МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра «Информационные системы»

ОТЧЕТ

по практической работе №2

по дисциплине «Программирование»

Тема: одномерные статические массивы

Студент гр. 0323 _____ Балашевич К.Д.

Преподаватель _____ Глущенко А.Г.

Санкт-Петербург

2020

Цель работы.

Изучение структуры одномерных массивов, обработка данных одномерных массивов. Изучение различных видов сортировок. Проведение временной оценки различных действий с массивами.

Основные теоретические положения.

Для работы с большими объёмами однотипных данных в языке c++ используются массивы.

Массив представляет собой индексированную последовательность однотипных элементов с заранее определенным количеством элементов.

Элементы массива в c++ нумеруются с нуля, однако при объявлении массива его размерность указывается с помощью числа элементов, указанного в квадратных скобках.

Стандарт c++ не поддерживает использование массивов с не предопределённым количеством элементов, хотя некоторые компиляторы и реализуют подобную возможность.

Хотя массивы можно передавать в функции как аргументы (причём, при описании функции размер массива указывать не обязательно для большинства компиляторов), использовать массив как возвращаемое значение функций стандарт c++ не позволяет (кроме возвращения в составе других структур данных и использования классов).

Для работы с массивами часто необходимо использовать алгоритмы сортировки.

Сортировка - это процесс перегруппировки однотипных элементов структуры данных в некотором определенном порядке. Цель сортировки - облегчить последующие поиск, обновление, исключение, включение элементов в структуру данных. Эффективность алгоритма сортировки может зависеть от ряда факторов, таких, как: количество сортируемых элементов; диапазон значений элементов; степень первоначальной отсортированности элементов;

сложность алгоритма; время сортировки; количество произведенных сравнений; место размещения элементов в памяти и т. д

Среди простейших алгоритмов сортировки можно выделить:

1) Пузырьковая сортировка. Суть метода заключается в сравнивании попарном сравнении элементов и последующем обмене. Таким образом, если следующий элемент меньше текущего, то они меняются местами, максимальный элемент массива постепенно смещается в конец массива, а минимальный – в начало. Один полный проход по массиву может гарантировать, что в конце массива находится максимальный элемент.
2) Шейкер-сортировка. Принцип работы этой сортировки аналогичен bubble sort: попарное сравнение элементов и последующий обмен местами. Но имеется существенное отличие. Как только максимальный элемент становится на свое место, алгоритм не начинает новую итерацию с первого элемента, а запускает сортировку в обратную сторону. Алгоритм гарантирует, что после выполнения первой итерации, минимальный и максимальный элемент будут в начале и конце массива соответственно.
3) Сортировка расчёской. Основная идея «расчёски» в том, чтобы первоначально брать достаточно большое расстояние между сравниваемыми элементами и по мере упорядочивания массива сужать это расстояние вплоть до минимального. Таким образом мы как бы причёсываем массив, постепенно разглаживая на всё более аккуратные пряди. Математически доказано, что для большей эффективности, первоначально расстояние между элементами должно быть равно длине массива, а с каждой итерацией оно должно уменьшаться на величину, приблизительно равную 1.247.
4) Сортировка вставками. Сперва перебираются элементы в неотсортированной части массива, а потом каждый элемент

вставляется в отсортированную часть массива на то место, где он должен находится.

Постановка задачи.

Необходимо написать программу, которая:

1) Создает целочисленный массив размерности N = 100. Элементы массивы должны принимать случайное значение в диапазоне от -99 до 99.

2) Отсортировать заданный в пункте 1 элементы массива шейкер-сортировкой. Определить время, затраченное на сортировку, используя библиотеку chrono.

3) Найти максимальный и минимальный элемент массива. Подсчитайте время поиска этих элементов в отсортированном массиве и неотсортированном, используя библиотеку chrono.

4) Выводит среднее значение (если необходимо, число нужно округлить) максимального и минимального значения. Выводит все числа, которые равны этому значению.

5) Выводит количество элементов в отсортированном массиве, которые меньше числа a, которое инициализируется пользователем.

6) Выводит количество элементов в отсортированном массиве, которые больше числа b, которое инициализируется пользователем.

7) Выводит информацию о том, есть ли введенное пользователем число в отсортированном массиве. Реализуйте алгоритм бинарного поиска. Сравните скорость его работы с обычным перебором.

8) Меняет местами элементы массива, индексы которых вводит пользователь. Выведите скорость обмена, используя библиотеку chrono.

Должна присутствовать возможность запуска каждого пункта многократно.

Сделать вывод по проделанной работе.

Выполнение работы.

Для решения поставленной задачи была написана программа на языке C++. Итоговый код программы представлен в приложении А.

Было проведено тестирование программы с использование различных ЭВМ и компиляторов. Результаты тестирования представлены в приложении Б. При этом были получены результаты, соответствующие расчётным.

Выводы.

Различные алгоритмы работы с массивами обладают различной эффективностью. При написании программ для ЭВМ надлежит подходить к вопросу выбора используемого алгоритма ответственно применимо к каждому конкретному случаю.

ПРИЛОЖЕНИЕ А
ПОЛНЫЙ КОД ПРОГРАММЫ

```
#include <iostream>
#include <climits>

#include <iostream>
#include <chrono>
#include <windows.h>
#include <ctime>
#define N 100

using namespace std;
using namespace chrono;


int mainMenu (int array[]);
void printArray (int array[]);
void generateArray (int array[]);
int binarySearch (int array[], int n);
double bubbleSort (int array[]);
double shakerSort (int array[]);
double combSort (int array[]);
double insertionSort (int array[]);
double searchMinMax (int array[], int elements[2]);
double searchMinMaxSorted (int array[], int elements[2]);
void printMean (int array[]);
void numberElementsLeft(int array[]);
void numberElementsRight(int array[]);
int search (int array[], int n);
void changeElements (int array[]);

int main ()
  {
    //ВНИМАНИЕ!!!  В  СЛУЧАЕ  ПРОБЛЕМ  С  ОТОБРАЖЕНИЕМ  МЕНЮ
РАСКОММЕНТИРУЙТЕ ДРУГУЮ СТРОЧКУ SETLOCALE!!!
    //setlocale(LC_ALL, "ru_RU.utf8");
    setlocale(0, "");
    //setlocale (LC_ALL, "Russian");
    srand(time (0));
    int array[N];
    generateArray(array);
    int exit=1;
```

```
          while (exit) exit=mainMenu (array);
          return 0;
        }

      int mainMenu (int array[])
        {
          system("cls");
          cout << "Текущее состояние массива:" << endl << endl;
          printArray (array);
          cout << endl;
          cout << endl << "Главное меню:" << endl << "1) Отсортировать
массив пузырьк"
                  "овой сортировкой." << endl << "2) Отсортировать массив
шейкер-сорт"
                  "ировкой."  <<  endl  <<  "3)  Отсортировать  массив
сортировкой расчёско"
                  "й." << endl << "4) Отсортировать массив сортировкой
вставками." <<
                  endl << "5) Найти минимальный и максимальный элементы
массива (1 с"
                  "пособ)." << endl << "6) Найти минимальный и максимальный
элементы"
                  " массива (2 способ, только отсортированный массив)." <<
endl << "7"
                  ") Вывести среднее значение минимального и максимального
элементов"
                  " массива." << endl << "8) Вывести число элементов в
массиве, мень"
                  "ших  числа,  вводимиго  пользователем  (только
отсортированный массив)"
                  "." << endl << "9) Вывести число элементов в массиве,
меньших числ"
                  "а,  вводимого  пользователем  (только  отсортированный
массив)." <<
                  endl << "10) Проверить, есть ли введённое число в массиве
с помощью"
                  " бинарного поиска (только отсортированный массив)." <<
endl << "11"
                  ") Проверить, есть ли введённое число в массиве с помощью
перебора."
                  << endl << "12) Поменять местами элементы массива." <<
endl << "0)"
                  "  Выход  из  программы."  <<  endl  <<  "Введите  желаемый
вариант: ";
```

```
          int choise;
          cin >> choise;
          switch (choise)
            {
              case 0:
                return 0;
              case 1:
                {
                  system ("cls");
                  cout << "Время, затраченное на сортировку: " <<
bubbleSort(array) << ".";
                  Sleep(2000);
                  return 1;
                }
              case 2:
                {
                  system ("cls");
                  cout << " Время, затраченное на сортировку " <<
shakerSort(array) << ".";
                  Sleep(2000);
                  return 1;
                }
              case 3:
                {
                  system ("cls");
                  cout << " Время, затраченное на сортировку " <<
combSort(array) << ".";
                  Sleep(2000);
                  return 1;
                }
              case 4:
                {
                  system ("cls");
                  cout << " Время, затраченное на сортировку " <<
insertionSort(array) << ".";
                  Sleep(2000);
                  return 1;          }
              case 5:
                {
                  int elements[2];
                  double searchTime=searchMinMax(array, elements);
                  cout << endl << "Минимальный элемент равен: " <<
elements[0] <<
```

```
                        endl << "Максимальный элемент равен: " <<
elements[1] <<
                        endl << "На поиск потрачено времени: " <<
searchTime << ".";
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 6:
              {
                int elements[2];
                double searchTime=searchMinMaxSorted(array, elements);
                cout << endl << "Минимальный элемент равен: " <<
elements[0] <<
                        endl << "Максимальный элемент равен: " <<
elements[1] <<
                        endl << "На поиск потрачено времени: " <<
searchTime << ".";
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 7:
              {
                printMean(array);
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;

              }
            case 8:
              {
                numberElementsLeft(array);
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 9:
              {
                numberElementsRight(array);
```

```
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 10:
              {
                int n;
                cout << endl << "Введите число: ";
                cin >> n;
                auto begin = system_clock::now();
                int result=binarySearch(array, n);
                auto end = system_clock::now();
                duration<double> wastedTime = end - begin;
                cout << (result?"В массиве есть такое число.":"В массиве
такого числа нет.") << endl;
                cout   <<   "Время,   затраченное   на   поиск:   "   <<
wastedTime.count() <<".";
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 11:
              {
                int n;
                cout << endl << "Введите число: ";
                cin >> n;
                auto begin = system_clock::now();
                int result=search(array, n);
                auto end = system_clock::now();
                duration<double> wastedTime = end - begin;
                cout << (result?"В массиве есть такое число.":"В массиве
такого числа нет.") << endl;
                cout   <<   "Время,   затраченное   на   поиск:   "   <<
wastedTime.count() <<".";
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            case 12:
              {
                auto begin = system_clock::now();
```

```
                changeElements(array);
                auto end = system_clock::now();
                duration<double> wastedTime = end - begin;
                cout << "Время, затраченное на поиск: " <<
wastedTime.count() <<".";
                cout << endl << "Возврат в главное меню произойдёт в
течении 10 секунд.";
                Sleep(10000);
                return 1;
              }
            default:
              {
                system("cls");
                cout << "Некоррекный ввод!";
                Sleep(1000);
                return 1;
              }
          }
      }

    void printArray (int array[])
      {
        for (int i=0; i<N; i++)
          {
            if (!(i%10)) cout << endl;
            cout << array[i] << "\t";
          }
      }

    void generateArray (int array[])
      {
        for (int i=0; i<N; i++)
          {
            array[i]=(rand()%199)-99;
          }
      }

    double bubbleSort (int array[])
      {
        auto begin = system_clock::now();
        int sorted=0;
        int currentMaxUnsorted=N-1;
        while (!sorted)
          {
```

```
        sorted=1;
        for (int i=0; i<currentMaxUnsorted; i++)
          {
            if (array[i]-array[i+1]>0)
              {
                array[i] = array[i] + array[i+1];
                array[i+1] = array[i] - array[i+1];
                array[i] = array[i] - array[i+1];
                sorted=0;
              }
          }
        currentMaxUnsorted--;
        }
    auto end = system_clock::now();
    duration<double> wastedTime = end - begin;
    return wastedTime.count();
  }

double shakerSort (int array[])
  {
    auto begin = system_clock::now();
    int sorted=0;
    int currentMaxUnsorted=N-1;
    int currentMinUnsorted=0;
    while (!sorted)
      {
        sorted=1;
        for (int i=currentMinUnsorted; i<currentMaxUnsorted; i++)
          {
            if (array[i]-array[i+1]>0)
              {
                array[i] = array[i] + array[i+1];
                array[i+1] = array[i] - array[i+1];
                array[i] = array[i] - array[i+1];
                sorted=0;
              }
          }
        currentMaxUnsorted--;
        for (int i=currentMaxUnsorted; i>currentMinUnsorted; i--)
          {
            if (array[i]-array[i-1]<0)
              {
                array[i] = array[i] + array[i-1];
                array[i-1] = array[i] - array[i-1];
```

```
                array[i] = array[i] - array[i-1];
                sorted=0;
              }
          }
        currentMinUnsorted++;
        }
    auto end = system_clock::now();
    duration<double> wastedTime = end - begin;
    return wastedTime.count();
  }

double combSort (int array[])
  {
    auto begin = system_clock::now();
    for (int step=N-1; step; step/=1.2473309)
      {
        for (int i=0; i+step-N; i++)
          {
            if (array[i]-array[i+step]>0)
              {
                array[i] = array[i] + array[i+step];
                array[i+step] = array[i] - array[i+step];
                array[i] = array[i] - array[i+step];
              }
          }
      }
    auto end = system_clock::now();
    duration<double> wastedTime = end - begin;
    return wastedTime.count();
  }

double insertionSort (int array[])
  {
    auto begin = system_clock::now();
    for(int i=1; i<N; i++)
      {
        int j= i;
        int buffer = array[i];
        while(j)
          {
            if (buffer>array[j-1]) break;
            array[j] = array[j-1];
            j = j-1;
          }
```

```
        array[j] = buffer;
      }
    auto end = system_clock::now();
    duration<double> wastedTime = end - begin;
    return wastedTime.count();
  }

int binarySearch (int array[], int n)
  {
    int numberElements=0;
    int minRange=0;
    int maxRange=N-1;
    int half;
    while (1)
      {
        half=(minRange+maxRange)/2;
        if (array[half]==n)
          {
            break;
          }
        else
          {
            if (array[half]<n) minRange=half;
            if (array[half]>n) maxRange=half;
          }
        if ((maxRange-minRange)<=1) break;
      }
    if (array[half]-n)
      {
        return numberElements;
      }
    else
      {
        numberElements=1;
        return numberElements;
      }
  }

double searchMinMax (int array[], int elements[2])
  {
    auto begin = system_clock::now();
    elements[0]=array[0];
    elements[1]=array[N-1];
    for (int i=N-1; i>-1; i--)
```

```
        {
          if (elements[1]<array[i]) elements[1]=array[i];
        }
      for (int i=0; i<N; i++)
        {
          if (elements[0]>array[i]) elements[0]=array[i];
        }
      auto end = system_clock::now();
      duration<double> wastedTime = end - begin;
      return wastedTime.count();
    }

  double searchMinMaxSorted (int array[], int elements[2])
    {
      auto begin = system_clock::now();
      elements[0]=array[0];
      elements[1]=array[N-1];
      auto end = system_clock::now();
      duration<double> wastedTime = end - begin;
      return wastedTime.count();
    }

  void printMean (int array[])
    {
      int elements[2];
      searchMinMax(array, elements);
      int mean=(elements[0]+elements[1])/2;
      cout << endl << "Среднее значение: " << mean << endl << "Индексы
элементов, равных этому значению: ";
      int isHere=0;
      for (int i=0; i<N; i++)
        {
          if (array[i]==mean)
            {
              cout << i+1 << " ";
              isHere++;
            }
        }
      if (!(isHere))
        {
          cout << "такого элемента в массиве нет!" << endl;
        }
      else
        {
```

```
            cout << endl << "Всего в массиве таких элементов: " << isHere
<< "." << endl;
          }
      }

    void numberElementsLeft(int array[])
      {
        cout << "Введите число: ";
        int n, result=0;
        cin >> n;
        int count=0;
        for (int i=0; i<N; i++)
          {
            if (array[i]<n)
              {
                count++;
                result=i+1;
              }
            else break;
          }
        cout << "Элементов меньше " << n << " в массиве: " << result <<
".";
      }

    void numberElementsRight(int array[])
      {
        cout << "Введите число: ";
        int n, result=0;
        cin >> n;
        int count=0;
        for (int i=N-1; i>-1; i--)
          {
            if (array[i]>n)
              {
                count++;
                result=N-i;
              }
            else break;
          }
        cout << "Элементов больше " << n << " в массиве: " << result <<
".";
      }

    int search (int array[], int n)
```

```
  {
    for (int i=0; i<N; i++)
      {
        if (array[i]==n) return 1;
      }
    return 0;
  }

void changeElements (int array[])
  {
    int a, b;
    cout << endl << "Введите индекс первого элемента: ";
    cin >> a;
    cout << endl << "Введите индекс второго элемента: ";
    cin >> b;
    array[a] = array[a] + array[b];
    array[b] = array[a] - array[b];
    array[a] = array[a] - array[b];
  }
```

ПРИЛОЖЕНИЕ Б
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Тестирование программы осуществлялось на компьютере с процессором x64 под управлением ОС Windows 10 Pro (версия 2004, 64-бит) с использованием следующих компиляторов:

- Оптимизирующий компилятор Microsoft (R) C/C++ версии 19.16.27043 для x86

Результаты тестирования программы представлены в табл. 1.

Таблица 1 – Результаты тестирования программы

Ожидаемый результат	Полученный результат
Массив, состоящий из 100 элементов от -99 до 99, сгенерированных псевдослучайным способом.	50 -39 94 32 -59 -70 56 17 36 -54 33 -20 -66 42 74 93 7 7 -32 -74 -36 -56 91 -5 -46 91 82 -50 15 56 34 29 10 43 3 -73 15 94 64 64 -56 -87 38 -65 -83 16 -69 -97 -34 70 -4 65 31 15 -70 39 60 87 -71 62 -49 -33 -40 -62 92 7 -80 95 45 -77 -29 19 -32 -64 -78 -25 -50 56 1 -94 -89 6 -86 -52 33 -6 47 40 -11 5 -26 82 -18 -76 -62 -31 24 -30 -2 -95
-97 -95 -94 -89 -87 -86 -83 -80 -78 -77 -76 -74 -73 -71 -70 -70 -69 -66 -65 -64 -62 -62 -59 -56 -56 -54 -52 -50 -50 -49 -46 -40 -39 -36 -34 -33 -32 -32 -31 -30 -29 -26 -25 -20 -18 -11 -6 -5 -4 -2	-97 -95 -94 -89 -87 -86 -83 -80 -78 -77 -76 -74 -73 -71 -70 -70 -69 -66 -65 -64 -62 -62 -59 -56 -56 -54 -52 -50 -50 -49 -46 -40 -39 -36 -34 -33 -32 -32 -31 -30 -29 -26 -25 -20 -18 -11 -6 -5 -4 -2 1 3 5 6 7 7 7 10 15 15 15 16 17 19 24 29 31 32 33 33 34 36 38 39 40 42 43 45 47 50

1 3 5 6 7 7 7 10 15 15 15 16 17 19 24 29 31 32 33 33 34 36 38 39 40 42 43 45 47 50 56 56 56 60 62 64 64 65 70 74 82 82 87 91 91 92 93 94 94 95	56 56 56 60 62 64 64 65 70 74 82 82 87 91 91 92 93 94 94 95
Минимальный элемент равен: -97 Максимальный элемент равен: 95 На поиск потрачено времени: 8e-07.	Минимальный элемент равен: -97 Максимальный элемент равен: 95 На поиск потрачено времени: 8e-07.
Среднее значение: -1 Индексы элементов, равных этому значению: такого элемента в массиве нет!	Среднее значение: -1 Индексы элементов, равных этому значению: такого элемента в массиве нет!
Введите число: 150 Элементов меньше 150 в массиве: 100.	Введите число: 150 Элементов меньше 150 в массиве: 100.
Введите число: 50 Элементов меньше 50 в массиве: 79.	Введите число: 50 Элементов меньше 50 в массиве: 79.
Введите число: 100 Элементов больше 100 в массиве: 0.	Введите число: 100 Элементов больше 100 в массиве: 0.
Введите число: 50 Элементов больше 50 в массиве: 20.	Введите число: 50 Элементов больше 50 в массиве: 20.
Введите число: 7 В массиве есть такое число. Время, затраченное на поиск: 8e-07.	Введите число: 7 В массиве есть такое число. Время, затраченное на поиск: 8e-07.
Введите число: 11 В массиве такого числа нет. Время, затраченное на поиск: 5e-07.	Введите число: 11 В массиве такого числа нет. Время, затраченное на поиск: 5e-07.
Введите индекс первого элемента: 1 Введите индекс второго элемента: 99 Время, затраченное на поиск: 5.00672. -97 95 -94 -89 -87 -86 -83 -80 -78 -77 -76 -74 -73 -71 -70 -69 -66 -65 -64 -62 -62 -59 -56 -56 -54 -52 -50 -49 -46 -40 -39 -36 -34 -33 -32 -32 -31 -30 -29 -26 -25 -20 -18 -11 -6 -5 -4 -2 1 3 5 6 7 7 7 10 15 15 15 16 17 19 24 29 31 32 33 33 34 36 38 39 40 42 43 45 47 50	Введите индекс первого элемента: 1 Введите индекс второго элемента: 99 Время, затраченное на поиск: 5.00672. -97 95 -94 -89 -87 -86 -83 -80 -78 -77 -76 -74 -73 -71 -70 -70 -69 -66 -65 -64 -62 -62 -59 -56 -56 -54 -52 -50 -50 -49 -46 -40 -39 -36 -34 -33 -32 -32 -31 -30 -29 -26 -25 -20 -18 -11 -6 -5 -4 -2 1 3 5 6 7 7 7 10 15 15 15 16 17 19 24 29 31 32 33 33 34 36 38 39 40 42 43 45 47 50 56 56 56 60 62 64 64 65 70 74 82 82 87 91 91 92 93 94 94 -95

56	56	56	60	62	
64	64	65	70	74	
82	82	87	91	91	
92	93	94	94	-	
		95			