Федеральное государственное автономное образовательное учреждение высшего образования **"Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского" (ННГУ).**

**Институт информационных технологий, математики и механики**

**ОТЧЕТ**

По лабораторной работе

На тему:

**«Генерация случайных чисел. Выполнение операций с**

**ними.»**

**Выполнила:**

Студентка группы 2831Б1ПМ3

Киселева К.В.

**Проверил:**

Заведующий лабораторией суперкомпьютерных технологий и высокопроизводительных вычислений

Лебедев И.Г.

Нижний Новгород

2021 г

Оглавление

[Введение. 3](#_Toc89791139)

[1. Постановка задачи. 4](#_Toc89791140)

[2. Руководство пользователя. 5](#_Toc89791141)

[3. Руководство программиста. 8](#_Toc89791142)

[Заключение. 12](#_Toc89791143)

[Литература. 13](#_Toc89791144)

[Приложение. 14](#_Toc89791145)

[Приложение 1. 14](#_Toc89791146)

[Приложение 2. 14](#_Toc89791147)

[Приложение 3 16](#_Toc89791148)

[Приложение 4 16](#_Toc89791149)

# Введение.

Программирование - это интересный, полезный и увлекательный процесс, благодаря которому мы, с помощью специальных команд, заставляем компьютер, выполнять для нас различные задачи, от выполнения операций с числами и навигации, до управления самолетами, спутниками и прочей техникой.

Случайные числа являются одной из основных составляющих любого языка программирования, на них строятся различные алгоритмы.

В данной лабораторной работе для изучения методов и особенностей работы со случайными числами была поставлена задача, которую нужно было выполнить, используя язык программирования «С».

# Постановка задачи.

Сравнение сортировок.

Реализовать сортировки массивов данных (тип данных “double”) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая.

Сравнить время работы, сделать выводы.

Первая программа создает текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия.

# Руководство пользователя.

Используется две программы. После запуска первой программы выводится сообщение на экран: «Введите количество значений» (см. Рисунок 1)

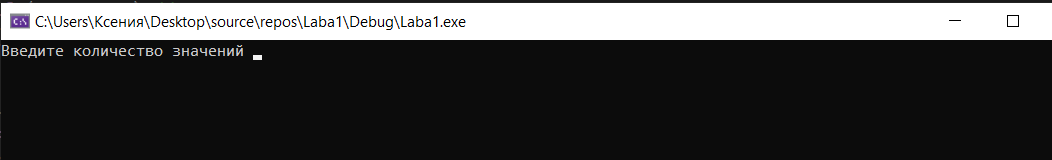


Рисунок 1. Консоль

Введите, выбранное вами, количество значений и нажмите enter. Для примера выбрано число 3 (см. Рисунок 2)

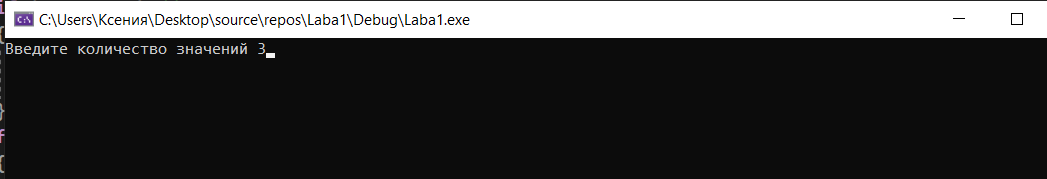


Рисунок 2. Введение числа

Если выбранное вами число окажется меньше равно нуля, программа выведет на экран: «Ошибка! n<=0» (см. Рисунок 3), а так же завершит работу.

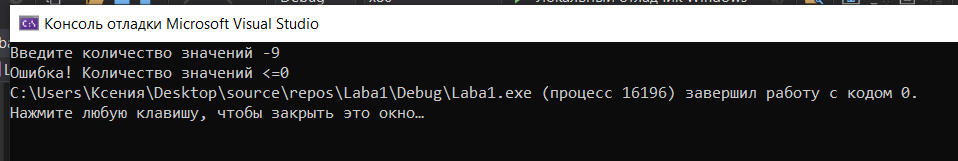


Рисунок 3. Ошибка 1

Если вы введёте слишком большое число, программа выведет на экран: «Ошибка! Не удалось создать массив» и завершит работу (см. Рисунок 4)

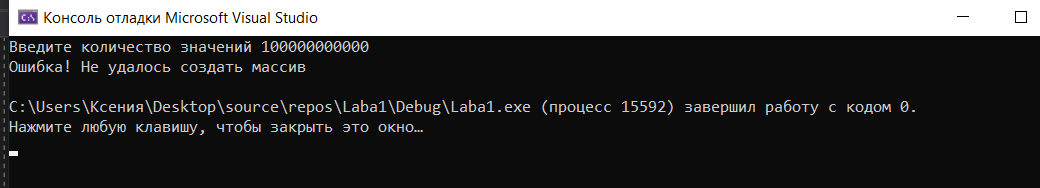


Рисунок 4. Ошибка 2

Если введённое значение корректно, программа продолжит работать и выведет на экран: «Введите значение min» (см. изобр.4)

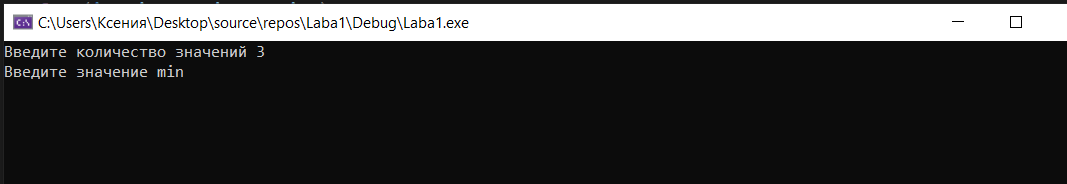


Рисунок 5. Консоль

Введите, выбранное вами, наименьшее значение. Для примера выбрано число 1. (см. Рисунок 6)

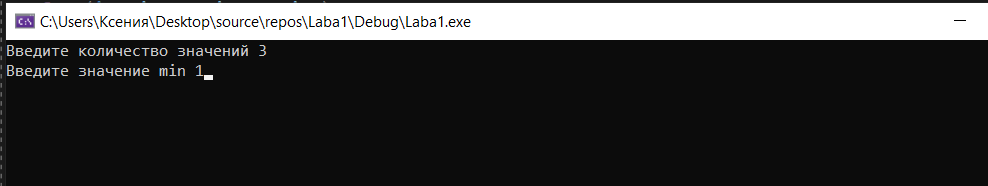


Рисунок 6. Введение значения

Нажмите enter. Программа выведет на экран: «Введите значение max». (см. Рисунок 7)

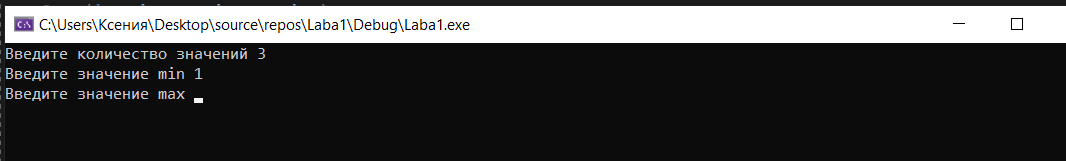


Рисунок 7. Консоль

Введите, выбранное вами, наибольшее значение. Для примера выбрано число 5. (см. Рисунок 8)

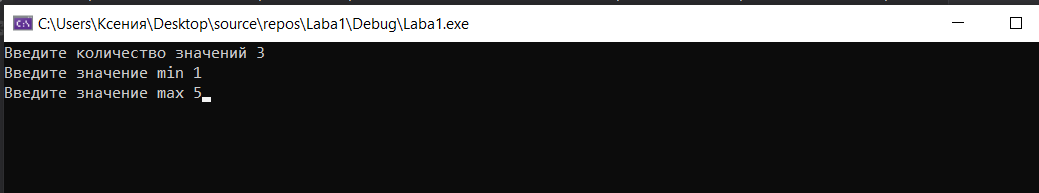


Рисунок 8 Введение значения

Если, введённое вами, наименьшее значение окажется больше наибольшего значения, программа выведет на экран: «Ошибка! min>max», и завершит работу. (см. Рисунок 9)

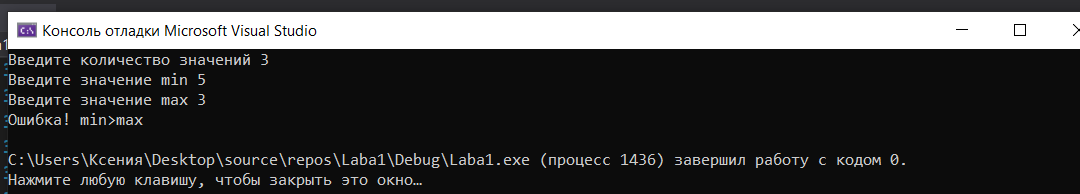


Рисунок 9. Ошибка 3

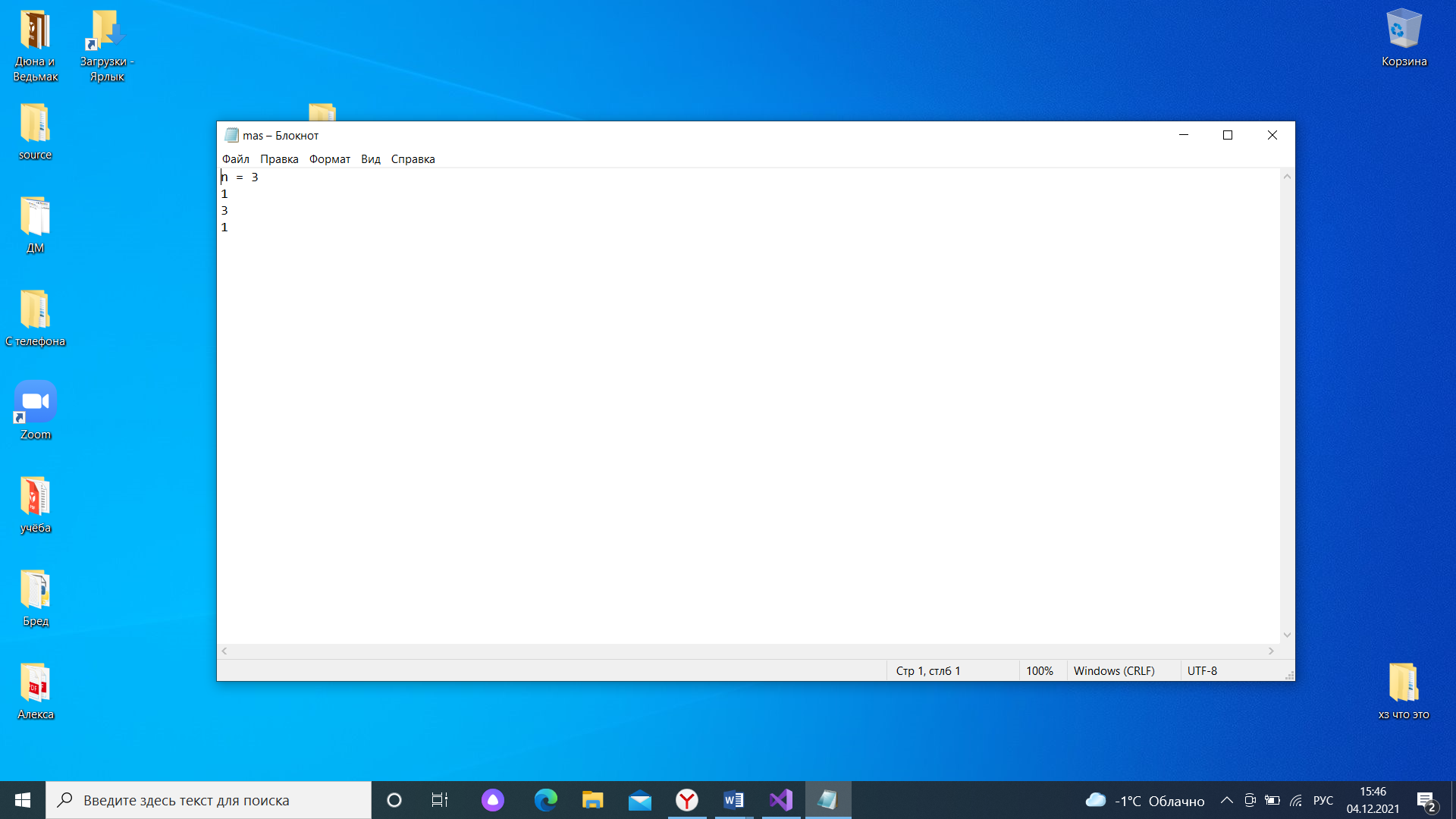
Если введённые данные оказались коректными, в файле появятся числа. (см. Рисунок 10)

Рисунок 10. Числа в файле

Если в процессе работы программы вы введи некорректные данные,запустите программу заново, введите новые значения.

После завершения первой программы следует запустить вторую. После ее запуска на экран будет выведен консольный интерфейс. (см. Рисунок 11)

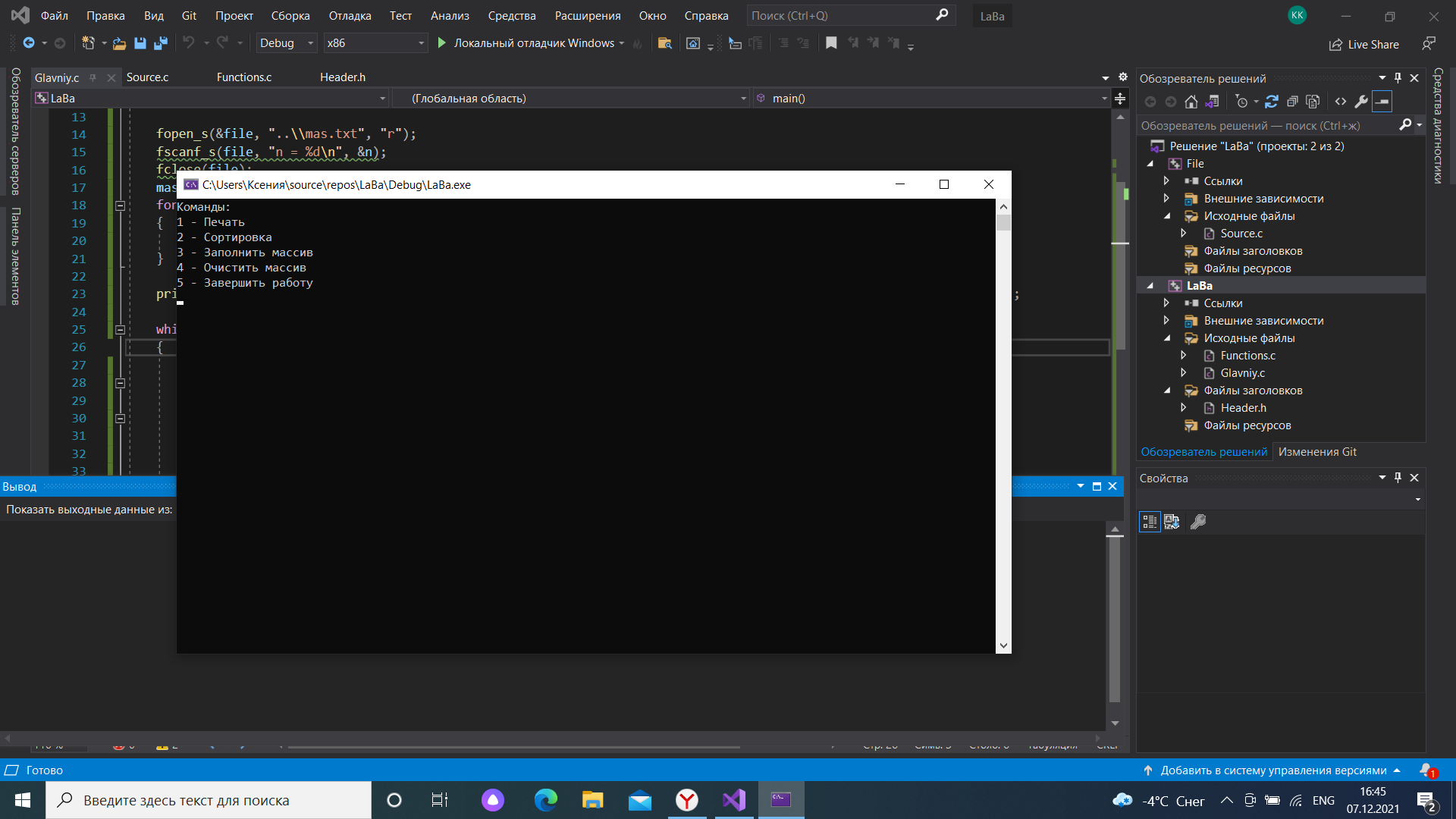


Рисунок 11 Консольный интерфейс

Консольный интерфейс представляет собой набор из четырех команд:

Команды:

1. Печать

2. Сортировка

3. Заполнить массив

4. Очистить массив

5. Завершить работу

Команда 1 печатает массив на экран

Команда 2 вызывает вторую консоль с выбором вида сортировки. (см. Рисунок 12)

Команда 3 заполняет массив числами из файла

Команда 4 очищает массив или заполняет массив нулями

Команда 5 завершает работу программы

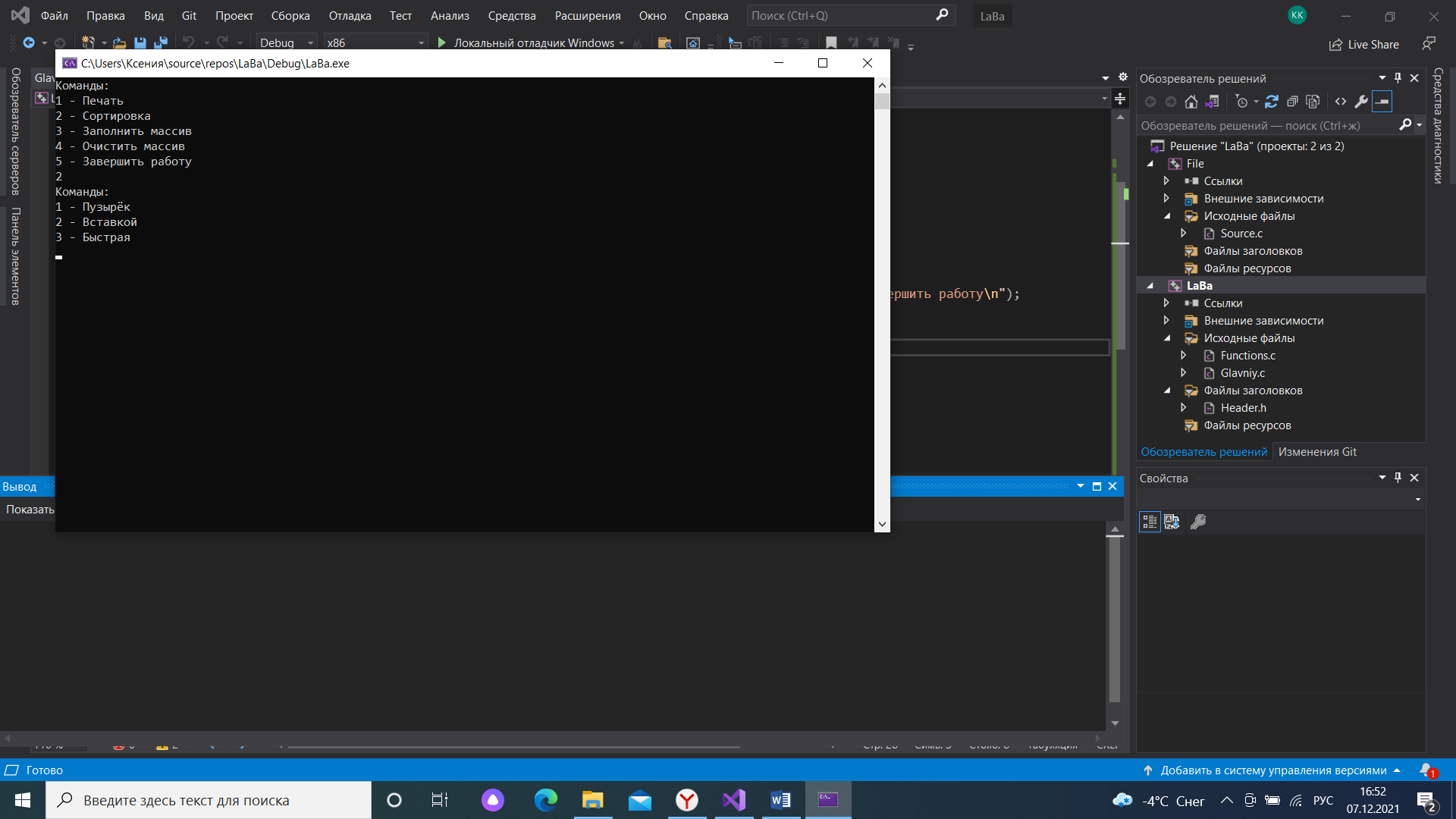


Рисунок 12 Консоль 2

# Руководство программиста.

***Описание структуры программы.***

Первая программа состоит из одного модуля int main() {…}, в котором находиться код создания чисел и записи их в файл. Вторая программа также состоит из одного модуля int main() {…}, в котором находится код сортировки чисел.

***Описание алгоритмов.***

1. **Алгоритм записи случайных чисел в файл.**

|  |
| --- |
| mas = (long int\*)malloc(n \* sizeof(long int));  for (int i = 0; i < n; i++)  {  mas[i] = (((double)rand()) / RAND\_MAX) \* (max - min) + min;  }  fopen\_s(&file, "..\\mas.txt", "w");  fprintf(file, "n = %d\n", n);  for (int i = 0; i < n; i++)  {  fprintf(file, "%ld\n", mas[i]);  }  fclose(file);  free(mas); |

1. **Алгоритм считывания чисел из файла.**

|  |
| --- |
| void ScanMas(FILE\* file, long int\* mas, int n)  {  for (int i = 0; i < n; i++)  fscanf\_s(file, "%ld\n", &(mas[i]));  } |

1. **Алгоритм консольного интерфейса**

См. Рисунок 13

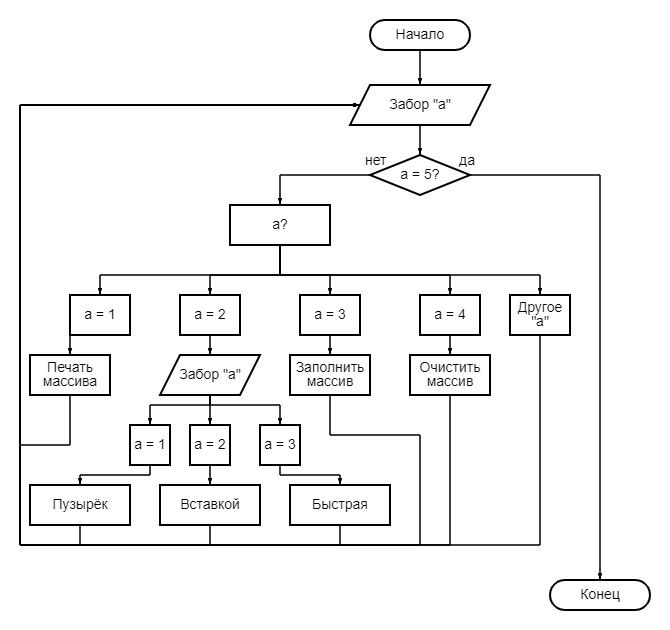
****

Рисунок 13 Блок схема

**4. Сортировки.**

**4.1. Сортировка пузырьком.**

**4.1 Сортировка пузырьком.**

|  |
| --- |
| int tmp = 0;  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n - i - 1; j++)  if (mas[j] > mas[j + 1])  {  tmp = mas[j];  mas[j] = mas[j + 1];  mas[j + 1] = tmp;  }  } |

**4.2. Сортировка вставкой.**

|  |
| --- |
| int tmp = 0;  for (int i = 1; i < n; i++)  {  tmp = mas[i];  for (int j = i - 1; j > 0; j--)  {  if (mas[j] > tmp)  {  mas[j + 1] = mas[j];  mas[j] = tmp;  }  }  } |

**4.3. Быстрая сортировка.**

|  |
| --- |
| void QuickSort(long int\* mas, int n)  {  int i = 0;  int j = n - 1;  int mid = mas[n / 2];  do  {  while (mas[i] < mid) i++;  while (mas[j] > mid) j--;  if (i <= j)  {  int tmp = mas[i];  mas[i] = mas[j];  mas[j] = tmp;  i++;  j--;  }  } while (i <= j);  if (j > 0) quick(mas, j + 1);  if (i < n) quick(&mas[i], n - i);  }  void quick(long int\* mas, int n)  {  int i = 0;  int j = n - 1;  int mid = mas[n / 2];  do  {  while (mas[i] < mid) i++;  while (mas[j] > mid) j--;  if (i <= j)  {  int tmp = mas[i];  mas[i] = mas[j];  mas[j] = tmp;  i++;  j--;  }  } while (i <= j);  if (j > 0) quick(mas, j + 1);  if (i < n) quick(&mas[i], n - i);  } |

1. **Эксперименты.**

|  |  |  |
| --- | --- | --- |
| **Тип сортировки** | **Размер входных данных** | **Время работы, мс** |
| Пузырьком | 10000 |  |
| Вставками | 10000 |  |
| Быстрая | 10000 |  |
| Пузырьком | 100000 |  |
| Вставками | 100000 |  |
| Быстрая | 100000 |  |

# Заключение.

В ходе лабораторной работы была написана программа на языке программирования «С», которая полностью выполняет поставленную задачу, а именно:

«Сравнение сортировок.

Реализовать сортировки массивов данных (тип данных “double”) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая.

Сравнить время работы, сделать выводы.

Первая программа создает текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия».

Интерфейс программы простой, понятный и удобный в использовании, все сообщения выводятся на русском языке и при своих небольших размерах содержат нужную информацию, которую необходимо донести до пользователя.

В ходе сравнения сортировок можно сделать вывод: «быстрая сортировка (quick sort)» выполняет сортировку чисел быстрее остальных типов сортировки и очень быстро работает с любыми типами данных; сортировка «пузырьком» занимает большее время чем «quick sort», но тоже довольно быстро сортирует данные; «сортировка вставками» работает медленнее всех на большом диапазоне чисел, ее следует использовать на коротком отрезке не больше, чем десятки, или уже на частично отсортированном массиве данных.

Литература.  
1. Т.А. Павловская Учебник по программированию на языках высокого  
уровня(С/С++) – Режим доступа: http://cph.phys.spbu.ru/documents/First/books/7.pdf  
2. Бьерн Страуструп. Язык программирования С++ - Режим доступа:  
<http://8361.ru/6sem/books/Straustrup-Yazyk_programmirovaniya_c.pdf>

# Приложение.

## Приложение 1.

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #include<locale.h>  int main()  {  setlocale(LC\_ALL, "Rus");  int n;  long int min, max;  long int\* mas;  FILE\* file = 0;  printf\_s("Введите количество значений ");  scanf\_s("%d", &n);  if (n <= 0) //проверка n  {  printf\_s("Ошибка! Kоличество значений <=0");  return 0;  }  mas = (long int\*)malloc(n \* sizeof(long int));  if (mas == 0) //проверка создался ли массив  {  printf\_s("Ошибка! Hе удалось создать массив \n");  return 0;  }  printf\_s("Введите значение min ");  scanf\_s("%ld", &min);  printf\_s("Введите значение max ");  scanf\_s("%ld", &max);  if (min > max) //проверка max и min  {  printf\_s("Ошибка! min>max \n");  return 0;  }  for (int i = 0; i < n; i++)  {  mas[i] = (((double)rand()) / RAND\_MAX) \* (max - min) + min;  //printf\_s("%lf \n", mas[i]);  }  fopen\_s(&file, "..\\mas.txt", "w");  fprintf(file, "n = %d\n", n);  for (int i = 0; i < n; i++)  {  fprintf(file, "%ld\n", mas[i]);  }  fclose(file);  free(mas);  return 0;  } |

## Приложение 2.

|  |
| --- |
| #include<stdio.h>  #include <locale.h>  #include"Header.h"  #include <time.h>  int main()  {  setlocale(LC\_ALL, "Rus");  int n, a = 0;  long int\* mas = 0;  FILE\* file = 0;  int start = 0, end = 0, tim = 0;  printf\_s("Введите 1 для сортировки пузырьком, 2 - вставкой, 3 - быстрой, 4 - завершить работу\n");  while (a != 4)  {  scanf\_s("%d", &a);  if (a == 1)  {  ScanFile(&mas, &n);  start = clock();  BubbleSort(mas, n);  end = clock();  tim = end - start;  printf\_s("Время работы: %d мс\n", tim);  printf\_s("Введите число\n");  }  else if (a == 2)  {  ScanFile(&mas, &n);  start = clock();  VstavkaSort(mas, n);  end = clock();  tim = end - start;  printf\_s("Время работы: %d мс\n", tim);  printf\_s("Введите число\n");  }  else if (a == 3)  {  ScanFile(&mas, &n);  start = clock();  QuickSort(mas, n);  end = clock();  tim = end - start;  printf\_s("Время работы: %d мс\n", tim);  printf\_s("Введите число\n");  }  else if (a == 4)  {  break;  }  else  {  printf\_s("Вы ввели неправильное значение\n");  printf\_s("Введите 1 для сортировки пузырьком, 2 - вставкой, 3 - быстрой, 4 - завершить работу\n");  }  }  free(mas);  return 0;  } |

## Приложение 3

|  |
| --- |
| #ifndef \_\_HEADER\_H\_\_  #define \_\_HEADER\_H\_\_  void PrintMas(long\* mas, int n);  void CreateMas(long\*\* mas, int n);  void ScanN(FILE\* file, int\* n);  void ScanMas(FILE\* file, long int\* mas, int n);  void ScanFile(long int\*\* mas, int\* n);  void BubbleSort(long int\* mas, int n);  void VstavkaSort(long int\* mas, int n);  void QuickSort(long int\* mas, int n);  void quick(long int\* mas, int n);  #endif |

## Приложение 4

|  |
| --- |
| #include<stdio.h>  #include"Header.h"  void PrintMas(long int\* mas, int n)  {  for (int i = 0; i < n; i++)  {  printf\_s("mas[%d] = %ld\n", i, mas[i]);  }  }  void CreateMas(long int\*\* mas, int n)  {  \*mas = (long int\*)malloc(n \* sizeof(long int));  for (int i = 0; i < n; i++)  {  (\*mas)[i] = 0;  }  }  void ScanN(FILE\* file, int\* n)  {  fscanf\_s(file, "n = %d\n", n);  }  void ScanMas(FILE\* file, long int\* mas, int n)  {  for (int i = 0; i < n; i++)  fscanf\_s(file, "%ld\n", &(mas[i]));  }  void ScanFile(long int\*\* mas, int\* n)  {  FILE\* file = 0;  fopen\_s(&file, "..\\mas.txt", "r");  ScanN(file, n);  CreateMas(mas, \*n);  ScanMas(file, \*mas, \*n);  fclose(file);  }  void BubbleSort(long int\* mas, int n)  {  int tmp = 0;  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n - i - 1; j++)  if (mas[j] > mas[j + 1])  {  tmp = mas[j];  mas[j] = mas[j + 1];  mas[j + 1] = tmp;  }  }  }  void VstavkaSort(long int\* mas, int n)  {  int tmp = 0;  for (int i = 1; i < n; i++)  {  tmp = mas[i];  for (int j = i - 1; j > 0; j--)  {  if (mas[j] > tmp)  {  mas[j + 1] = mas[j];  mas[j] = tmp;  }  }  }  }  void QuickSort(long int\* mas, int n)  {  int i = 0;  int j = n - 1;  int mid = mas[n / 2];  do  {  while (mas[i] < mid) i++;  while (mas[j] > mid) j--;  if (i <= j)  {  int tmp = mas[i];  mas[i] = mas[j];  mas[j] = tmp;  i++;  j--;  }  } while (i <= j);  if (j > 0) quick(mas, j + 1);  if (i < n) quick(&mas[i], n - i);  }  void quick(long int\* mas, int n)  {  int i = 0;  int j = n - 1;  int mid = mas[n / 2];  do  {  while (mas[i] < mid) i++;  while (mas[j] > mid) j--;  if (i <= j)  {  int tmp = mas[i];  mas[i] = mas[j];  mas[j] = tmp;  i++;  j--;  }  } while (i <= j);  if (j > 0) quick(mas, j + 1);  if (i < n) quick(&mas[i], n - i);  } |