ТЕМА 6
«Списки. Массивы.»

Алексеева Галина

Содержание Темы 6:

- Основные понятия. Виды (одномерные и многомерные), назначение
- Основные операции со структурами данных
- Поиск. Виды поиска
- Сортировка. Алгоритмы сортировки одномерных массивов
- Базовые операции с одномерными массивами чисел и строк: создание, удаление, доступ к ячейке
- Обработка двумерных массивов
- Домашнее задание

Основные понятия

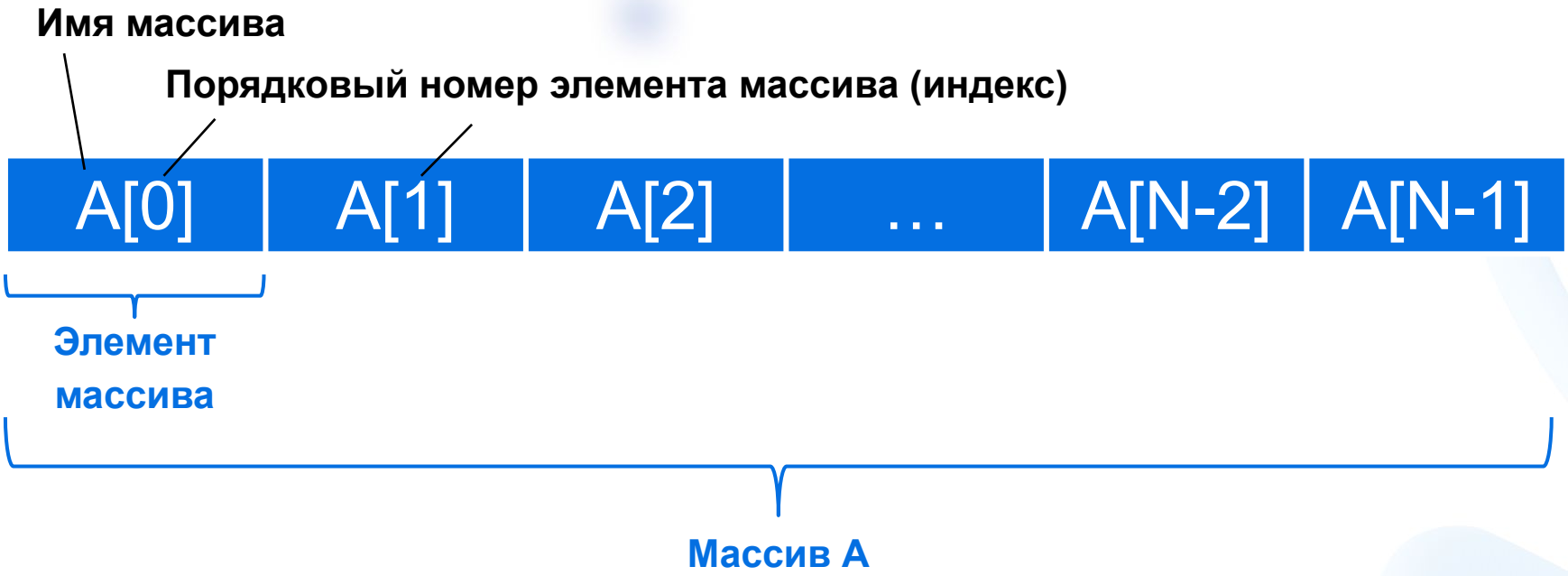
Массив –

последовательность переменных одного типа.

Список –

это абстрактный тип данных, представляющий собой упорядоченный набор значений, в котором некоторое значение может встречаться более одного раза.

Основные понятия



N – размерность массива (количество элементов)

Основные понятия

Классификация массивов

- одномерные и многомерные;
- статические и динамические;
- порядковые и ассоциативные;
- данные одного типа и гетерогенные.

Основные понятия

Объявления массивов:

- `let massiv = new Array();`
- `let massiv = [];`

Основные понятия

Примеры объявления массивов:

- `let massiv = new Array();`
- `let massiv = new Array(2);`
- `let massiv = new Array(23, 34, 12, 89);`
- `let massiv = new Array("а", "б", "в", "г");`
- `let massiv = [];`
- `let massiv = [2];`
- `let massiv = [23, 34, 12, 89];`
- `let massiv = ["а", "б", "в", "г"];`

Основные понятия

Пример объявления многомерного массива

```
let massiv = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9],
];
```

Основные понятия

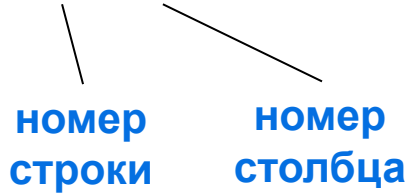
Обращение к элементу массива

- одномерный

  massiv [0];

- многомерный

  massiv [1][2];



Основные понятия

```
let people = [
        ["Tom", 25, false],
        ["Bill", 38, true],
        ["Alice", 21, false]
    ];

console.log(people[0]);
console.log(people[1]);
console.log("Имя: " + people[0][0]);
console.log("Возраст: " + people[0][1]);
```

Tom	25	false
Bill	38	true
Alice	21	false

Основные понятия

```
let people = [
        ["Tom", 25, false],
        ["Bill", 38, true],
        ["Alice", 21, false]
    ];

console.log(people[0]); // ["Tom", 25, false]
console.log(people[1]); // ["Bill", 38, true]
console.log("Имя: " + people[0][0]); // Tom
console.log("Возраст: " + people[0][1]); // 25
```

Основные понятия

Массивы в JavaScript представлены объектом **Array**

Свойство	Описание
length	**Устанавливает или возвращает количество элементов в массиве**
prototype	**Позволяет добавлять свойства и методы к объекту**

Основные понятия

Метод	Описание
concat()	Объединяет массивы и возвращает их объединенную копию
join()	Соединяет все элементы массива в строку
pop()	Удаляет последний элемент массива и возвращает его значение
push()	Добавляет новый элемент в конец массива и возвращает новую длину массива
reverse()	Изменяет порядок следования элементов в массиве на противоположный
shift()	Удаляет первый элемент массива и возвращает его значение

Основные понятия

Метод	Описание
slice()	Извлекает часть элементов массива и создает из них новый массив.
sort()	Сортирует элементы массива.
splice()	Удаляет/добавляет элементы массива.
toString()	Преобразует массив в строку.
unshift()	Добавляет элемент в начало массива и возвращает его новую длину (массива).
valueOf()	Возвращает содержимое массива.

Основные понятия

Ассоциативные массивы:

- **с помощью объекта Map**
- **посредством объектов**

Основные понятия

Ассоциативные массивы:

- **с помощью объекта Map**
- **посредством объектов**

Основные понятия

```
let massiv = new Map();
```

```
let massiv = new Map([
        ['key1', 'value1'],
        ['key2', 'value2'],
        ['key3', 'value3']
    ]);
```

Основные понятия

Метод	Описание
set()	Записывает по ключу key значение value
get()	Возвращает значение по ключу или undefined , если ключ key отсутствует.
has()	Возвращает true , если ключ key присутствует в коллекции, иначе false
delete()	Удаляет элемент по ключу key
clear()	Очищает коллекцию от всех элементов
size()	Возвращает текущее количество элементов

Основные понятия

Перебор элементов массива

- обращение по индексу элемента

  ```
  for (let i = 0; i < massiv.length; i++)
  ```

- обращение непосредственно к элементу

  ```
  for (let elemMassiva of massiv)
  ```

- с использованием метода

  massiv.forEach()

```
massiv.forEach((elemMassiva) => {
    console.log(elemMassiva
})
```

Практическое задание

Практическое задание 1

Вывести значения элементов массива с четными порядковыми номерами, начиная с 0.

```
let massiv = [10, 34, 101, 89, 1002, 5, 103];
```

Практическое задание

Практическое задание 2

Задана таблица, которая связывает типы MIME с расширениями файлов. По введённому расширению файла определить его тип MIME. Если такого расширения в таблице нет вывести "неизвестно".

Практическое задание

Практическое задание 2

Расширение	MIME тип
txt	text/plain
png	image/png
xml	text/xml
pdf	application/pdf
mp3	audio/mpeg
js	application/javascript

Практическое задание

Практическое задание 3

Что будет выведено в результате выполнения следующего кода?

```
let fruits = ["Яблоки", "Груша", "Апельсин", "Мандарин"];
let fruitsNew = fruits;
fruitsNew.push("Банан");
alert(fruits.length);
```

Основные операции со структурами данных

1. Формирование
2. Просмотр
3. Добавление
4. Извлечение
5. Удаление
6. Сдвиг
7. Изменение
8. **Сортировка**
9. **Поиск**

Поиск

1. Поиск экстремумов (минимума, максимума)

2. Поиск заданного значения (местоположения)

Поиск экструмумов

1. Метод полного перебора (поиск минимального, максимального значений)
2. Метод градиентного спуск
3. Метод наискорейшего спуска
4. Метод золотого сечения
5. Метод Фибоначчи

Поиск минимального значения

Начальное значение искомой величины (минимума)

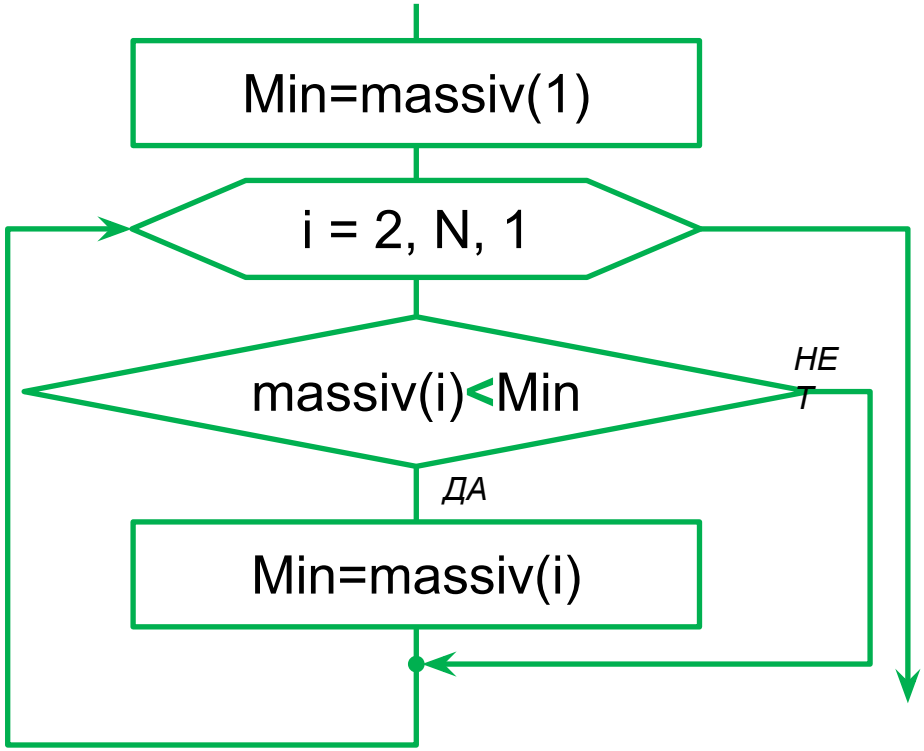
- Должно меняться внутри цикла
- Равно значению первого элемента массива (если данные известны)
- Больше любого возможного значения элемента массива (если значения не известны)

Поиск минимального значения

Условные обозначения

- **N** – количество элементов массива (размерность)
- **Massiv** – массив, в котором содержится N значений
- **Min** – значение минимального элемента
- **i** – порядковый номер текущего элемента массива

Поиск минимального значения




```
let Min = massiv[0];

for (i = 1; i < N; i++) {

    if (massiv[i] < Min) {

        Min = massiv[i];

    }

}
```

Поиск максимального значения

Начальное значение искомой величины (максимума)

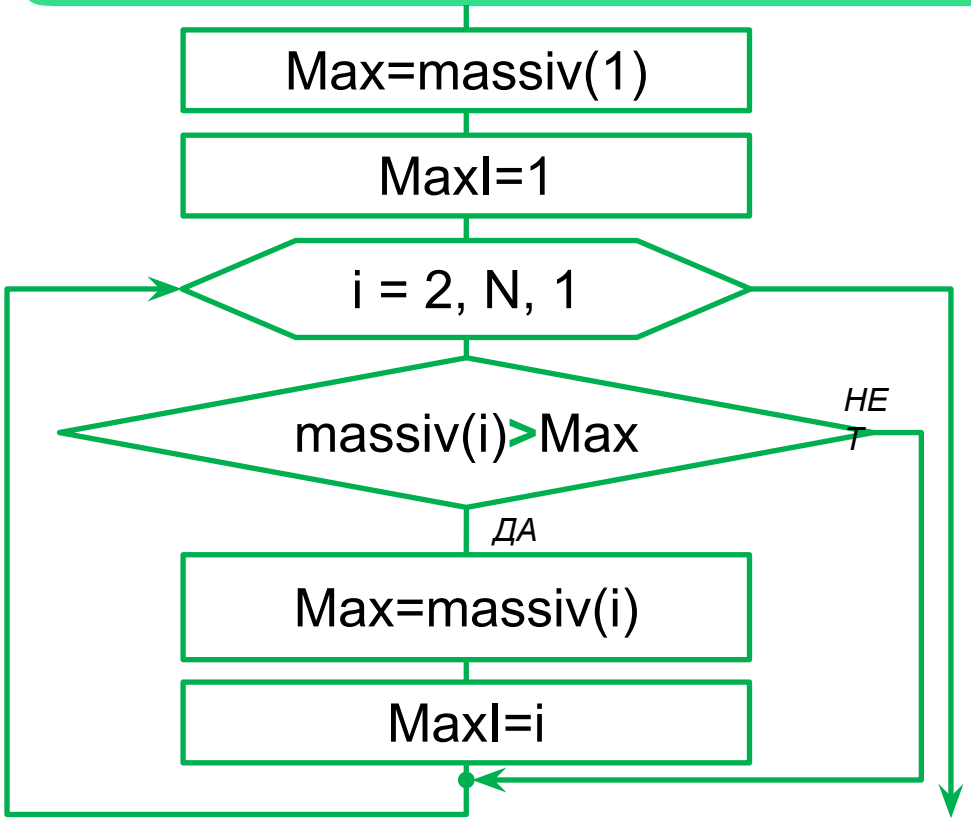
- Должно меняться внутри цикла
- Равно значению первого элемента массива (если данные известны)
- Меньше любого возможного значения элемента массива (если значения не известны)

Поиск максимального значения

Условные обозначения

- **N** – количество элементов массива (размерность)
- **Massiv** – массив, в котором содержится N значений
- **Max** – значение максимального элемента
- **MaxI** – местоположение максимального элемента
- **i** – порядковый номер текущего элемента массива

Поиск максимального значения




```
let Max = massiv[0];
let MaxI = 0;
for (i = 1; i < N; i++) {
    if (massiv[i] > Max) {
        Max = massiv[i];
    MaxI = i;
    }
```

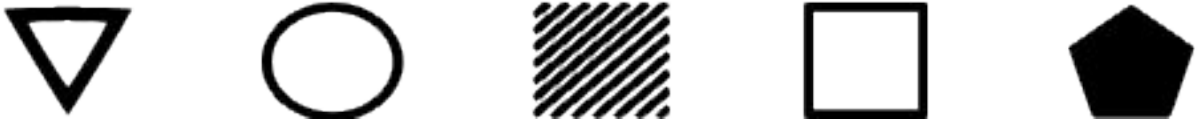
Поиск заданного значения

1. В неупорядоченном массиве
   - ✔ Последовательный (линейный)
2. В упорядоченном массиве
   - ✔ Бинарный (двоичный)
   - ✔ Интерполяционный
   - ✔ Экспоненциальный
3. Поиск текста по заданному шаблону
   - ✔ Алгоритм Кнута – Морриса – Пратта

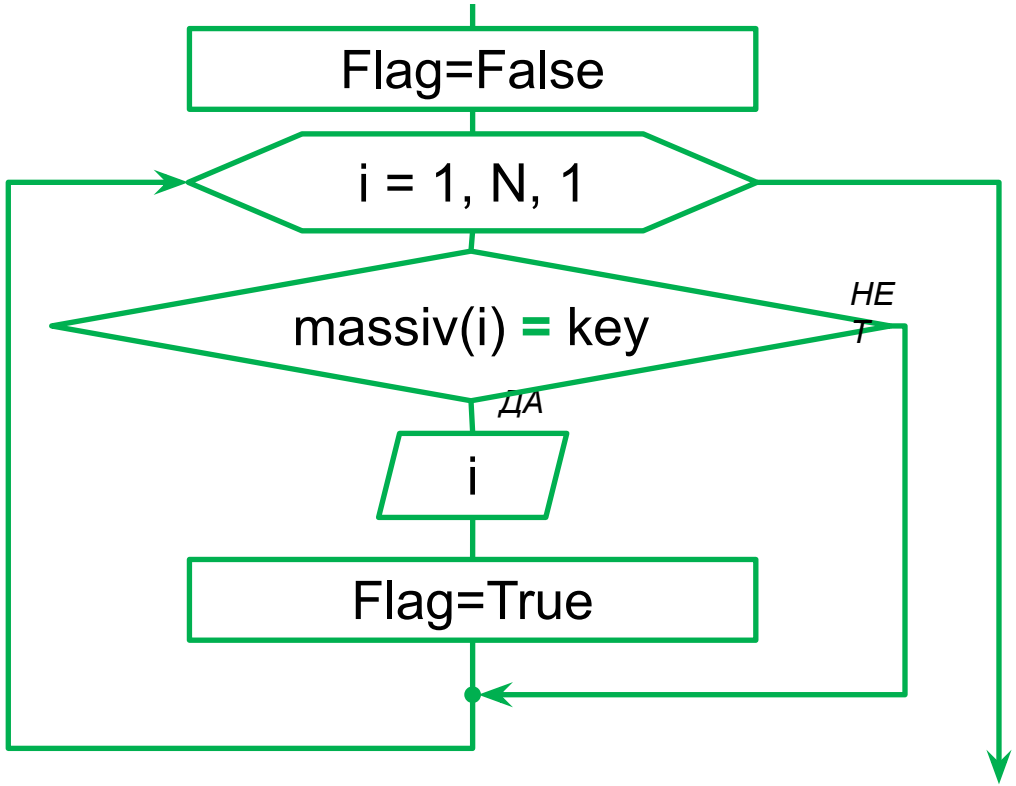
Последовательный поиск

- простейший алгоритм поиска
- нет предварительных условий к состоянию структуры данных
- поиск элемента в заданной структуре данных осуществляется последовательным сравнением искомого с каждым элементом структуры данных
- временная сложность равна O(N) - потребуется N итераций для нахождения элемента
- пространственная сложность равна O(1) - требует всего одну единицу памяти для хранения искомого элемента

Последовательный поиск



Последовательный поиск



- **key** – значение искомого элемента
- **Flag** – логическая переменная (False – элемент не найден, True – найден)
- В программном алгоритме требуется предусмотреть досрочный выход из цикла

Бинарный поиск

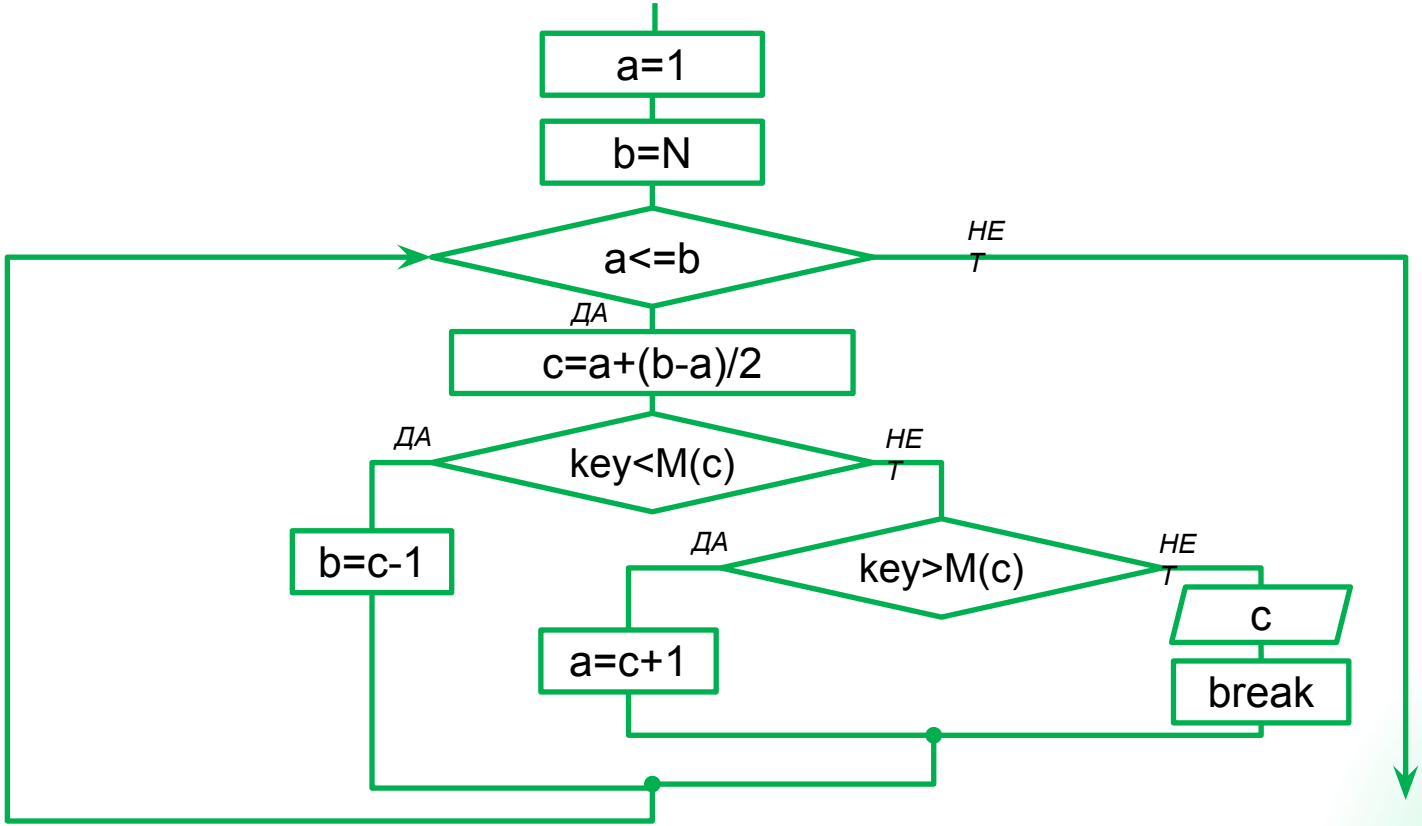
- данные должны быть отсортированы
- алгоритм делит массив данных на равные половины, и с каждой итерацией сравнивает искомый элемент с элементом в середине
- временная сложность равна O(log(N))
- пространственная сложность равна O(1)

Бинарный поиск

шаг \ i	1	2	3	4	5	6	7	8	9	a, b, c
1	1	4	9	16	25	36	49	64	81	a=1 b=9 c=5
2	1	4	9	16	25	36	49	64	81	a=1 b=4 c=2
3	1	4	9	16	25	36	49	64	81	a=3 b=4 c=3
4	1	4	9	16	25	36	49	64	81	a=4 b=4 c=4

- **a** – левая граница интервала для поиска
- **b** – правая граница интервала для поиска
- **c** – середина рассматриваемого интервала

Бинарный поиск



Бинарный поиск

```
let N = 9;
let M = [1, 4, 9, 16, 25, 36, 49, 64, 81];
let a=0;
let b=N-1;
let key = Number(prompt("введите искомое значение"));
while (a<=b) {
    c=Math.round(a+(b-a)/2);
    if (key < M[c]) {
        b=c-1;
    } else if (key > M[c]) {
        a=c+1;
    } else{
        alert(`Элемент ${key} хранится под номером  ${c}`);
        break;
    }
}
```

Интерполяционный поиск

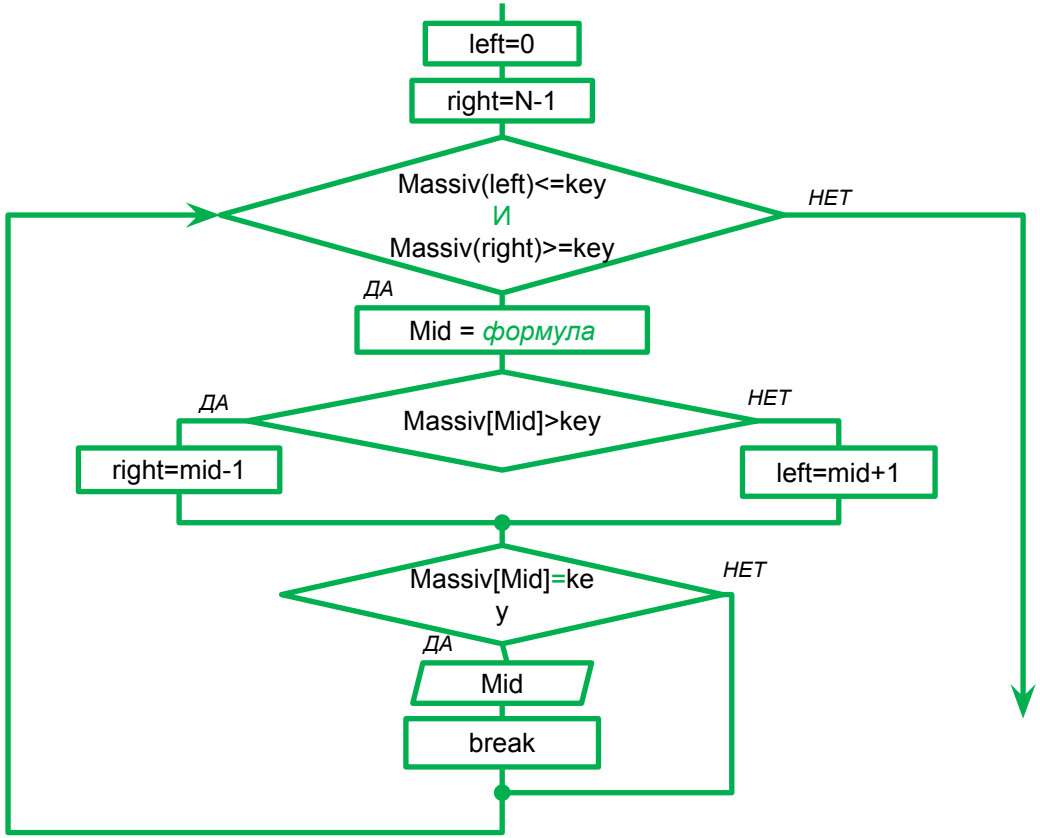
- данные должны быть отсортированы
- полезен для равномерно распределенных в структуре данных
- для поиска элементов в массиве алгоритм использует формулы интерполяции
- эффективнее применять эти формула для больших массивов, в противном случае алгоритм работает как линейный поиск
- временная сложность равна O(log(log(N))) – для равномерного распределения
- пространственная сложность равна O(1)

Интерполяционный поиск

$$Mid = left + \frac{\left(key - Massiv(left)\right) * (right - left)}{Massiv(right) - Massiv(left)}$$

- key – искомый элемент
- $Massiv$ – упорядоченный массив, в котором происходит поиск
- Mid – номер элемента, с которым сравнивается искомый элемент
- $left$ – левая граница области поиска
- $right$ – правая граница области поиска

Интерполяционный поиск



Экспоненциальный поиск

- данные должны быть отсортированы
- используется для поиска элементов путём перехода в экспоненциальные позиции, то есть во вторую степень
- используется с большими массивами, когда бинарный поиск затратен
- ищется сравнительно меньший диапазон и на нём применяется бинарный алгоритм
- временная сложность равна O(log(N)) – для равномерного распределения
- пространственная сложность равна O(1)

Практическое задание

Практическое задание 4

Дана последовательность действительных чисел. Заменить все ее члены, большие заданного числа К, этим числом. Подсчитать количество замен.

Сортировка

Алгоритм сортировки – алгоритм для упорядочивания элементов в массиве.

Ключ сортировки – параметр, по значениям которого требуется упорядочить данные.

Сортировка

Методы сортировки

1. Сортировка подсчётом
2. Сортировка методом прямого включения
3. Сортировка методом прямого выбора
4. Сортировка методом прямого обмена (пузырьковая)
5. Сортировка методом бинарных включений
6. Сортировка перемешиванием
7. Сортировка Шелла
8. Быстрая сортировка

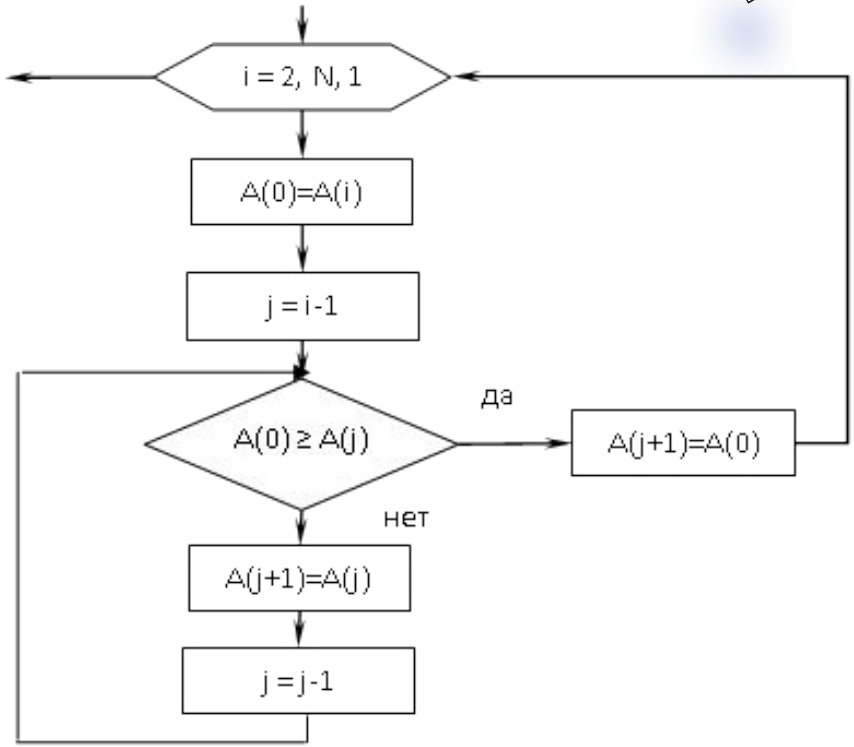
Метод прямого включения

- элементы делятся на уже "готовую" последовательность и "оставшуюся" (не сортированную) часть
- при каждом шаге, из неотсортированной части извлекается элемент и помещается в "готовую" часть, при этом он вставляется на нужное место
- временная сложность равна $O(N^2)$
- пространственная сложность равна $O(1)$

Метод прямого включения

Исходная последовательность		44	55	12	42	94	18	06	67
	A (0)								
I = 2	55	44	55	12	42	94	18	06	67
I = 3	12	44	55	12	42	94	18	06	67
I = 4	42	12	44	55	42	94	18	06	67
I = 5	94	12	42	44	55	94	18	06	67
I = 6	18	12	42	44	55	94	18	06	67
I = 7	06	12	18	42	44	55	94	06	67
I = 8	67	06	12	18	42	44	55	94	67
Результат		06	12	18	42	44	55	67	94

Метод прямого включения



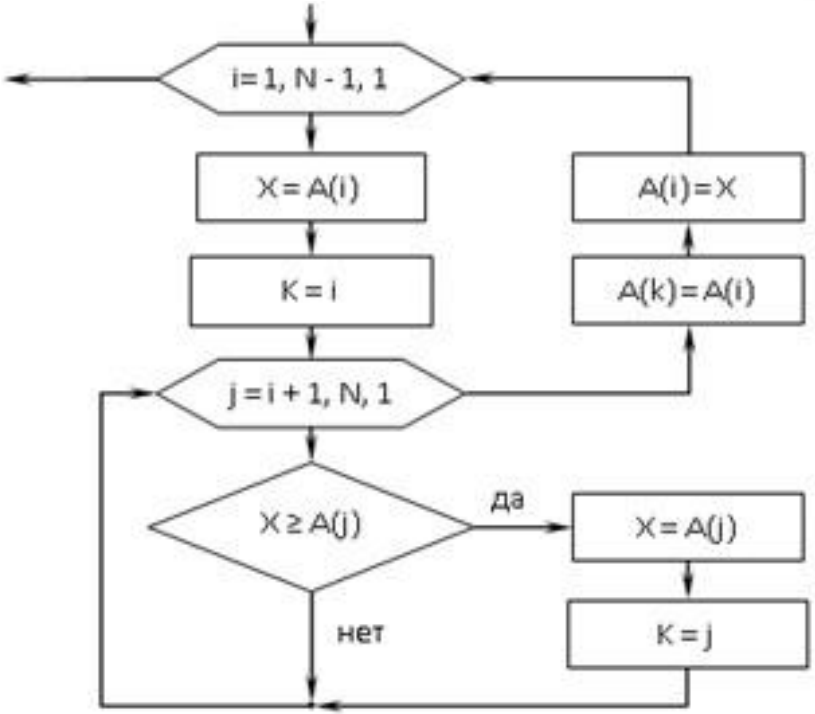
Метод прямого выбора

- элементы делятся на уже "готовую" последовательность и "оставшуюся" (не сортированную) часть
- для поиска одного (минимального) элемента просматривают все элементы неотсортированной части, и этот минимальный элемент помещается как очередной элемент в уже "готовую" часть
- временная сложность равна $O(N^2)$
- пространственная сложность равна $O(1)$

Метод прямого выбора

Исходная последовательность	44	55	12	42	94	18	06	67
I = 1	06	55	12	42	94	18	44	67
I = 2	06	12	55	42	94	18	44	67
I = 3	06	12	18	42	94	55	44	67
I = 4	06	12	18	42	94	55	44	67
I = 5	06	12	18	42	44	55	94	67
I = 6	06	12	18	42	44	55	94	67
I = 7	06	12	18	42	44	55	67	94

Метод прямого выбора



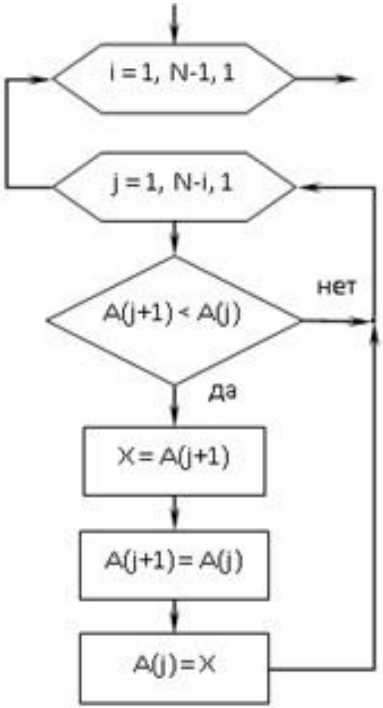
Метод прямого обмена

- основан на сравнении и смене мест для пары соседних элементов и продолжении этого процесса до тех пор, пока не будут упорядочены все элементы
- временная сложность равна $O(N^2)$
- пространственная сложность равна $O(1)$

Метод прямого обмена

Исходная последовательность	I = 1	I = 2	I = 3	I = 4	I = 5	I = 6	I = 7	I = 8
44	44	12	12	12	12	06	06	06
55	12	42	42	18	06	12	12	12
12	42	44	18	06	18	18	18	18
42	55	18	06	42	42	42	42	42
94	18	06	44	44	44	44	44	44
18	06	55	55	55	55	55	55	55
06	67	67	67	67	67	67	67	67
67	94	94	94	94	94	94	94	94

Метод прямого обмена



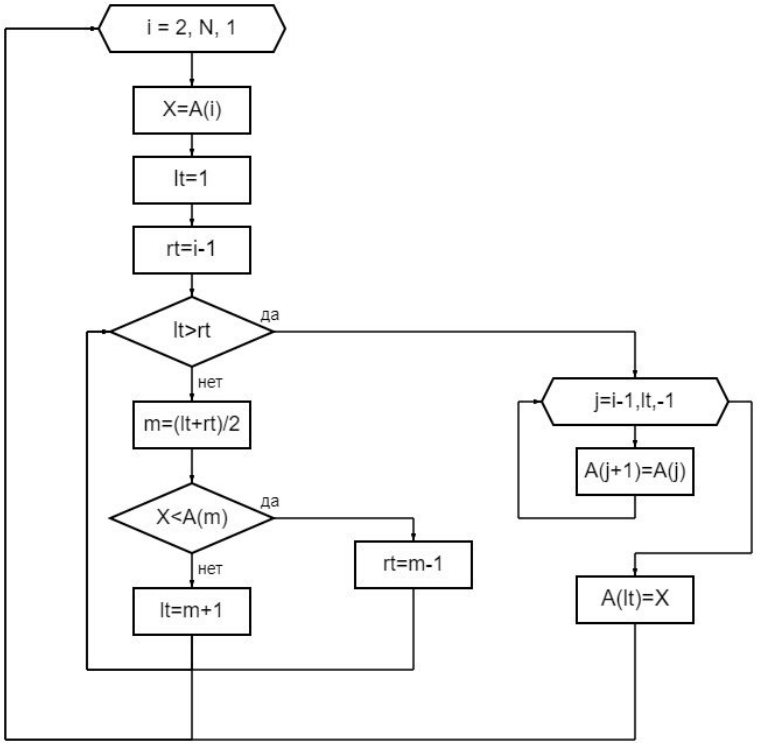
Метод бинарных включений

- элементы делятся на уже "готовую" последовательность и "оставшуюся" (не сортированную) часть
- при каждом шаге, из неотсортированной части извлекается элемент и помещается в "готовую" часть, при этом он вставляется на нужное место
- поиск нужного места осуществляется аналогично бинарному поиску
- временная сложность равна $O(N^2)$
- пространственная сложность равна $O(1)$

Метод бинарных включений

Исходная последовательность	44	55	12	42	94	18	06	67
I = 2	44	55	12	42	94	18	06	67
I = 3	12	44	55	42	94	18	06	67
I = 4	12	42	44	55	94	18	06	67
I = 5	12	42	44	55	94	18	06	67
I = 6	12	18	42	44	55	94	06	67
I = 7	06	12	18	42	44	55	94	67
I = 8	06	12	18	42	44	55	67	94
Результат	06	12	18	42	44	55	67	94

Метод бинарных включений



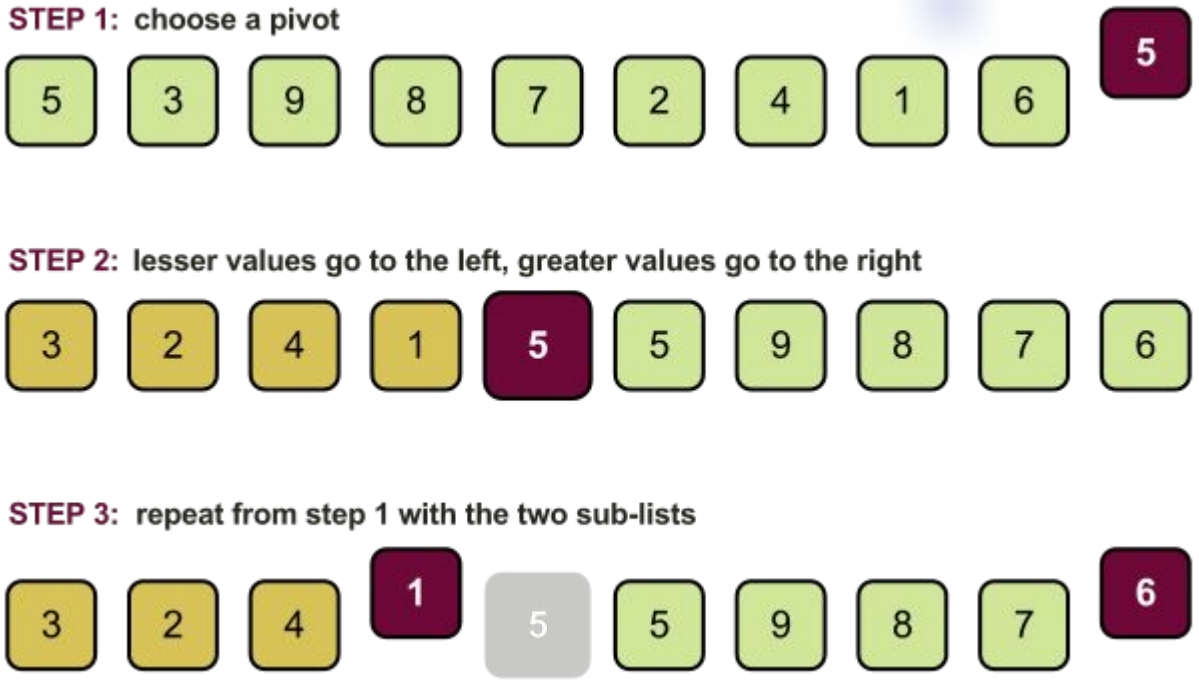
Быстрая сортировка

- выбрать из массива элемент, называемый опорным (пивот)
- сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить массив на два непрерывных отрезка, следующих друг за другом: «элементы меньшие опорного», «равные и большие»
- для полученных отрезков значений выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы

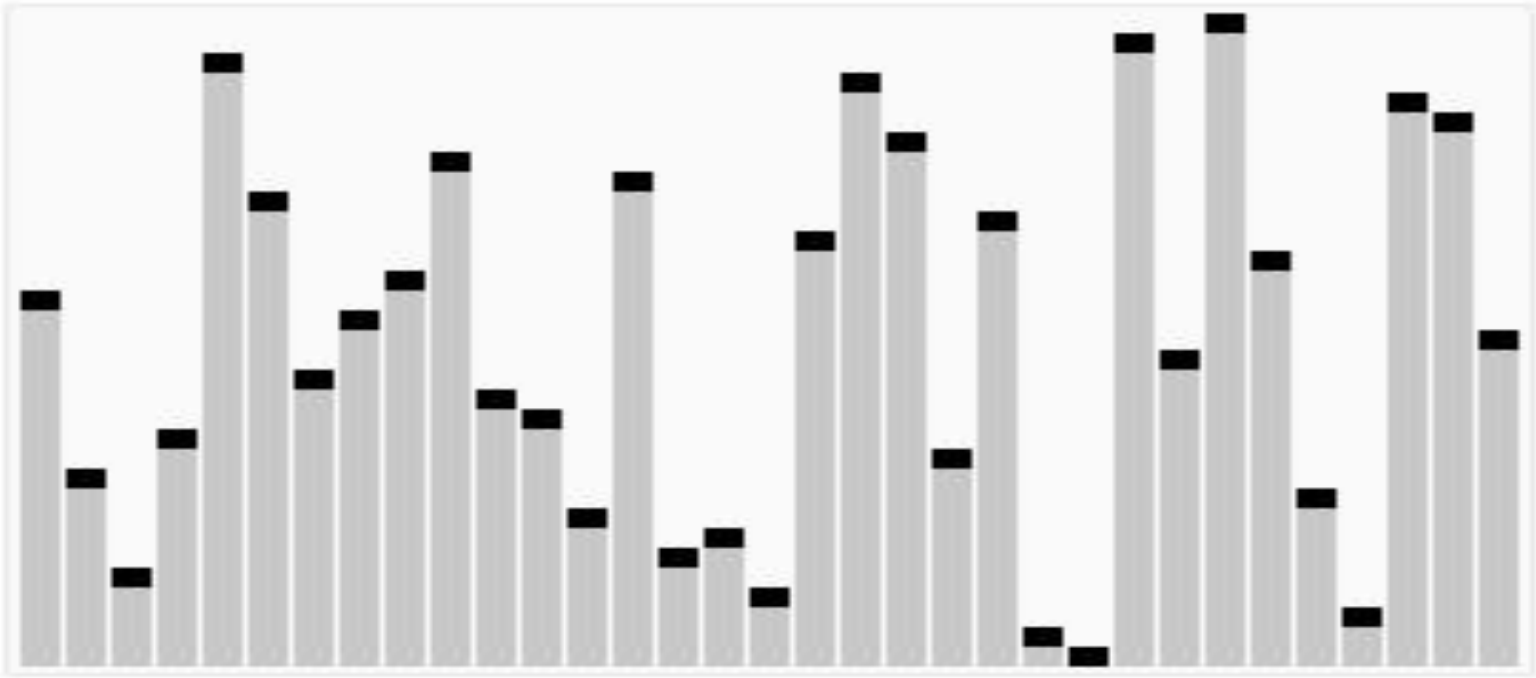
Быстрая сортировка

- временная сложность равна O(N log(N))
- пространственная сложность равна O(1)

Быстрая сортировка



Быстрая сортировка



Практическое задание

Практическое задание 5

Известен список спортсменов и результаты их прыжков в длину. Вывести на экран общий список спортсменов в порядке возрастания результата

Выводы по теме

- познакомились с массивами;
- изучили алгоритмы поиска;
- узнали некоторые алгоритмы сортировки.

Домашнее задание

1. В целочисленной последовательности есть нулевые элементы. Создать массив из номеров этих элементов.

2. Дана последовательность натуральных чисел. Создать массив из четных чисел заданной последовательности. Если таких чисел нет, то вывести сообщение об этом.

Домашнее задание

3. В числовой последовательности определить количество и произведение элементов расположенных между максимальным и минимальным. Максимальный и минимальный элементы в расчёт количества и произведения не включать.

4. Задан числовой массив, состоящий из J элементов. Напечатать положительные числа в порядке убывания, используя метод сортировки бинарными включениями.

Домашнее задание

Дополнительное

1. Вычислить матрицу по заданному выражению $3 \times T^{T} + E + T^{2}$, где T – треугольная квадратная матрица порядка N, E – единичная матрица порядка N, T^{T} - транспонированная матрица T.

Домашнее задание

Дополнительное

2. В массиве целых чисел найти наиболее часто встречающееся число. Если таких чисел несколько, то определить наименьшее из них.

Домашнее задание

Дополнительное

3. Известен список биатлонистов и результаты их стрельбы на двух огневых рубежах (количество попаданий), с учётом того, что на каждом рубеже пять мишеней. Напечатать общий список биатлонистов в порядке убывания общего результата, используя метод сортировки прямым обменом. И список биатлонистов, совершивших не более двух промахов на втором огневом рубеже в порядке убывания количества попаданий на первом огневом рубеже.

Домашнее задание

Дополнительное

4. Системный администратор раз в неделю создаёт архив пользовательских файлов. Однако объём диска, куда он помещает архив, может быть меньше, чем суммарный объём архивируемых файлов. Известно, какой объём занимает файл каждого пользователя. По заданной информации об объёме файлов пользователей и свободном объёме на архивном диске определите максимальное число пользователей, чьи файлы можно сохранить в архиве, а также максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.

Домашнее задание

Дополнительное

Входные данные. S – размер свободного места на диске (натуральное число, не превышающее 100 000), N – количество пользователей (натуральное число, не превышающее 10000, значения объёмов файлов каждого пользователя (все числа натуральные, не превышающие 100).
Результат. Вывести сначала наибольшее число пользователей, чьи файлы могут быть помещены в архив, затем максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.

Домашнее задание

Дополнительное

Пример: S=100, N=4
Объём каждого файла для записи в архив: 80, 30, 50, 40
При таких исходных данных можно сохранить файлы максимум двух пользователей.
Возможные объёмы этих двух файлов 30 и 40, 30 и 50 или 40 и 50.
Наибольший объём файла из перечисленных пар – 50, поэтому ответ для приведённого примера:
2 50
Предполагается, что не все файлы могут быть сохранены на диске, то есть хотя бы для одного файла места не хватит.

Вопросы по теме

1. Каким объектом представлены массивы в JavaScript?
2. Чем отличается линейный поиск от бинарного?
3. На какие части делится последовательность при сортировке методом прямых включений?

Ответы

1. Каким объектом представлены массивы в JavaScript?

   Array

2. Чем отличается линейный поиск от бинарного?

   Бинарный – на отсортированном массиве

3. На какие части делится последовательность при сортировке методом прямых включений?

   "Готовая" и "Неотсортированная"

Анонс следующего занятия

Тема 7. Инструменты программирования.

Профилировщик кода (профайлер).
Система контроля версий. GIT.
Визуальный редактор интерфейса. Инструмент тестирования программного обеспечения.
Фреймворки.
Использование системы контроля версий GIT.

В заключение

1. Какие методы поиска и(или) сортировки Вам бы хотелось дополнительно рассмотреть?

2. Что Вы уже знали?