**1 Введение**

Информация является основным элементом передачи сведений в нашей жизни. С давних времён люди поняли, что данные удобнее хранить и передовать в структурированном виде. Примером такого хранения информации является библиотека. Книги сортируют по жанру, месту создания и многим другим критериям.

Раньше люди занимались сортировкой большого объёма данных вручную, и зачастую это был очень долгий и трудоёмкий процесс. За тысячи лет способы передачи, хранения и обработки информации эволюционировали. В наше время обработка данных происходит автоматически, при помощи специализированных программ.

Для этого они использую различны алгоритмы сортировки данных. Сортировка — это последовательное расположение или разбиение на группы чего-либо в зависимости от выбранного критерия.

За всю историю человечества было придумано огромное множество различных алгоритмов сортировок. От самых простых, например, «сортировка пузырьком» до сложных «timsort».

Для совместной разработки мы использовали  крупнейший [веб-сервис](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81) для [хостинга](https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3) [IT-проектов](https://ru.wikipedia.org/wiki/IT) и их совместной разработки - github.

По мимо удобства передачи проекта от одного участника команды к другому, git hub содержит полезные функции сохранения промежуточных результатов работы и, в случае ошибки, возвращения к ним.

Для удобства совместной работы мы выбрали каскадный метод разработки программы. Каждый участник выбрал себе по задаче котор

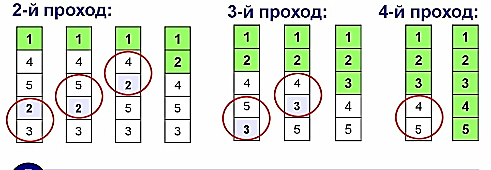
Для начала была создана работа с файлами и заполнение массива случайными числами от -1000 до 1001.

Потом была разработана программа, которая выполняет сортировку массива.

И в конце для готового алгоритма было разработано «Меню»

**Достоинства алгоритма**

* Простота реализации. Создаётся цикл, который поочерёдно сравнивает 2 соседних элемента и, если левый элемент больше правого меняет их местами.

****

* Возможность оптимизации процесса. Можно уменьшать каждое последующее количество проходов по массиву на 1 элемент
* Если массив уже отсортирован программа быстро завершит работу
* Допускает эффективный проход как от начала к концу, так и от конца к началу.

**2 Выбор решения**

Необходимо разработать алгоритм сортировки пузырьком. Для этого должны быть изучены следующие вопросы:

— способы работы с консолью

— реализация функций для работы с файлами

Для вывода удобного для пользователя интерфейса было решено использовать оператор switch(). Он удобен для создания меню программы, так как позволяет реализовать выбор тех или иных действий с помощью ввода символов.

Язык программирования С имеет множество инструментов для работы с файлами. Он использует стандартные библиотеки ввода-вывода, которые позволяют открывать, закрывать и работать с файлами на диске. Функциональность ввода-вывода языка Си по текущим стандартам реализуется на низком уровне. Язык Си превращает все операции с файлами в операции с потоками байтов, которые могут быть как «потоками ввода», так и «потоками вывода». Однако в отличие от некоторых ранних языков программирования, язык Си не имеет прямой поддержки произвольного доступа к файлам данных; Из-за этого приходится создавать поток, ищущий ту или иную часть файла, прежде чем работать с ней.

**4. Совместная разработка программы**

Для совместной разработки мы использовали  крупнейший [веб-сервис](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81) для [хостинга](https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3) [IT-проектов](https://ru.wikipedia.org/wiki/IT) и их совместной разработки - github.

По мимо удобства передачи проекта от одного участника команды к другому, git hub содержит полезные функции сохранения промежуточных результатов работы и, в случае ошибки, возвращения к ним.

Для удобства совместной работы мы выбрали каскадный метод разработки программы. Каждый участник выбрал себе по задаче котор

Для начала была создана работа с файлами и заполнение массива случайными числами от -1000 до 1001.

Потом была разработана программа, которая выполняет сортировку массива.

И в конце для готового алгоритма было разработано «Меню»

**5. Заключение.**

# ***Список используемых источников***

1. Клеменс Б. Язык С в XXI веке / Клеменс Б. – ДМК-Пресс, 2018 г. – 376 с.
2. Керниган Б. Ритчи Д. Язык программирования Си / Керниган, Б. Ритчи Д. – Вильямс, 2019 г. – 288 с.

**Приложение А**

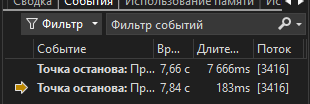


Рисунок А.1

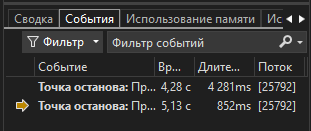


Рисунок А.2

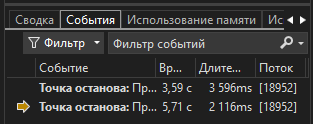


Рисунок А.3

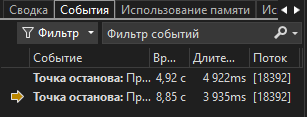


Рисунок А.4

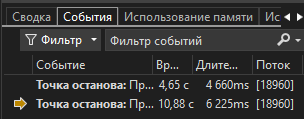


Рисунок А.5

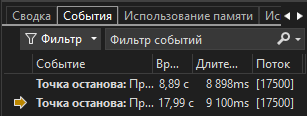


Рисунок А.6

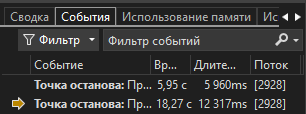


Рисунок А.7

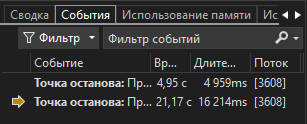


Рисунок А.8

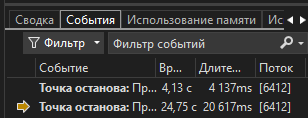


Рисунок А.9

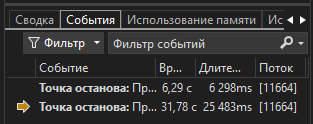


Рисунок А.10

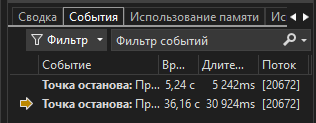


Рисунок А.11

# **Приложение Б**

**Листинги программы**

**Файл main.cpp**

#include "Header.h"

int main()

{

srand(time(NULL));

FILE\* input, \* output;

int size;

bool work = true;

int\* mas;

int x = 0;

input = fopen("input", "w+");

output = fopen("output", "w+");

setlocale(LC\_ALL, "Rus");

printf("Добро пожаловать! Данная программа выполняет сортировку массива случайно сгенерированых чисел.\n\nСначала введите размер массива:\n");

scanf\_s("%d", &size);

while (size <= 1)

{

printf("Попробуйте еще раз.");

scanf\_s("%d", &size);

}

mas = new int[size];

while (work == true)

{

system("cls");

printf("Выберете следующее действие:\n1. Изменить размер массива.\n2. Заполнить массив случайными числами.\n3. Отсортировать массив.\n");

getchar();

char comand[] = { getchar() };

switch (comand[0])

{

case '1':

printf(" Введите количество чисел: ");

scanf\_s("%d", &size);

while (size <= 1)

{

printf("Попробуйте еще раз.");

scanf\_s("%d", &size);

}

mas = new int[size];

break;

case '2':

input\_masize(mas, input, size);

break;

case '3':

sortid(size, mas);

output\_mas(size, mas, output);

work = false;

break;

}

}

return 0;

}

**Файл sortid.cpp**

#include"Header.h"

void sortid(int size, int\* mas) {

int save;

for (int exit = 1;exit < size; exit++) {

for (int r = 0; r < size - exit; r++) {

if (mas[r] > mas[r + 1]) {

save = mas[r];

mas[r] = mas[r + 1];

mas[r + 1] = save;

}

}

}

}

**Файл input.cpp**

#include"Header.h"

void input\_masize(int\* mas, FILE\* input, int size) {

for (int i = 0; i < size; i++) {

mas[i] = (rand() % 2001) - 1000;

fprintf(input, "%d ", mas[i]);

}

}

**Файл output.cpp**

#include"Header.h"

void output\_mas(int size, int \*mas, FILE\* output) {

for (int i = 0; i < size; i++) {

fprintf(output, "%d ", mas[i]);

}

}