**Array** – metody iterujące:

<https://jsbin.com/dumujagore/edit?js,console>

* **forEach** wykonuje funkcję na każdym elemencie tablicy:

"a"

"b"

"c"

function **wypisz**(value) {

    console.log(value)

}

const ar = ["a", "b", "c"]

ar.forEach(**wypisz**)

Odpowiada to:

for(let i = 0; i < ar.length; i++) {

    console.log(ar[i])

}

albo:

for(let value of ar) {

    console.log(value)

}

* albo:

ar.forEach(function(value) {

    console.log(value)

})

* albo:

ar.forEach(value => console.log(value))

* **filter** zwraca tablicę elementów spełniających warunek

let filtered = ar.filter(val => val % 2 == 0) // => [2, 4]

<https://jsbin.com/sugibawini/edit?js,console>

**Ćwiczenie 1**

Stwórz tablicę wypełnioną losowymi liczbami z zakresu 1-20, a następnie przy pomocy funkcji **filter** wybierz z niej wszystkie liczby z zakresu od 5 do 15 i wypisz je do konsoli.

**Ćwiczenie 2**

Zadeklaruj tablicę wypełnioną danymi studentów na wzór:

[ { imię: "Piotr", nazwisko: "Nowak", punkty: 43 },

  { imię: "Tomasz", nazwisko: "Kowalski", punkty: 68 },

  { imię: "Julia", nazwisko: "Bagińska", punkty: 75 }, ... ]

Następnie przy pomocy funkcji **filter** wybierz z niej osoby, których liczba punktów przekracza 50.

**Ćwiczenie 3**

Z listy studentów uzyskanej z poprzedniego zadania usuń przy pomocy funkcji **filter** te osoby, których nazwisko zaczyna się na literę K.

* **map** na podstawie jednej tablicy generuje drugą

<https://jsbin.com/hozofacixi/edit?js,console>

Funkcja Math.floor jest wykonywana na każdym elemencie tablicy

let ar = [1.2, 3.4, 5.6, 7.8]

let wynik = ar.**map**(Math.floor)

console.log(wynik) // => [1, 3, 5, 7]

a zwracana jest tablica wyników

* inny przykłady – przetwarzanie danych:

ar = [1, 2, 3, 5, 7].map(val => 1 / val)

console.log(ar.map(val => val.toFixed(3)).join(", "))

// 1.000, 0.500, 0.333, 0.200, 0.143"

* przykład użycia map do wyłuskania interesującej nas informacji:

const users = [ { name: "Anna", age: 25, score: 300 },

    { name: "Paweł", age: 30, score: 250 },

     { name: "Ewa", age: 20, score: 150 }, ]

console.log(users.map(user => user.name))

// ["Anna", "Paweł", "Ewa"]

**Ćwiczenie 4**

Wygeneruj tablicę liczb rzeczywistych. Następnie przy pomocy funkcji map zamień ją na tablicę wartości sformatowanych jako kwota w zł.

(np. wartość 2.5 zmieni się w "2.50 zł").

**Ćwiczenie 5**

Z listy studentów z **Ćwiczenia 3** korzystając z funkcji **map** wyświetl na ekranie same nazwiska.

**Ćwiczenie 6**

Zdefiniuj tablicę produktów na wzór poniższej:

[ { nazwa: "jajka", ilość: 6, cena: 1.2 },

  { nazwa: "mleko", ilość: 2, cena: 8.8 },

  { nazwa: "chleb", ilość: 1, cena: 7.5 }, ... ]

Następnie, korzystając z funkcji **map**, stwórz listę typu <ul> zawierającą nazwy towarów oraz ich ceny.

(np. "<ul><li>jajka (1.20 zł)</li><li>mleko (8.80 zł)</li>...</ul>").

* **every** i **some** sprawdzają czy każdy (lub przynajmniej jeden) element spełnia podany warunek

function parzysta(value) {

Czy każda jest parzysta?

    return (value % 2 == 0);

}

[1, 2, 3, 4].every(parzysta) // => false

[1, 2, 3, 4].some(parzysta) // => true

Czy istnieje jakaś parzysta?

Nie modyfikuje oryginału

<https://jsbin.com/sewucegope/edit?js,console>

* można użyć do sprawdzenia, czy dana wartość znajduje się w tablicy:

const ar = ["a", "b", "c", "d"]

console.log(ar.some(val => val === "c")) // => true

* ale do tego mamy prostszą metodę **includes**:

console.log(ar.includes("c")) // => true

* **find** oraz **findIndex** pozwalają na znalezienie elementu (lub jego indeksu) spełniającego określony warunek

Zwróć pierwszy element parzysty

const ar = [3, 5, 6, 4, 1]

let value = ar.find(val => val % 2 == 0) // => 6

let index = ar.findIndex(val => val % 2 == 0) // => 2

Zwróć indeks pierwszego elementu parzystego

* jeśli interesuje nas indeks konkretnej wartości, możemy posłużyć się metodą indexOf:

let index = ar.indexOf(6) // => 2

**Ćwiczenie 7**

Sprawdź czy choć jedna osoba z **Ćwiczenia 2** zdobyła przynajmniej 90 punktów. Użyj metody **some**.

**Ćwiczenie 8**

Sprawdź czy wszystkie osoby z **Ćwiczenia 2** zdobyły ponad 33 punkty. Użyj metody **every**.

**Ćwiczenie 9**

Na liście produktów z **Ćwiczenia 6** znajdź pierwszy produkt o cenie powyżej 5 zł. Użyj metody **find**.

* **reduce** i **reduceRight** redukują (przez akumulację) zawartość do pojedynczej wartości

Pierwszy parametr to wynik ostatniego wywołania

function suma(last, curr) {

    return last + curr

}

[1, 2, 3, 4, 5].reduce(suma) // => 15

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [1, | 2, | 3, | 4, | 5] |
| | | | | | | | | | |
| suma(1, 2) | | | | | | | |
| 3 | | | | | | | |
| | | | | | | | | |
| suma(3, 3) | | | | | | |
| 6 | | | | | | |
| suma(6, 4) | | | | | |
| 10 | | | | | |
| suma(10, 5) | | | | |
| 15 | | | | |

<https://jsbin.com/lasitinebu/edit?js,console>

**Ćwiczenie 10**

Stwórz tablicę losowych wartości. Użyj funkcji **reduce**, aby obliczyć ich średnią (**reduce** liczy sumę, a później dzielimy ją przez liczbę elementów).

**Ćwiczenie 11**

Na losowej tablicy z **Ćwiczenia 10** użyj **reduce**, aby znaleźć wartość elementu maksymalnego.

*Podpowiedź: funkcja przekazana do* ***reduce*** *powinna zwracać większą wartość ze swych dwóch parametrów.*

**Ćwiczenie 12**

Na podstawie listy produktów z **Ćwiczenia 6** korzystając z metody **reduce** oblicz łączną wartość zakupów (pamiętaj o przemnożeniu ilości przez cenę).

*Podpowiedź: należy przekazać 0 jako drugi parametr* ***reduce*** *(patrz kolejna strona). Funkcja przekazana do* ***reduce*** *przyjmuje dwa parametry: pierwszy to aktualna wartość sumy, a drugi – kolejny obiekt produktu do zsumowania.*

**Ćwiczenie 13**

Na liście studentów z **Ćwiczenia 2** znajdź przy pomocy metody **reduce** osobę, która zdobyła najwięcej punktów.

* jako drugi parametr reduce można przekazać wartość początkową

[1, 2, 3, 4, 5].reduce(suma, 100) // => 115

Wówczas pierwsze działanie to suma(100, 1)

* Można to wykorzystać do zliczania elementów:

const ar = [3, 4, 1, 2, 5]

let ile = ar.reduce((licz, val) =>

val % 2 == 0 ? licz + 1 : licz, 0)

Początkowa wartość licznika

<https://jsbin.com/widefopapa/edit?js,console>

* Albo liczenia powtórzeń:

const zbior = ["a", "b", "c", "b", "c", "b"]

let wynik = zbior.reduce((slownik, element) => {

    slownik[element] = (slownik[element] ?? 0) + 1

    return slownik

Początkowy słownik

}, {})

// wynik: {a: 1, b: 3, c: 2}

<https://jsbin.com/xarirevulu/edit?js,console>