

WEEK 1

DAY 2



[]

Array

ПЛАН

- ❏ Что такое массив
- ❏ Методы массива
- ❏ Перебор массива
- ❏ Копирование массива
- ❏ Советы по парному программированию

МАССИВ

Массив - упорядоченный набор чего либо.

```
const emptyArr = []; // пустой массив
```

```
const arr = ["Kate", 42, true, "Hello world!"];  
arr[1]; // 42
```

МЕТОДЫ МАССИВА

UNSHIFT / SHIFT

```
const arr = ["a", "b", "c"];
```

- ▶ Добавить элемент в начало массива

```
arr.unshift("e");
```

```
console.log(arr); // ["e", "a", "b", "c"]
```

- ▶ Извлечь элемент из начала массива

```
const x = arr.shift();
```

```
console.log(x); // "e"
```

```
console.log(arr); // ["a", "b", "c"]
```

PUSH / POP

```
const arr = ["a", "b", "c"];
```

- ▶ Добавить элемент в конец массива

```
arr.push("d");  
console.log(arr); // ["a", "b", "c", "d"]
```

- ▶ Извлечь элемент с конца массива

```
const x = arr.pop();  
console.log(x); // "d"  
console.log(arr); // ["a", "b", "c"]
```

SPLIT ("") / JOIN ([])

```
const names = "Igor,Anna,Vasya";
```

- ▶ Преобразовать строку в массив

```
const arr = names.split(","); // ["Igor", "Anna", "Vasya"]
```

- ▶ Преобразовать массив в строку

```
const str = arr.join("-"); // "Igor-Anna-Vasya"
```


SLICE

```
const arr = ["a", "b", "c", "d", "e"];
```

- ▶ Вернуть новый массив с 1 индекса

```
arr.slice(1); // ["b", "c", "d", "e"]
```

- ▶ Вернуть новый массив начиная со 2 индекса и заканчивая 4 индексом (не включая его)

```
arr.slice(2,4); // ["c", "d"]
```

SPLICE

```
const arr = ["a", "b", "c", "d", "e"];
```

- ▶ Удалить 2 элемента с 1 индекса

```
arr.splice(1,2); // arr = ["a", "d", "e"]
```

- ▶ Удалить 2 элемента с 1 индекса и добавить другие вместо них

```
arr.splice(1,2,"x","y","z");
```

```
// arr = ["a", "x", "y", "z", "d", "e"]
```

- ▶ Удалить 0 элементов с 1 индекса и добавить другие

```
arr.splice(1,0,"x","y");
```

```
// arr = ["a", "x", "y", "b", "c", "d", "e"]
```

CONCAT / REVERSE

```
const male = ["Vasya", "Igor"];
```

```
const female = ["Anna", "Marina"];
```

- ▶ Создать новый массив из нескольких

```
const people = male.concat(female, ["Max", "Nataly"]);  
// ["Vasya", "Igor", "Anna", "Marina", "Max", "Nataly"]
```

- ▶ Переставить элементы массива в обратном порядке

```
female.reverse(); // ["Marina", "Anna"]
```

SOME / EVERY

```
function isBigEnough(element) {  
  return element >= 10;  
}
```

- ▶ Проверяет, удовлетворяют ли **все** элементы массива заданному условию

```
[12, 5, 8, 130, 44].every(isBigEnough); // false  
[12, 54, 18, 130, 44].every(isBigEnough); // true
```

- ▶ Проверяет, удовлетворяет ли **хотя бы один** элемент массива заданному условию

```
[12, 5, 8, 130, 44].some(isBigEnough); // true
```

FILTER

```
function getLongWord(word) {  
  return word.length > 5;  
}
```

- ▶ Создаёт новый массив со всеми элементами, удовлетворяющими заданному условию

```
const words = ["lemon", "papaya", "apple", "cherry"];  
words.filter(getLongWord); // ["papaya", "cherry"];
```

MAP

- ▶ Создаёт новый массив с результатом вызова функции

```
const words = ["lemon", "papaya", "apple", "cherry"];  
words.map(function(word) {  
  return `${word}-${word.length}`;  
});  
// ["lemon-5", "papaya-6", "apple-5", "cherry-6"]
```

SORT

- ▶ Сортирует элементы массива и возвращает отсортированный массив. Порядок сортировки по умолчанию соответствует порядку кодовых точек Unicode.

```
const numbers = [1, 5, 11, 28, 3, 30, 9, 4];  
numbers.sort();  
// [1, 11, 28, 3, 30, 4, 5, 9]
```

```
numbers.sort(function(a, b) {  
  return a - b;  
});  
// [1, 3, 4, 5, 9, 11, 28, 30]
```

REDUCE / REDUCERIGHT

- ▶ `reduce()` применяет функцию к каждому значению массива (слева-направо), сводя его к одному значению (аккумулятору).

```
const arr = [1, 2, 3, 4, 5, 4, 4, 4, 4];
```

```
arr.reduce(function(count, current) {  
  if (current % 2 === 0) {  
    return count + 1;  
  }  
  return count;  
}, 0); // 6
```

- ▶ `reduceRight()` делает то же, что и `reduce()` для элементов массива справа-налево

ПЕРЕБОР МАССИВА

FOR

```
const arr = ["a", "b", "c"];
```

```
for(let i = 0; i < arr.length; i++) {  
  // перебор всех элементов массива  
}
```

FOR IN

```
const arr = ["a", "b", "c"];
```

```
for(const i in arr) {
```

```
  // перебор только тех элементов массива,  
  // которым присвоены значения
```

```
  // i - индекс текущего элемента
```

```
}
```

FOR OF

```
const arr = ["a", "b", "c"];
```

```
for(const val of arr) {
```

```
    // перебор всех элементов массива
```

```
    // val - значение текущего элемента
```

```
}
```

FOREACH

```
const arr = ["a", "b", "c"];
```

```
arr.forEach(function(val, i) {
```

```
    // перебор только тех элементов массива,  
    // которым присвоены значения
```

```
    // val - значение текущего элемента
```

```
    // i - индекс текущего элемента  
    (необязательный параметр)
```

```
});
```

КОПИРОВАНИЕ МАССИВА

ПОВЕРХНОСТНОЕ КОПИРОВАНИЕ

```
const arr = ["a", "b", "c"];
```

▶ Нельзя:

```
const fullCopy0 = arr;
```

▶ Можно:

```
const fullCopy1 = [...arr];
```

```
const fullCopy2 = arr.slice();
```

```
const fullCopy3 = arr.map(el => el);
```

ГЛУБОКОЕ КОПИРОВАНИЕ

```
const arr = ["a", "b", "c"];
```

```
const deepCopy = JSON.parse(JSON.stringify(arr));
```


Парное программирование

Инструкция

1. Работать в 1 форке репозитория. Напарника нужно добавить в Collaborators репозитория.
2. Роли меняются по таймеру каждые 30 минут
3. Использовать 1 компьютер (второй нужно выключить, иначе можно уйти в соло-групповую работу) и монитор
4. Договариваться о времени обеда. В это время не работать по одиночке, а ждать напарника.
5. Менять роли Драйвера и Навигатора
6. В конце парного программирования дать обратную связь что стоит улучшить в следующий раз.

Парное программирование

Преимущества парного программирования:

- ✓ Обмен опытом
- ✓ Знания о системе
- ✓ Коллективное владение кодом
- ✓ Наставничество
- ✓ Больше общения
- ✓ Стандарты кодирования
- ✓ Улучшение дисциплины
- ✓ Сопряжение потока
- ✓ Меньше прерываний

Парное программирование

Анти-паттерны

- ✗ Наблюдай за Мастером
- ✗ Диктатор
- ✗ Сходи за кофе
- ✗ Молчаливые партнеры
- ✗ Разделение задач за одним столом
- ✗ Неудобно сидеть
- ✗ Партнер занят своим делом
- ✗ Свои настройки окружения

Обратная связь

Как дать обратную связь?

1. Сильная сторона
2. Пожелание
3. Результат