САМАРСКИЙ УНИВЕРСИТЕТ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Лекция № 7

Основные алгоритмы обработки данных

2023

РУБРИКА: «ВОПРОСЫ ПО ЛЕКЦИИ № 6»

Предисловие о структуре данных

Что такое Структура данных?

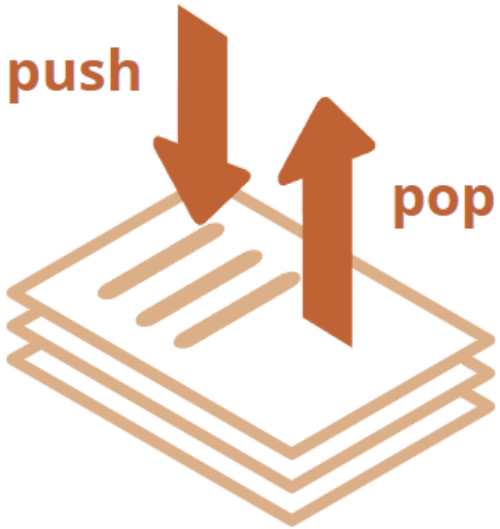
Это сведения о строении некоторой модели данных, над которым может выполнять действия реализованное программное обеспечение.

Зачем это нужно?

Именно с этой целью и был изобретен TypeScript, а именно для реализации упрощения построения структур данных и их ведения в процессе дополнения ПО новыми функциональными возможностями, их модификации, оптимизации и так далее... (повышение читабельности кода, повышение скорости разработки, модификации, повышение эффективности поиска ошибок в коде).
В чистом JavaScript структуры данных не имеют валидации и пишутся вслепую, так как отсутствует компиляция, от чего могут следовать неявные ошибки, которые могут попасть в ПО к конечному пользователю.

Что такое стек?

Это структура данных, которая позволяет последовательно складывать некоторые данные по схеме «Первый пришел – первый ушел (FIFO)»



Структуры данных :: Стек :: Пример в Typescript

Для реализации структуры данных достаточно иметь некоторый одномерный массив кастомного или примитивного типа.

Для реализации добавления элемента в стек используется метод push(elem: T | any), для его извлечения используется метод pop()

```
...
const array: number[] = [];
array.push(10);
array.push(20);
const a0 = array.pop();
const a1 = array.pop();
...
```

array = []
array = [10]
array = [10,20];
array = [10], a0 = 20;
array = [], a1 = 10;

Что такое очередь?

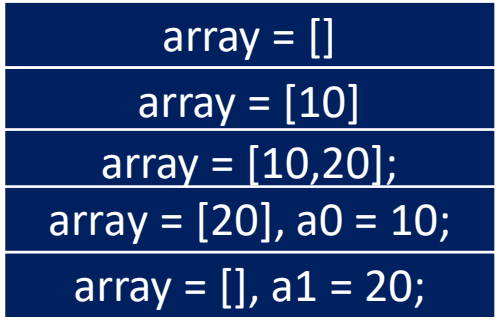
Это структура данных, которая позволяет последовательно складывать некоторые данные по схеме «Первый пришел – последний ушел (LIFO)»

Структуры данных :: Очередь :: Пример в Typescript

Для реализации структуры данных достаточно иметь некоторый одномерный массив кастомного или примитивного типа.

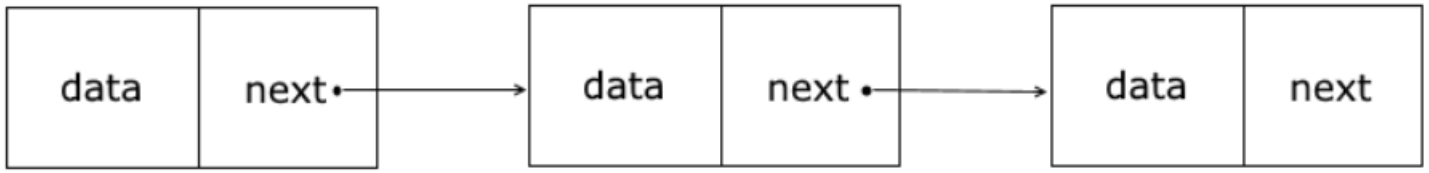
Для реализации добавления элемента в очередь используется метод push(elem: T | any), для его извлечения используется метод shift()

```
...
const array: number[] = [];
array.push(10);
array.push(20);
const a0 = array.shift();
const a1 = array.shift();
...
```



Что такое список?

Это структура данных, которая позволяет хранить данные одного типа в виде цепочки, где каждый объект-узел имеет данные некоторого типа и ссылку на следующий элемент (иногда и предыдущий)



Структуры данных :: Список :: Пример в Typescript

Для реализации этой структуры данных на Typescript достаточно иметь некоторый одномерный массив кастомного или примитивного типа.

Для реализации управления данными списка могут использоваться те же методы, что и для массива (push, pop, shift, unshift, slice, splice и другие). Для получения отдельных узлов предусмотрены методы поиска индекса(indexOf), прямой доступ по индексу (...array[i]) и др.

```
...
const array: number[] = [];
array.push(10);
array.push(20);
array.push(30);
const newArray: number[] = [];
let i = 0;
for(let item of array) {
  if(item > 10) {
     newArray.push(item);
     i++;
   } else
        newArray.splice(i, 1);
   console.log(item);
}
```

array = []

array = [10]

array = [10,20];

array = [10,20,30];

array = [10,20,30]; newArray = [];

array = [10,20,30]; newArray = []; i = 0;

array = [20,30]; item = 10; newArray = []; i = 0;

array = [20,30]; item = 20; newArray = [20]; i = 1;

array = [20,30]; item = 30; newArray = [20, 30]; i = 2;

Что такое алгоритм?

Это некоторый конечный набор действий над исходными данными, приводящий к некоторому результату их трансформации к необходимому виду.

Основные типы алгоритмов в программировании

- Сортировка
- Фильтрация
- Поиск

Фильтрация это

Исключение данных из некоторой ранее полученной исходной выборки данных этого же типа

```
...
const array: number[] = [];
array.push(10);
array.push(20);
array.push(30);
const newArray: number[] = [];
let i = 0;
for(let item of array) {
 if(item > 10) {
   newArray.push(item);
   i++;
  } else
      newArray.splice(i, 1);
  console.log(item);
}
```

array = []
array = [10]
array = [10,20];
array = [10,20,30];
array = [10,20,30]; newArray = [];
array = [10,20,30]; newArray = []; i = 0;
array = [20,30]; item = 10; newArray = []; i = 0;
array = [20,30]; item = 20; newArray = [20]; i = 1;
array = [20,30]; item = 30; newArray = [20, 30]; i = 2;

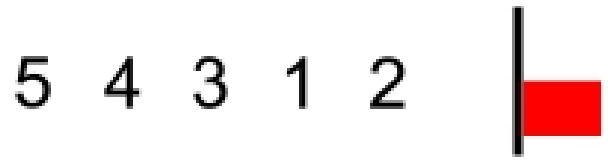
Сортировка это

Изменение порядка с упорядочиванием следования массива данных некоторого типа по некоторому параметру и/или параметрам

Сортировка пузырьком

Сортиро́вка простыми обменами, сортировка пузырько́м (bubble sort).
Для понимания и реализации этот алгоритм — простейший.
Эффективен лишь для небольших массивов.

```
function bubbleSortConcept1(arr) {
  for (let j = arr.length - 1; j > 0; j--) {
    for (let i = 0; i < j; i++) {
      if (arr[i] > arr[i + 1]) {
        let temp = arr[i];
        arr[i] = arr[i + 1];
        arr[i + 1] = temp;
      }
    }
  }
}
```



Сортировка выбором

Сортиро́вка простыми обменами, сортировка пузырько́м (bubble sort).
Для понимания и реализации этот алгоритм — простейший.
Эффективен лишь для небольших массивов.

```
function selectionSort(inputArr) {
    let n = inputArr.length;

    for(let i = 0; i < n; i++) {
        let min = i;
        for(let j = i+1; j < n; j++){
            if(inputArr[j] < inputArr[min]) {
                min=j;
            }
         }
         if (min != i) {
            let tmp = inputArr[i];
            inputArr[i] = inputArr[min];
            inputArr[min] = tmp;
        }
    }
    return inputArr;
}
```

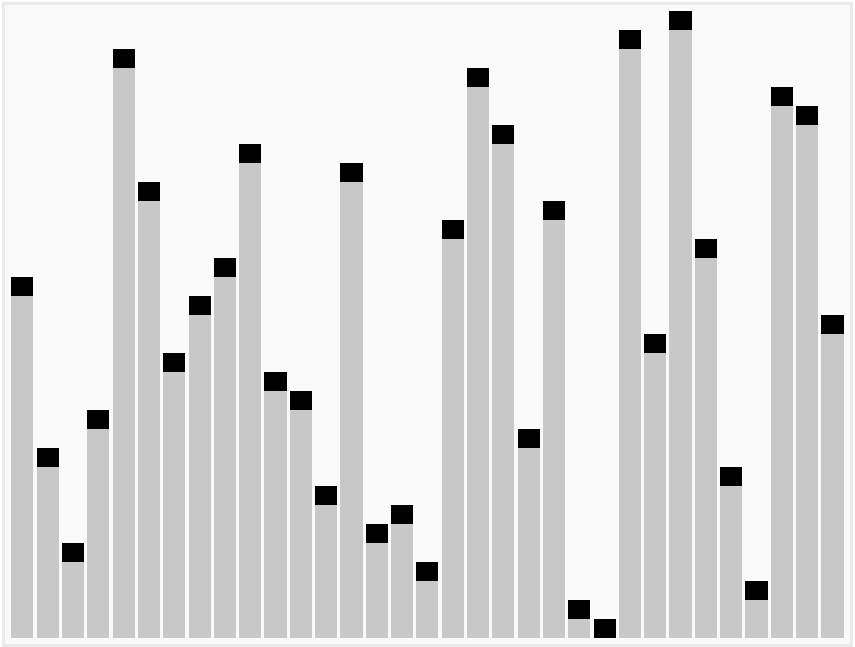
8	5	2	6	9	3	1	4	0	7

Быстрая сортировка

Один из самых быстрых известных универсальных алгоритмов сортировки массивов: в среднем O(n\log n) обменов при упорядочении n элементов; из-за наличия ряда недостатков на практике обычно используется с некоторыми доработками.

```
function quickSort(arr) {
  if (arr.length < 2) return arr;
  let pivot = arr[0];
  const left = [];
  const right = [];

  for (let i = 1; i < arr.length; i++) {
    if (pivot > arr[i]) {
      left.push(arr[i]);
    } else {
      right.push(arr[i]);
    }
  }
  return quickSort(left).concat(pivot, quickSort(right));
}
```



РУБРИКА: «ВОПРОСЫ СЛУШАТЕЛЕЙ»

КОНЕЦ ЛЕКЦИИ № 7
СПАСИБО ЗА ВНИМАНИЕ

КОНЕЦ КУРСА «Язык программирования TypeScript»