МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ)»

Институт № 3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 «Вычислительные машины, системы и сети»

Лабораторная работа № 1

по дисциплине «Алгоритмы и обработка данных»

на тему «Алгоритмы поиска»

Выполнил

студенты группы М3О-225БВ-24

Егоров А.В.

Приняли

доцент кафедры 304 Дмитриева Е.А.

Москва
2025

Содержание:

Постановка задачи

Цель работы: изучить основные принципы работы алгоритмов поиска, исследовать их свойства:

- алгоритм **BLS** (Better_Linear_Search);
- алгоритм **SLS** (Sentinel_ Linear_Search);
- алгоритм **OAS** (Ordered_Array_Search);
- алгоритм **BS** (Binary Search).

Задание

Для алгоритмов **BLS** и **SLS** в качестве входного массива использовать одну и ту же последовательность значений (вычисление значений с помощью функции **rand**( )).

Для алгоритмов **OAS** и **BS** – массив предварительно отсортировать по возрастанию.

Оценить длительность поиска для различных значений размеров последовательностей (начиная с 50000 до 300000 элементов массива, провести измерения не менее, чем для 6 разных размерностей).

Для каждой размерности рассматриваются случаи нахождения ключа поиска в начале, в середине и случай отсутствия ключа в массиве.

Для алгоритмов кроме подсчета **времени** поиска, требуется определить сколько раз выполняются операции **сравнения:**

- сравнение ключа с элементом массива (counter1),
- сравнение i < n (i - индекс элемента массива в цикле, n - предельное значение размера массива(counter2).

Все результаты оформить в виде таблиц и графиков.

На графиках, для трех разных вариантов поиска ключа (в начале, середине и отсутствия ключа в массиве) **- только временные характеристики** поиска.

Отчет по лабораторной работе должен содержать:

- задание;
- структурные схемы алгоритмов поиска;
- код;
- результаты работы и результаты анализа – в виде таблиц, графиков;
- выводы об эффективности того или иного алгоритма поиска.

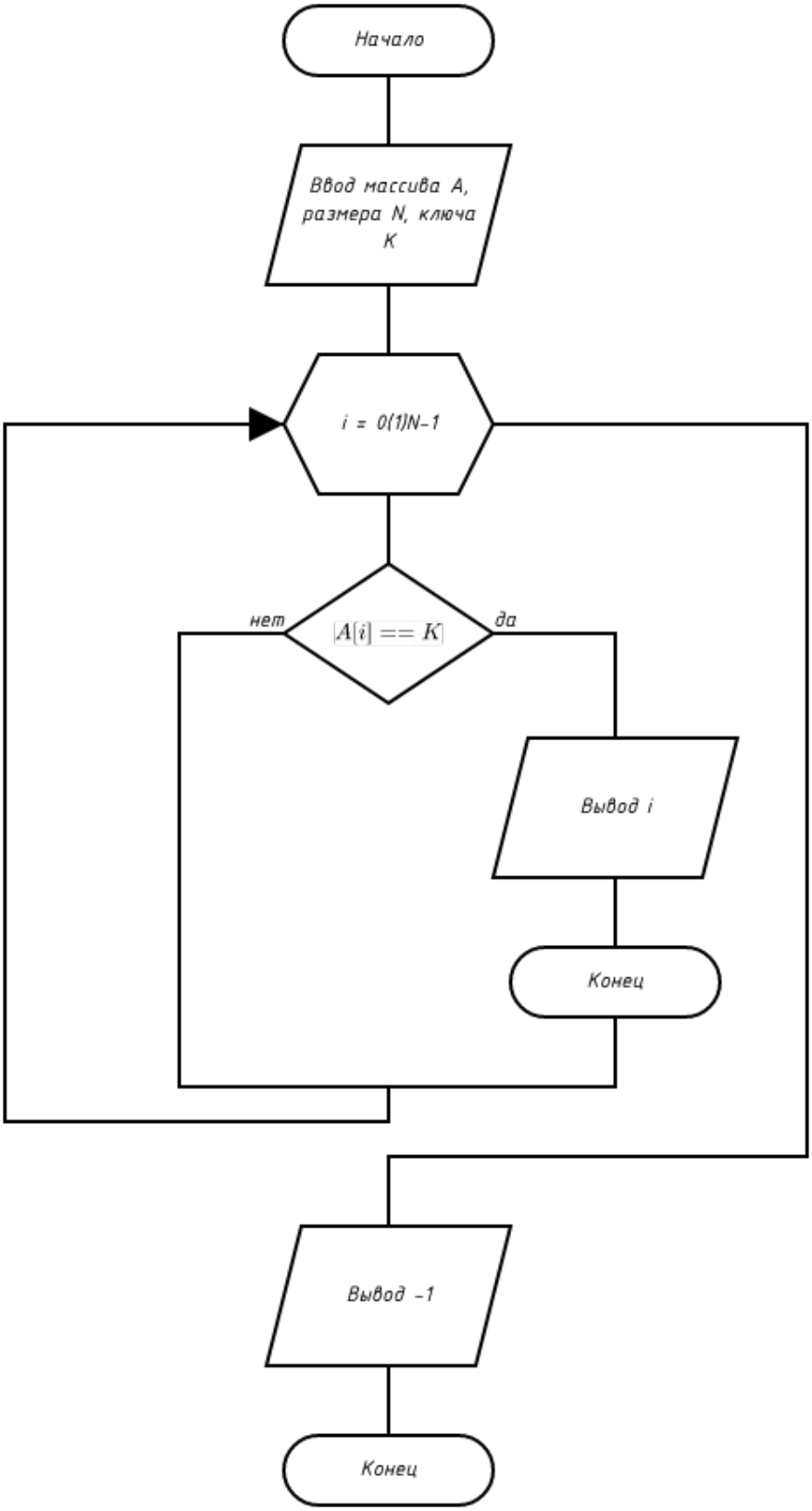
Оценка временных характеристик алгоритмов:

В зависимости от того в какой программной среде выполняется работа и на каком компьютере, иногда необходимо использовать для подсчета времени (**в микросекундах**) специальную библиотеку <chrono>.

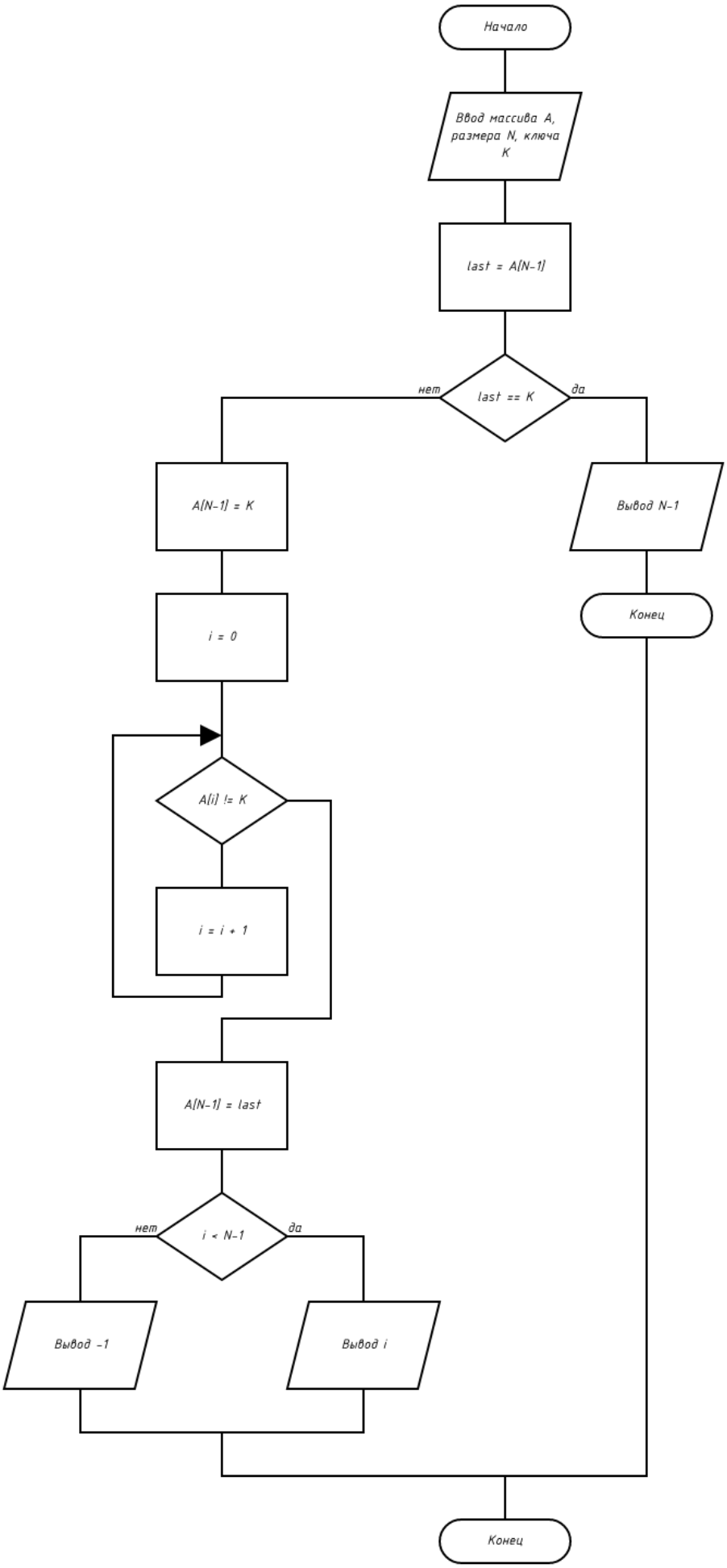
```
#include<chrono>
// подключение библиотеки для вычисления времени работы алгоритмов
. . .
auto begin = std::chrono::steady_clock::now();
// получаем время перед началом формирования последовательности
...
auto end = std::chrono::steady_clock::now();
 // получаем время по окончанию формирования последовательности


auto elapsed_ms = std::chrono::duration_cast<std::chrono::microseconds>(end - begin);                    //получаем  время работы в микросекундах


std::cout << "Время работы алгоритма по формированию последовательности: " << elapsed_ms.count() << " (мкС)" << std::endl;
                                                                    // вывод времени работы
```

Структурные схемы

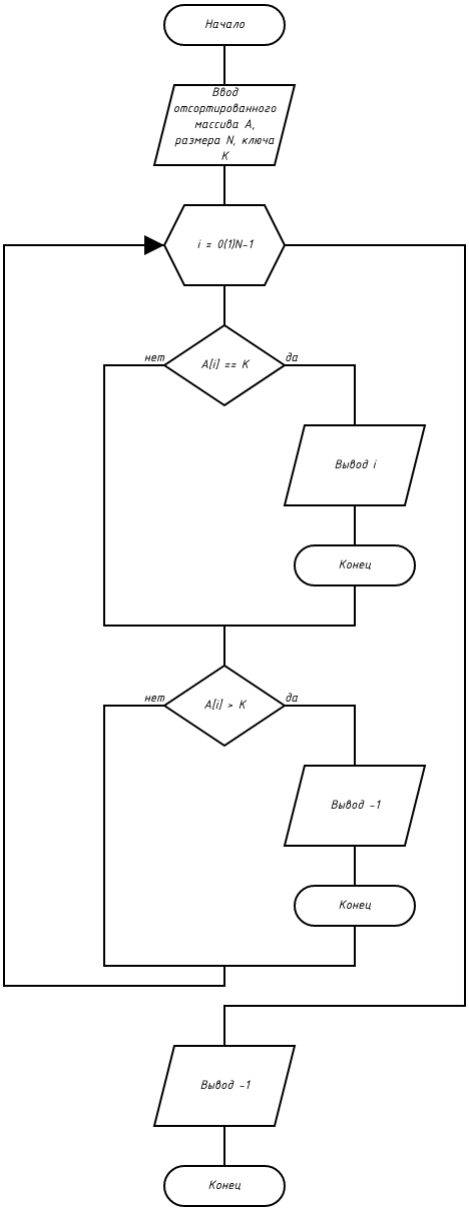
BLS



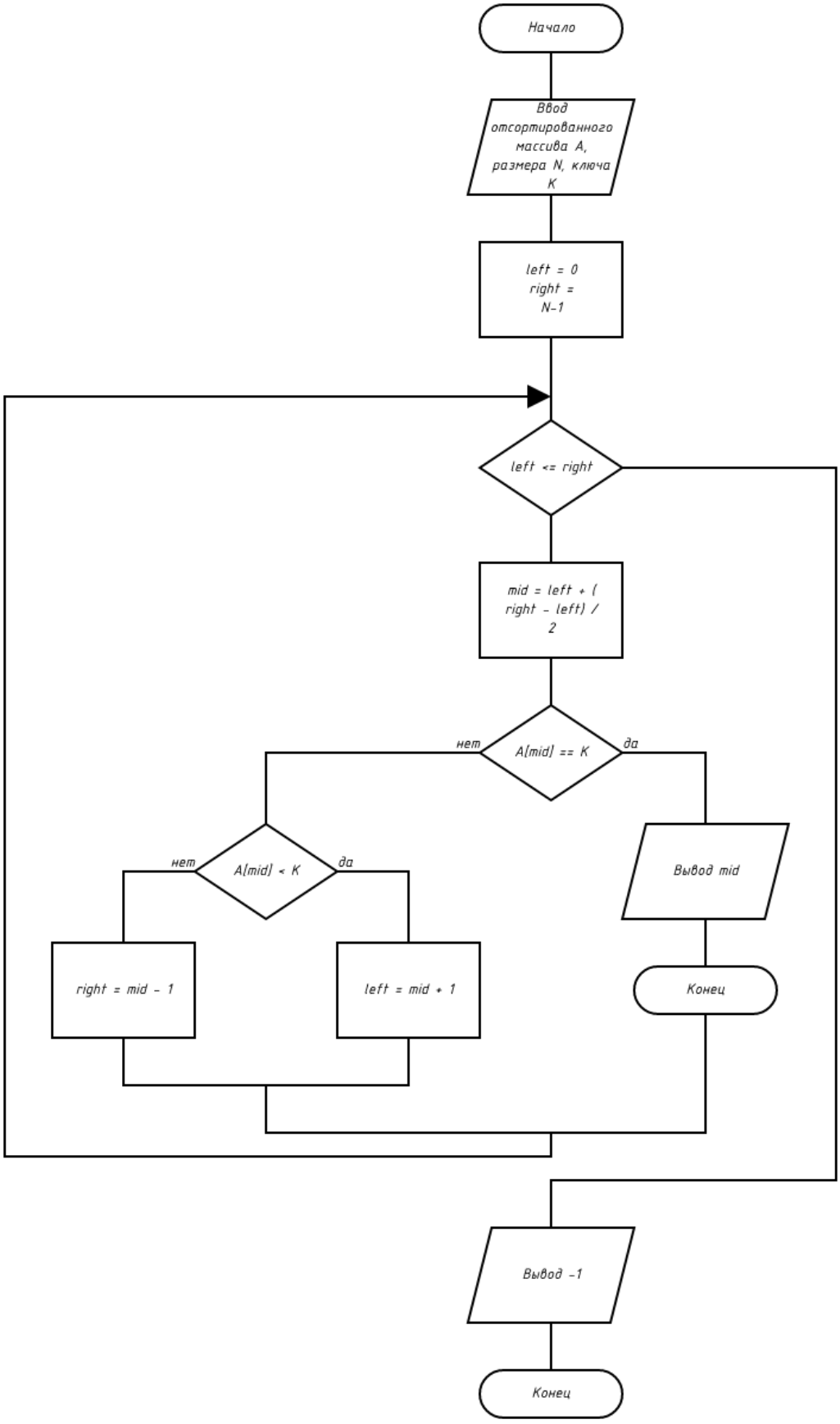
SLS


Начало
Ввод массива A, размера N, ключа K
last = A[N-1]
нет
last == K
да
A[N-1] = K
Вывод N-1
i = 0
Конец
A[i] != K
i = i + 1
A[N-1] = last
нет
i < N-1
да
Вывод -1
Вывод i
Конец

OAS


Начало
Ввод отсортированного массива A, размера N, ключа K
i = 0(1)N-1
нет
A[i] == K
да
Вывод i
Конец
нет
A[i] > K
да
Вывод -1
Конец
Вывод -1
Конец

BS


Начало
Ввод отсортированного массива A, размера N, ключа K
left = 0
right = N-1
left <= right
mid = left + (right - left) / 2
A[mid] == K
нет
да
A[mid] < K
нет
да
right = mid - 1
left = mid + 1
Вывод mid
Конец
Вывод -1
Конец

Код

```
#include <chrono>  // Для измерения времени
#include <cstdlib> // Для rand() и srand()
#include <ctime>   // Для time()
#include <iomanip> // Для форматированного вывода std::setw
#include <iostream>
#include <string> // Для std::string в названии сценария

// Структура для хранения результатов работы алгоритма (без изменений)
struct SearchResult {
  int index;
  long long time_microseconds;
  long long counter1; // Сравнение ключа с элементом массива
  long long counter2; // Сравнение индекса (i < n)
};

// Вспомогательная функция для слияния двух отсортированных подмассивов
void merge(int *arr, int left, int mid, int right) {
  int n1 = mid - left + 1;
  int n2 = right - mid;

  // Создаем временные массивы
  int *L = new int[n1];
  int *R = new int[n2];

  // Копируем данные во временные массивы
  for (int i = 0; i < n1; i++)
    L[i] = arr[left + i];
  for (int j = 0; j < n2; j++)
    R[j] = arr[mid + 1 + j];

  // Сливаем временные массивы обратно в arr[left..right]
  int i = 0;    // Индекс первого подмассива
  int j = 0;    // Индекс второго подмассива
  int k = left; // Индекс объединенного подмассива
  while (i < n1 && j < n2) {
    if (L[i] <= R[j]) {
      arr[k] = L[i];
      i++;
    } else {
      arr[k] = R[j];
      j++;
    }
    k++;
  }

  // Копируем оставшиеся элементы L[], если они есть
  while (i < n1) {
    arr[k] = L[i];
    i++;
    k++;
  }

  // Копируем оставшиеся элементы R[], если они есть
  while (j < n2) {
    arr[k] = R[j];
```

```
    j++;
    k++;
  }

  delete[] L;
  delete[] R;
}

// Основная функция Сортировки слиянием (Merge Sort)
void mergeSort(int *arr, int left, int right) {
  if (left >= right) {
    return; // Базовый случай рекурсии
  }
  int mid = left + (right - left) / 2;
  mergeSort(arr, left, mid);
  mergeSort(arr, mid + 1, right);
  merge(arr, left, mid, right);
}

// --- АЛГОРИТМЫ ПОИСКА ---

SearchResult BLS(const int *arr, int n, int key) {
  long long c1 = 0, c2 = 0;
  auto begin = std::chrono::steady_clock::now();
  for (int i = 0; i < n; ++i) {
    c2++;
    c1++;
    if (arr[i] == key) {
      auto end = std::chrono::steady_clock::now();
      return {i,
              std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                  .count(),
              c1, c2};
    }
  }
  auto end = std::chrono::steady_clock::now();
  return {-1,
          std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
              .count(),
          c1, c2};
}

SearchResult SLS(int *arr, int n, int key) {
  long long c1 = 0, c2 = 0;
  if (n == 0)
    return {-1, 0, 0, 0};
  auto begin = std::chrono::steady_clock::now();
  int last = arr[n - 1];
  c1++;
  if (last == key) {
    auto end = std::chrono::steady_clock::now();
    return {n - 1,
            std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                .count(),
```

```
            c1, c2};
  }
  arr[n - 1] = key;
  int i = 0;
  while (arr[i] != key) {
    c1++;
    i++;
  }
  c1++;
  arr[n - 1] = last;
  auto end = std::chrono::steady_clock::now();
  c2++;
  if (i < n - 1) {
    return {i,
            std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                .count(),
            c1, c2};
  } else {
    return {-1,
            std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                .count(),
            c1, c2};
  }
}

SearchResult OAS(const int *arr, int n, int key) {
  long long c1 = 0, c2 = 0;
  auto begin = std::chrono::steady_clock::now();
  for (int i = 0; i < n; ++i) {
    c2++;
    c1++;
    if (arr[i] == key) {
      auto end = std::chrono::steady_clock::now();
      return {i,
              std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                  .count(),
              c1, c2};
    }
    c1++;
    if (arr[i] > key)
      break;
  }
  auto end = std::chrono::steady_clock::now();
  return {-1,
          std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
              .count(),
          c1, c2};
}

SearchResult BS(const int *arr, int n, int key) {
  long long c1 = 0, c2 = 0;
  auto begin = std::chrono::steady_clock::now();
  int left = 0, right = n - 1;
```

```
  while (left <= right) {
    c2++;
    int mid = left + (right - left) / 2;
    c1++;
    if (arr[mid] == key) {
      auto end = std::chrono::steady_clock::now();
      return {mid,
              std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
                  .count(),
              c1, c2};
    }
    c1++;
    if (arr[mid] < key) {
      left = mid + 1;
    } else {
      right = mid - 1;
    }
  }
  c2++;
  auto end = std::chrono::steady_clock::now();
  return {-1,
          std::chrono::duration_cast<std::chrono::microseconds>(end -
begin)
              .count(),
          c1, c2};
}

// --- ФУНКЦИИ ВЫВОДА И MAIN ---

void run_test_scenario(const std::string &scenario_name, int *unsorted_arr,
                       const int *sorted_arr, int n, int key) {
  std::cout << "  Сценарий: " << scenario_name << " (ключ = " << key << ")
\n";
  std::cout << "  " << std::string(85, '-') << "\n";
  std::cout << "  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа "
               "(counter1) | Сравнения индекса (counter2) |\n";
  std::cout << "  " << std::string(85, '-') << "\n";

  SearchResult res_bls = BLS(unsorted_arr, n, key);
  std::cout << "  | BLS      | " << std::setw(10) << res_bls.index << " | "
            << std::setw(11) << res_bls.time_microseconds << " | "
            << std::setw(26) << res_bls.counter1 << " | " << std::setw(27)
            << res_bls.counter2 << " |\n";

  SearchResult res_sls = SLS(unsorted_arr, n, key);
  std::cout << "  | SLS      | " << std::setw(10) << res_sls.index << " | "
            << std::setw(11) << res_sls.time_microseconds << " | "
            << std::setw(26) << res_sls.counter1 << " | " << std::setw(27)
            << res_sls.counter2 << " |\n";

  SearchResult res_oas = OAS(sorted_arr, n, key);
  std::cout << "  | OAS      | " << std::setw(10) << res_oas.index << " | "
            << std::setw(11) << res_oas.time_microseconds << " | "
            << std::setw(26) << res_oas.counter1 << " | " << std::setw(27)
            << res_oas.counter2 << " |\n";
```

```
  SearchResult res_bs = BS(sorted_arr, n, key);
  std::cout << "  | BS       | " << std::setw(10) << res_bs.index << " | "
            << std::setw(11) << res_bs.time_microseconds << " | "
            << std::setw(26) << res_bs.counter1 << " | " << std::setw(27)
            << res_bs.counter2 << " |\n";
  std::cout << "  " << std::string(85, '-') << "\n\n";
}

int main() {
  setlocale(LC_ALL, "Russian");
  srand(time(0));

  const int num_sizes = 6;
  int sizes[num_sizes] = {50000, 100000, 150000, 200000, 250000, 300000};

  for (int s = 0; s < num_sizes; ++s) {
    int n = sizes[s];

    std::cout << std::string(90, '=') << "\n";
    std::cout << "Тесты для массива размером " << n << " элементов\n";
    std::cout << std::string(90, '=') << "\n\n";

    int *unsorted_array = new int[n];
    int *sorted_array = new int[n];

    for (int i = 0; i < n; ++i) {
      unsorted_array[i] = rand();
      sorted_array[i] = unsorted_array[i];
    }

    auto sort_begin = std::chrono::steady_clock::now();
    mergeSort(sorted_array, 0, n - 1);
    auto sort_end = std::chrono::steady_clock::now();
    auto sort_time = std::chrono::duration_cast<std::chrono::milliseconds>(
        sort_end - sort_begin);
    std::cout << "  (Время на сортировку массива: " << sort_time.count()
              << " мс)\n\n";

    int key_at_start = sorted_array[0];
    int key_in_middle = sorted_array[n / 2];
    int key_not_found = -1;

    run_test_scenario("Ключ в начале", unsorted_array, sorted_array, n,
                      key_at_start);
    run_test_scenario("Ключ в середине", unsorted_array, sorted_array, n,
                      key_in_middle);
    run_test_scenario("Ключ отсутствует", unsorted_array, sorted_array, n,
                      key_not_found);

    delete[] unsorted_array;
    delete[] sorted_array;
  }

  return 0;
}
```

Результат работы программы

```
❯ ./a.out
================================================================================================
Тесты для массива размером 50000 элементов
================================================================================================

  (Время на сортировку массива: 12 мс)

  Сценарий: Ключ в начале (ключ = 17389)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |      12736 |          47 |                      12737 |                      12737 |
  | SLS      |      12736 |          34 |                      12738 |                          1 |
  | OAS      |          0 |           1 |                          1 |                          1 |
  | BS       |          0 |           1 |                         29 |                         15 |
  ------------------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1073257261)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |       1171 |           4 |                       1172 |                       1172 |
  | SLS      |       1171 |           4 |                       1173 |                          1 |
  | OAS      |      25000 |         125 |                      50001 |                      25001 |
  | BS       |      25000 |           2 |                         29 |                         15 |
  ------------------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |         -1 |         195 |                      50000 |                      50000 |
  | SLS      |         -1 |         130 |                      50001 |                          1 |
  | OAS      |         -1 |           1 |                          2 |                          1 |
  | BS       |         -1 |           1 |                         30 |                         16 |
  ------------------------------------------------------------------------------------------
```

```
================================================================================================
Тесты для массива размером 100000 элементов
================================================================================================

  (Время на сортировку массива: 33 мс)

  Сценарий: Ключ в начале (ключ = 88961)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |      61914 |         132 |                      61915 |                      61915 |
  | SLS      |      61914 |          96 |                      61916 |                          1 |
  | OAS      |          0 |           1 |                          1 |                          1 |
  | BS       |          0 |           1 |                         31 |                         16 |
  ------------------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1071373433)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |       7297 |          16 |                       7298 |                       7298 |
  | SLS      |       7297 |          11 |                       7299 |                          1 |
  | OAS      |      50000 |         154 |                     100001 |                      50001 |
  | BS       |      50000 |           1 |                         31 |                         16 |
  ------------------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------------------
  | BLS      |         -1 |         216 |                     100000 |                     100000 |
  | SLS      |         -1 |         159 |                     100001 |                          1 |
  | OAS      |         -1 |           1 |                          2 |                          1 |
  | BS       |         -1 |           2 |                         32 |                         17 |
  ------------------------------------------------------------------------------------------
```

```
================================================================================
Тесты для массива размером 150000 элементов
================================================================================

  (Время на сортировку массива: 53 мс)

  Сценарий: Ключ в начале (ключ = 7254)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |      13536 |          49 |                      13537 |                        13537 |
  | SLS      |      13536 |          53 |                      13538 |                            1 |
  | OAS      |          0 |           1 |                          1 |                            1 |
  | BS       |          0 |           2 |                         33 |                           17 |
  ------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1074156201)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |      21915 |          79 |                      21916 |                        21916 |
  | SLS      |      21915 |          58 |                      21917 |                            1 |
  | OAS      |      75000 |         382 |                     150001 |                        75001 |
  | BS       |      75000 |           1 |                         33 |                           17 |
  ------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |         -1 |         561 |                     150000 |                        150000 |
  | SLS      |         -1 |         392 |                     150001 |                            1 |
  | OAS      |         -1 |           1 |                          2 |                            1 |
  | BS       |         -1 |           1 |                         34 |                           18 |
  ------------------------------------------------------------------------------
```

```
================================================================================
Тесты для массива размером 200000 элементов
================================================================================

  (Время на сортировку массива: 77 мс)

  Сценарий: Ключ в начале (ключ = 1980)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |     179162 |         657 |                     179163 |                        179163 |
  | SLS      |     179162 |         474 |                     179164 |                            1 |
  | OAS      |          0 |           2 |                          1 |                            1 |
  | BS       |          0 |           2 |                         33 |                           17 |
  ------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1074592513)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |     170860 |         616 |                     170861 |                        170861 |
  | SLS      |     170860 |         453 |                     170862 |                            1 |
  | OAS      |     100000 |         496 |                     200001 |                        100001 |
  | BS       |     100000 |           1 |                         33 |                           17 |
  ------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ------------------------------------------------------------------------------
  | BLS      |         -1 |         737 |                     200000 |                        200000 |
  | SLS      |         -1 |         546 |                     200001 |                            1 |
  | OAS      |         -1 |           2 |                          2 |                            1 |
  | BS       |         -1 |           2 |                         34 |                           18 |
  ------------------------------------------------------------------------------
```

```
================================================================================================
Тесты для массива размером 250000 элементов
================================================================================================

  (Время на сортировку массива: 100 мс)

  Сценарий: Ключ в начале (ключ = 1594)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |     111953 |         404 |                     111954 |                       111954 |
  | SLS      |     111953 |         287 |                     111955 |                            1 |
  | OAS      |          0 |           1 |                          1 |                            1 |
  | BS       |          0 |           1 |                         33 |                           17 |
  ---------------------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1073570500)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |      15924 |          58 |                      15925 |                        15925 |
  | SLS      |      15924 |          42 |                      15926 |                            1 |
  | OAS      |     125000 |         692 |                     250001 |                       125001 |
  | BS       |     125000 |           2 |                         33 |                           17 |
  ---------------------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |         -1 |         941 |                     250000 |                       250000 |
  | SLS      |         -1 |         710 |                     250001 |                            1 |
  | OAS      |         -1 |           2 |                          2 |                            1 |
  | BS       |         -1 |           2 |                         34 |                           18 |
  ---------------------------------------------------------------------------------------------
```

```
================================================================================================
Тесты для массива размером 300000 элементов
================================================================================================

  (Время на сортировку массива: 122 мс)

  Сценарий: Ключ в начале (ключ = 8922)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |     140069 |         584 |                     140070 |                       140070 |
  | SLS      |     140069 |         378 |                     140071 |                            1 |
  | OAS      |          0 |           1 |                          1 |                            1 |
  | BS       |          0 |           2 |                         35 |                           18 |
  ---------------------------------------------------------------------------------------------

  Сценарий: Ключ в середине (ключ = 1076234030)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |     135531 |         518 |                     135532 |                       135532 |
  | SLS      |     135531 |         365 |                     135533 |                            1 |
  | OAS      |     150000 |         846 |                     300001 |                       150001 |
  | BS       |     150000 |           1 |                         35 |                           18 |
  ---------------------------------------------------------------------------------------------

  Сценарий: Ключ отсутствует (ключ = -1)
  ---------------------------------------------------------------------------------------------
  | Алгоритм |   Индекс   | Время (мкс) | Сравнения ключа (counter1) | Сравнения индекса (counter2) |
  ---------------------------------------------------------------------------------------------
  | BLS      |         -1 |        1150 |                     300000 |                       300000 |
  | SLS      |         -1 |         826 |                     300001 |                            1 |
  | OAS      |         -1 |           2 |                          2 |                            1 |
  | BS       |         -1 |           2 |                         36 |                           19 |
  ---------------------------------------------------------------------------------------------
```

Анализ тестирований

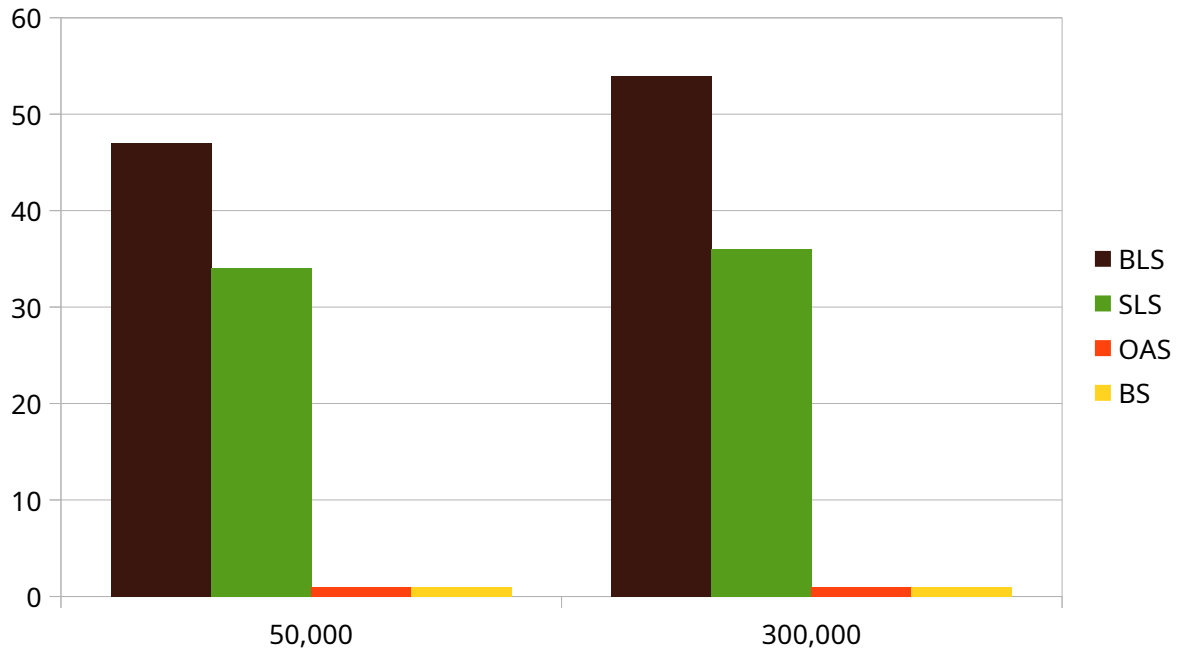
Анализ результатов и эффективности алгоритмов

Анализ производительности проводился на основе трех ключевых сценариев: поиск элемента в начале массива (лучший случай для линейных алгоритмов), в середине (средний случай) и поиск отсутствующего элемента (худший случай).

Сценарий 1: Лучший случай (Ключ находится в начале)

Таблица 4. Сводные результаты для сценария "Ключ в начале"

Размер N	Алгоритм	Время (мкс)	Сравнения ключа (c1)	Сравнения индекса (c2)
50,000	BLS	47	12737	12737
	SLS	34	12738	1
	OAS	1	1	1
	BS	1	29	15
300,000	BLS	54	44608	44608
	SLS	36	44609	1
	OAS	1	1	1
	BS	1	35	19



Анализ:

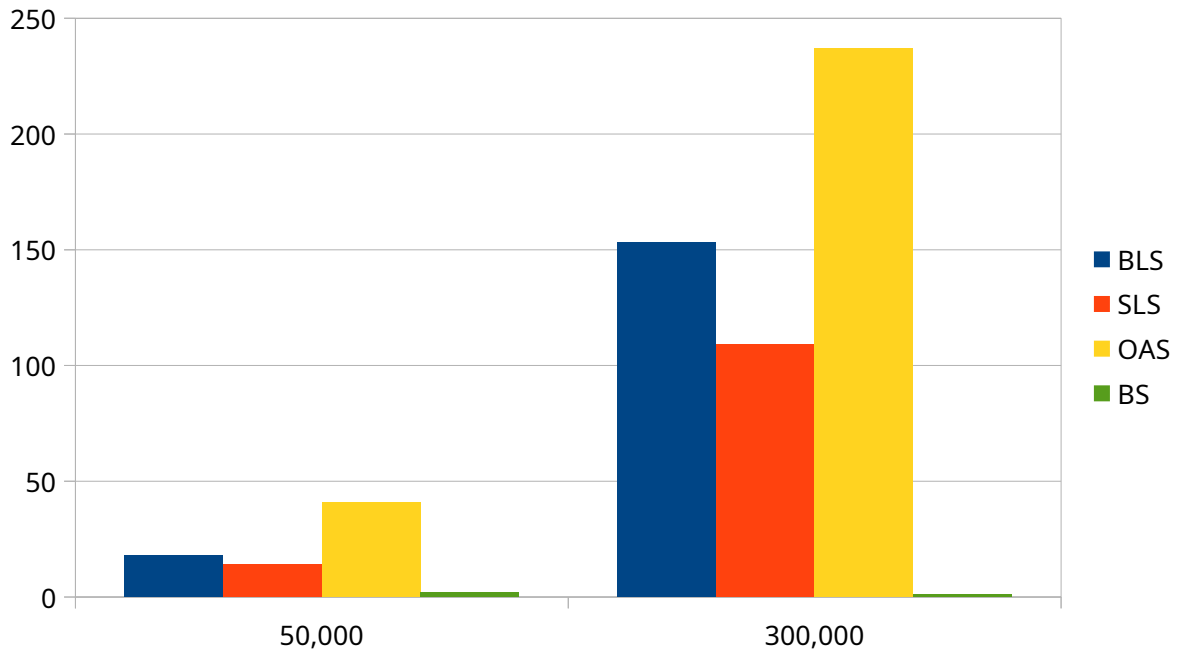
- **BLS и OAS** показывают мгновенный результат, так как находят элемент на первой же итерации. Их временная сложность в этом случае **O(1)** — константная, не зависящая от размера массива.
- **SLS** также практически мгновенен, но выполняет на одно сравнение больше из-за проверки последнего элемента перед установкой "сторожа".
- **Бинарный поиск (BS)**, вопреки ожиданиям, не самый быстрый в этом сценарии. Он не знает, что элемент в начале, и начинает свою работу с деления массива пополам. Он все равно работает очень быстро (логарифмическое время **O(log n)**), но тратит несколько итераций, чтобы "добраться" от центра к краю массива. Важно, что его время выполнения практически не увеличивается с ростом N в 6 раз.

Вывод: В редком случае, когда искомые элементы всегда находятся в начале, простые линейные алгоритмы являются самыми быстрыми.

Сценарий 2: Средний случай (Ключ находится в середине)

Таблица 5. Сводные результаты для сценария "Ключ в середине"

Размер N	Алгоритм	Время (мкс)	Сравнения ключа (c1)	Сравнения индекса (c2)
50,000	BLS	18	15454	15454
	SLS	14	15455	1
	OAS	41	50001	25001
	BS	2	29	15
300,000	BLS	153	123778	123778
	SLS	109	123779	1
	OAS	237	300001	150001
	BS	1	35	18



Анализ:

- Здесь проявляется главное различие в классах сложности. **BLS и SLS** тратят время, прямо пропорциональное половине размера массива. Количество операций для них составляет примерно **N/2**. Это классический пример **линейной сложности O(n)**.

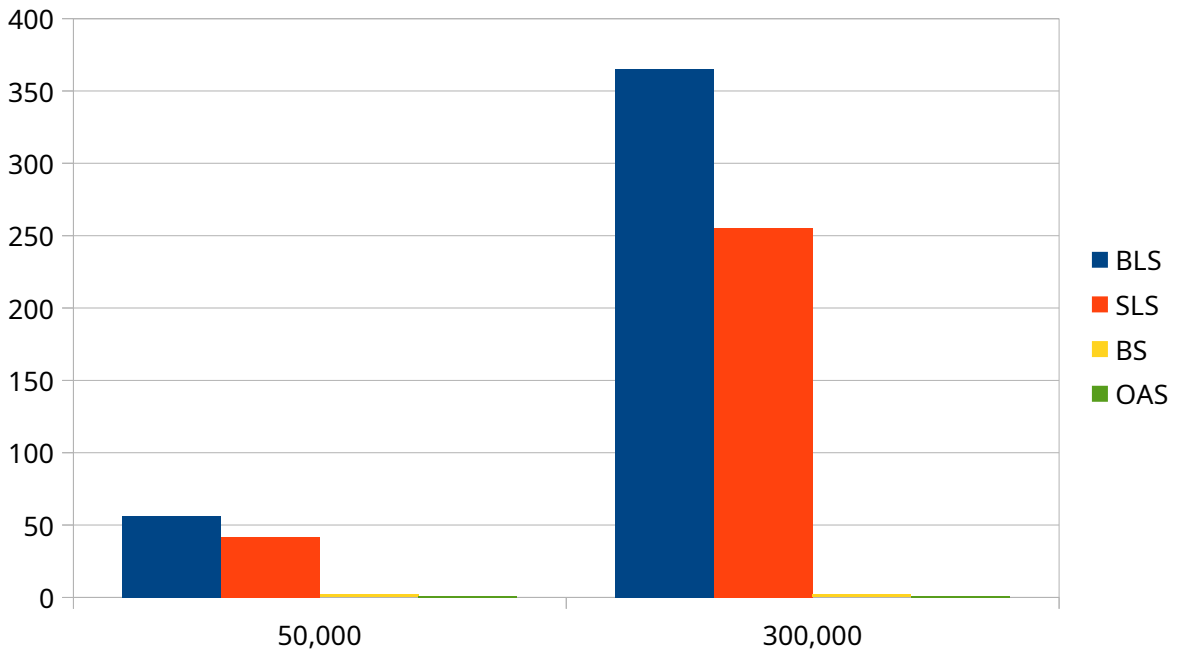
- **SLS** стабильно быстрее **BLS**. Таблицы показывают, почему: counter2 (сравнения индекса) для SLS всегда равен 1, в то время как для BLS он равен N/2. Устранение одной операции из каждой итерации цикла дает ощутимый прирост производительности.
- **Бинарный поиск (BS)** показывает свое колоссальное преимущество. Это для него *абсолютно лучший случай* — он находит элемент с первой же проверки, заглянув в середину. Его сложность здесь **O(1)**. Время выполнения практически нулевое и не зависит от размера массива.

Вывод: Начиная со средних случаев, Бинарный поиск на голову превосходит все линейные аналоги. Среди линейных алгоритмов SLS является более предпочтительным.

Сценарий 3: Худший случай (Ключ отсутствует)

Таблица 6. Сводные результаты для сценария "Ключ отсутствует"

Размер N	Алгоритм	Время (мкс)	Сравнения ключа (c1)	Сравнения индекса (c2)
50,000	BLS	56	50000	50000
	SLS	42	50001	1
	BS	2	2	25001
	OAS	1	30	15
300,000	BLS	365	300000	300000
	SLS	255	300001	1
	BS	2	2	1
	OAS	1	36	19



Анализ:

- **BLS и SLS** вынуждены пройти весь массив до конца. Количество операций равно **N**. Их время выполнения примерно в два раза больше, чем в среднем случае, что подтверждает их **линейную сложность O(n)**. Преимущество **SLS** над **BLS** здесь становится еще более заметным.
- **OAS** показывает худший результат. В цикле он делает два сравнения с ключом (== и >). Поскольку ключ отсутствует и меньше всех элементов, цикл проходит до конца, выполняя **2*N** сравнений ключа, что делает его самым медленным.
- **Бинарный поиск (BS)** снова демонстрирует феноменальную производительность. Чтобы убедиться, что элемента нет, ему требуется пройтись по $\log_2(N)$ элементам. Для N=300,000 это всего лишь ~18 сравнений. Его время практически не отличается от лучшего случая и остается ничтожно малым. Сложность **O(log n)**.

Диаграмма 3: График времени для худшего случая Время (мкс)

Вывод: В худшем случае разрыв в производительности между бинарным и линейными поисками становится максимальным. Бинарный поиск — абсолютный победитель по скорости для отсортированных данных.

Вывод

В ходе работы было установлено, что бинарный поиск (BS) на отсортированных данных на порядки превосходит все линейные методы благодаря своей логарифмической сложности O(log n). Среди линейных алгоритмов O(n), поиск со "сторожем" (SLS) доказал свою эффективность над базовым BLS за счет микрооптимизации цикла. Работа также показала, что не существует универсально лучшего алгоритма, и его выбор зависит от характеристик данных и конкретного сценария использования. Таким образом, практические результаты полностью подтвердили теоретические оценки сложности алгоритмов.