyString); // Formuła podstawowa
// Lepiej jednak usuwać także przecinki i średniki:
// $liczbyJakoTablica = preg_split('/[\s,;]+/', $oczyszczonyString); // Nawiasy kwadratowe
// oznaczają zbiór znaków (spacja, przecinek, średnik)
// A jeszcze lepiej zrobić to tak (obsługa liczb ujemnych i rzeczywistych):
$liczbyJakoStringiWTablicy = preg_split('/[^\d\.\-]+/', $oczyszczonyString); // Usuń wszystko,
// co nie wygląda jak liczba, kropka lub minus

// Sprawdzamy każdy element tablicy, czy na pewno jest stringiem o wartości liczbowej, choć w
// praktyce sprawę zalatwia nam  przecież już preg_split(), więc jest to zabezpieczenie
// dodatkowe.
foreach($liczbyJakoStringiWTablicy as $kolejnyElementTablicy){
    if(!is_numeric($kolejnyElementTablicy)){ // Czy string zawiera dane podobne do liczby
        $_SESSION['error'] = "Wprowadzono nieprawidłowe dane.";
        header("Location: formularz-sortowanie.php");
        exit(); // Zatrzymujemy przetwarzanie dalszej części kodu
    }
}

// Funkcja array_map() wywołuje funkcję floatval dla każdego elementu tablicy, czyli
// zamienia każdy element tablicy  (nadal typu string) na liczbę zmiennoprzecinkową.
$liczbyJakoTablica = array_map('floatval', $liczbyJakoStringiWTablicy);
```

```php
// Zapamiętanie pierwotnie wprowadzonych liczb:
$pierwotneLiczby = $liczbyJakoTablica;

// SORTOWANIE BĄBELKOWE
// Na początku wartość "true", aby w ogóle uruchomić pętlę:
$czyKolejnyPrzebieg = true;

while ($czyKolejnyPrzebieg) {  // Czy w danym przebiegu nastąpiła jakaś zamiana liczb?
        // Optymistycznie i przekornie zakładamy, że tym razem żadna zamiana liczb nie nastąpi
        // (że są już posortowane):
        $czyKolejnyPrzebieg = false;
        // Licznik pętli ustawiamy na 0:
        $licznik = 0;
        $dlugoscTablicy = count($liczbyJakoTablica) - 1; // Czyli ile liczb wpisał użytkownik
        // (indeksujemy od 0)
        for ($licznik = 0; $licznik < $dlugoscTablicy; $licznik++) { // Pętla wykona się tyle razy, ile
        // jest wpisanych liczb.
        // Jeśli bieżący element w tablicy jest większy niż następny element, zamień je ze sobą:
                if ($liczbyJakoTablica[$licznik] > $liczbyJakoTablica[$licznik + 1]) {
                        // Zamiana miejscami elementów:
                        [$liczbyJakoTablica[$licznik], $liczbyJakoTablica[$licznik + 1]] =
                        [$liczbyJakoTablica[$licznik + 1], $liczbyJakoTablica[$licznik]];
                        // Taką zamianę można również przeprowadzić z użyciem zmiennej
                        // tymczasowej $temp:
                        // $temp = $liczbyJakoTablica[$licznik];
                        // $liczbyJakoTablica[$licznik] = $liczbyJakoTablica[$licznik + 1];
                        // $liczbyJakoTablica[$licznik + 1] = $temp;
                        $czyKolejnyPrzebieg = true;    // Nastąpiła jakaś zamiana liczb, więc nie są
                        // posortowane i  jest potrzebny kolejny przebieg.
                }
        }
}

$_SESSION['pierwotne'] = implode(" ", $pierwotneLiczby);
$_SESSION['posortowane']   = implode(" ", $liczbyJakoTablica);
```

```
// Po obliczeniach, wracamy na poprzednią stronę, gdzie zostanie wypisany komunikat.
header("Location: formularz-sortowanie.php");
exit();
?>
```

Ostatnia aktualizacja: 19 marca 2026.