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

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Программирование»
Тема: «Обработка последовательности значений с использованием массивов»

Студентка гр. 3352 _____ Калюжная М.И.

Преподаватель _____ Калмычков В.А.

Санкт-Петербург
2023

Исходная формулировка

Даны два одномерных массива X и Y из разного числа элементов. Найти количество пар элементов: Xt и Yj, имеющих одинаковые знаки.

Формальная постановка задачи

Дано	Файлы ввода Array1.txt и Array2.txt, содержащий элементы массива *B*и *A*-значение количества используемых чисел типа int.
Найти	Найти количество пар элементов Xt и Yj, имеющих одинаковые знаки.
Способ решения	Поочередно перемножать элементы через цикл for и проверять, больше ли их провезведение чем 0.

Организация диалога с пользователем

После запуска программа выводит в консоль результат. Также выводит в файл result.txt длины массивов и элементы исходных массивов. После выполнения операций вычисления выводится итоговое значение кол-ва пар элементов. Затем пользователю предлагается пройти в файл вывода result.txt.

Контрольный пример

Для 1 версии:	Для 2 версии:
После обработки tests9.txt и tests11.txt результат с статич. массивами составил: 3 Массив 1: (Длина 3) 1 1 1 Массив 2: (Длина 2) -1 3	После обработки dynamic_array1.txt и dynamic_array2.txt результат с динамич. массивами составил: 11 Массив 1: (Длина 4) 3 12 2 -3 Массив 2: (Длина 5) 4 1 2 -1 -88

Формат хранения данных

Для первой версии программы

Тип	Имя	Назначение
string	temp	Вспомогательные переменные
float	B[B_len], A[A_len]	Статические массивы
int	A_len, B_len	Переменные, хранящие длины массивов
int	k, i, j	Переменные, отвечающие за индекс элемента
const int	MAX_ARRAY_SIZE	Максимальная длина массивов
int	result	Кол-во пар элементов, имеющих одинаковые знаки

Для второй версии программы

Тип	Имя	Назначение
float	tmp, tmp1, tmp2	Вспомогательные переменные
	*B *A	Указатели на динамические массивы
int	A_len, B_len	Переменные, хранящие длины массивов
int	k, i, j	Переменные, отвечающие за индекс элемента
int	result	Кол-во пар элементов, имеющих одинаковые знаки
	*pB, *p_b, *p_a	Вспомогательные указатели

Ограничение, условленное исполнением на компьютере

int – целочисленный типа данных. Диапазон: от -2147483648 до +2147483647
float – вещественное число увеличенной точности. Диапазон: от 3.4e-38 до 3.4e+38

Описание алгоритма

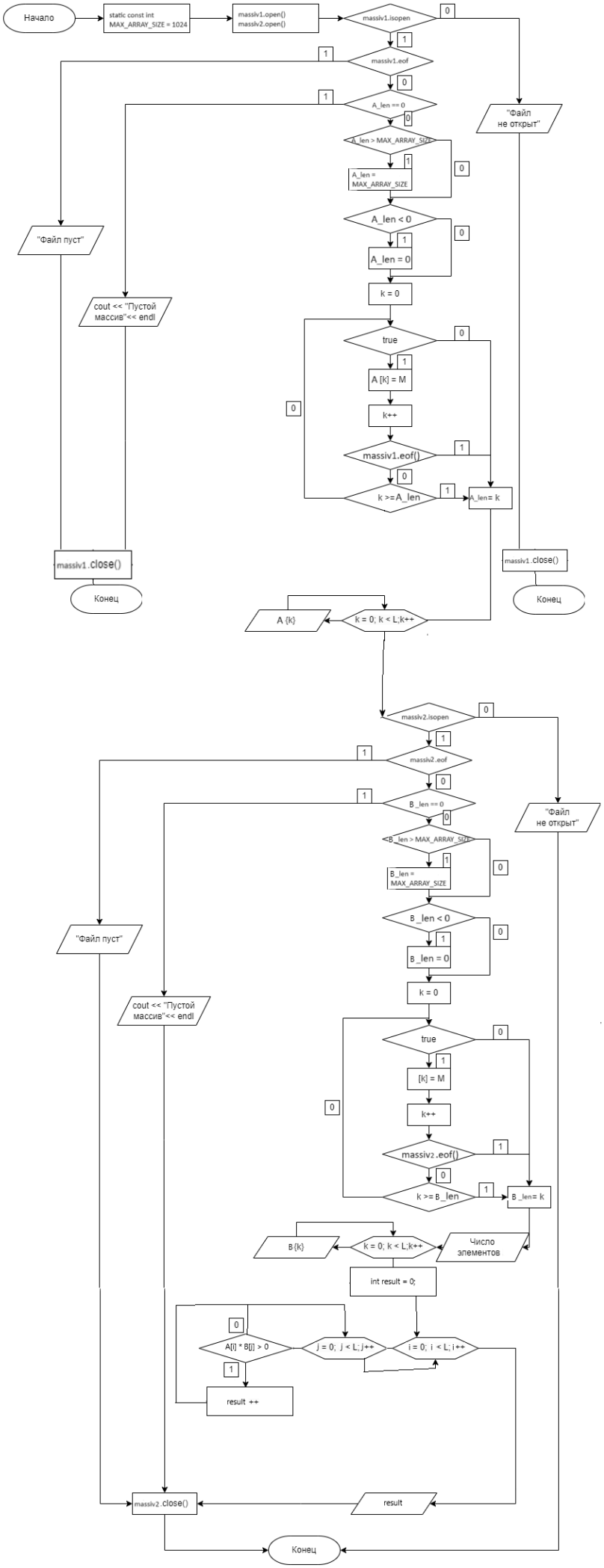
Для первой версии программы:	**Для второй версии программы:**
1. Обработка входящих файлов Array1.txt, Array2.txt, содержащих массивы. 2. Выполнение необходимых действий над массивом. 3. Вывод массивов, их длины и итогового результата в файл. 4. Завершение работы программы.	1. Вывод приветствия и условия задачи. 2. Обработка входящих файлов Array1.txt, Array2.txt содержащего массив. 3. Выполнение необходимых действий над массивом. 4. Вывод массивов, их длины и итогового результата в файл. 5. Очистка памяти. 6. Завершение работы программы.

Средства обеспечения ввода/вывода

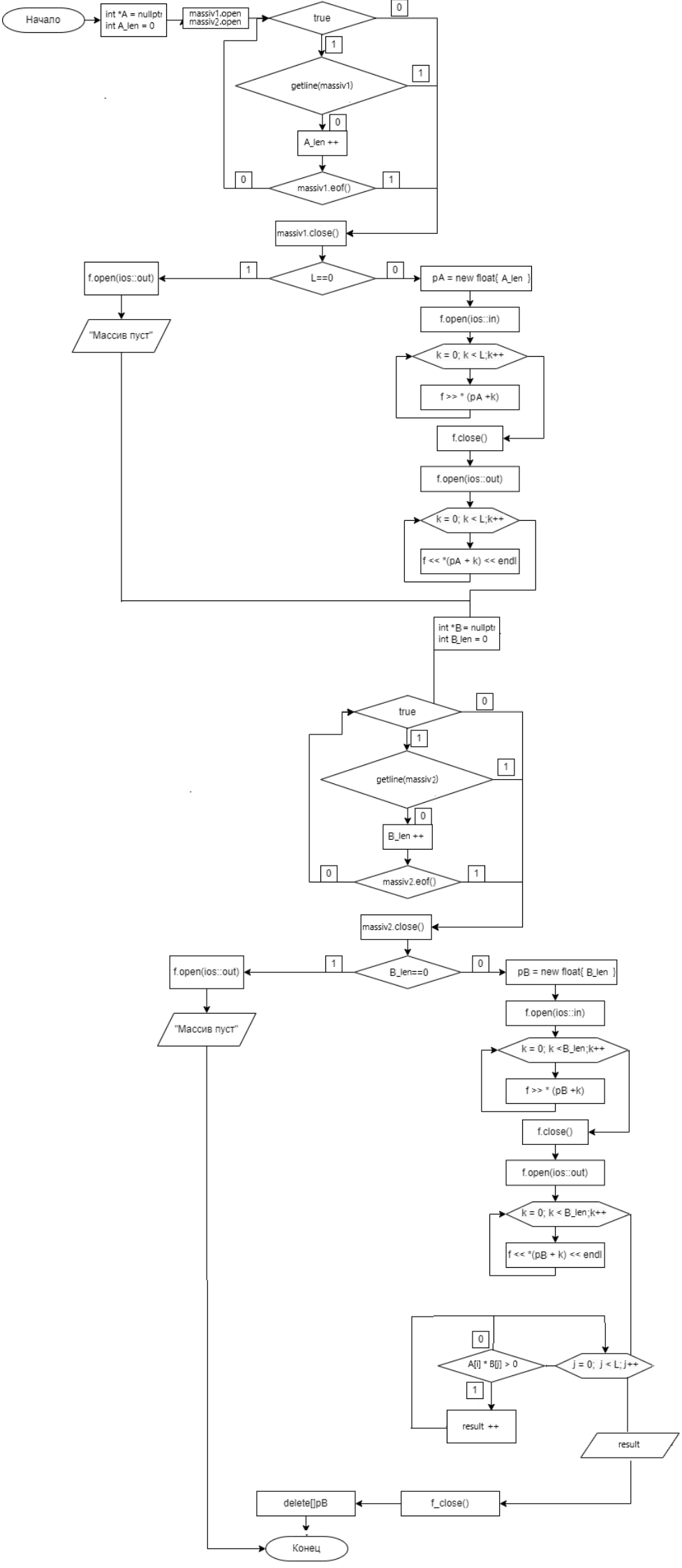
Библиотека	Команды
iostream	cout
fstream	open, close, out

Алгоритм решения

1 версия


Начало
static const int MAX_ARRAY_SIZE = 1024
massiv1.open()
massiv2.open()
massiv1.isopen
"Файл не открыт"
massiv1.eof
A_len == 0
"Файл пуст"
cout << "Пустой массив"<< endl
A_len > MAX_ARRAY_SIZE
A_len = MAX_ARRAY_SIZE
A_len < 0
A_len = 0
k = 0
true
A [k] = M
k++
massiv1.eof()
k >=A_len
A_len= k
massiv1.close()
Конец
A {k}
k = 0; k < L;k++
massiv2.isopen
massiv2.eof
B_len == 0
B_len > MAX_ARRAY_SIZE
B_len = MAX_ARRAY_SIZE
B_len < 0
B_len = 0
[k] = M
massiv2.eof()
k >= B_len
B_len= k
B {k}
Число элементов
int result = 0;
A[i] * B[j] > 0
j = 0; j < L; j++
i = 0; i < L; i++
result ++
massiv2.close()
result

2 версия


Начало
int *A = nullptr
int A_len = 0
massiv1.open
massiv2.open
true
getline(massiv1)
A_len ++
massiv1.eof()
massiv1.close()
L==0
f.open(ios::out)
"Массив пуст"
pA = new float{ A_len }
f.open(ios::in)
k = 0; k < L;k++
f >> * (pA +k)
f.close()
f.open(ios::out)
k = 0; k < L;k++
f << *(pA + k) << endl
int *B = nullptr
int B_len = 0
true
getline(massiv2)
B_len ++
massiv2.eof()
massiv2.close()
B_len==0
f.open(ios::out)
"Массив пуст"
pB = new float{ B_len }
f.open(ios::in)
k = 0; k <B_len;k++
f >> * (pB +k)
f.close()
f.open(ios::out)
k = 0; k < B_len;k++
f << *(pB + k) << endl
A[i] * B[j] > 0
j = 0; j < L; j++
result ++
result
f_close()
delete[]pB
Конец

Программа

```
#include <iostream>
#include <fstream>
using namespace std;
void StaticArrayFromFileStream(std::string filename1, std::string filename2)
{
   ofstream result_stream;
   result_stream.open("result_static.txt", std::ios_base::app);
   result_stream << "После обработки " << filename1 << " и " << filename2;
   static const int MAX_ARRAY_SIZE = 1024;
   float A[MAX_ARRAY_SIZE];
    int A_len = 0;
   ifstream massiv1;
   massiv1.open(filename1, std::ios_base::in);
   if (massiv1.eof())
   {  std::cout << " Файл 1 пуст, упс" << endl;
      result_stream << " Файл 1 пуст, упс" << endl;
      return;}
    std::string tmp;
   getline(massiv1, tmp);
   if (tmp.empty()){
      std::cout << "Файл пуст, упс" << endl;
      result_stream << " Файл пуст, упс" << endl;
      return;    }
   A_len = atoi(tmp.c_str());
   if (A_len < 0)
   {   std::cout << "Отрицательных количеств не бывает! Будем считать что это ноль";
      A_len = 0;
      std::cout << "Файл 1 пуст, упс" << endl;
      result_stream << " Файл 1 пуст, упс" << endl;
      return;}
   if (A_len == 0)
   {  std::cout << "Файл 1 пуст, упс" << endl;
      result_stream << " Файл 1 пуст, упс" << endl;
      return;   }
   if (A_len > MAX_ARRAY_SIZE)
   {       std::cout << "Похоже в файле больше чисел чем максимальное число элементов в нашем массиве, считаем сколько можем" << endl;
      A_len = MAX_ARRAY_SIZE;    }
   for (int k = 0; k < A_len; k++)
   {   getline(massiv1, tmp);
       if (tmp.empty())
      {   std::cout << "строка пуста, упс" << endl;
         result_stream << " строка пуста, упс" << endl;
         return;
      }
      A[k] = atof(tmp.c_str());
     }
   massiv1.close();
   float B[MAX_ARRAY_SIZE];
   int B_len = 0;
   ifstream massiv2;
   massiv2.open(filename2, std::ios_base::in);
     if (massiv2.eof())
   {  std::cout << "Файл 2 пуст, упс" << endl;
      result_stream << " Файл 2 пуст, упс" << endl;
      return;  }
   getline(massiv2, tmp);
   if (tmp.empty())
   {  std::cout << "Файл пуст, упс" << endl;
      result_stream << " Файл пуст, упс" << endl;
      return;   }
   B_len = atoi(tmp.c_str());
   if (B_len < 0)
   {  std::cout << "Отрицательных количеств не бывает! Будем считать что это ноль" << endl;
      B_len = 0;
      std::cout << "Файл 2 пуст, упс" << endl;
      result_stream << "  Файл 2 пуст, упс" << endl;
      return;   }
   if (B_len == 0)
   {   std::cout << "Файл 2 пуст, упс" << endl;
      result_stream << " Файл 2 пуст, упс" << endl;
      return; }
   if (B_len > MAX_ARRAY_SIZE)
   {
      std::cout << "Похоже в файле больше чисел чем максимальное число элементов в нашем массиве, считаем сколько можем" << endl;
      B_len = MAX_ARRAY_SIZE;
   }
   if (A_len == B_len)
   {
      std::cout << "Массивы не должны быть одного размера!" << endl;
      result_stream << "  Массивы не должны быть одного размера!" << endl;
      return;    }
   for (int k = 0; k < B_len; k++)
   {   getline(massiv2, tmp);
      if (tmp.empty())
      {  std::cout << "строка пуста, упс" << endl;
         result_stream << " строка пуста, упс" << endl;
         return; }
```

```
        B[k] = atof(tmp.c_str());
    }
    massiv2.close();
    int result = 0;
    for (int i = 0; i < A_len; i++)
    {
        for (int j = 0; j < B_len; j++)
        {
            if (A[i] * B[j] > 0)
            {
                result++;
            }
        }
    }

    result_stream << " результат с статич. массивами составил: " << result << std::endl;
    result_stream << " Массив 1: "
                  << "(Длина " << A_len << ")" << endl;
    for (int k = 0; k < A_len; k++)
    {       result_stream << A[k] << endl;    }
    result_stream << endl;
    result_stream << " Массив 2: "
                  << "(Длина " << B_len << ")" << endl;
    for (int k = 0; k < B_len; k++)
    {       result_stream << B[k] << endl;    }
    result_stream << endl;
    result_stream << endl;
    std::cout << result << " (result static + file)" << endl;
}

void DynamicArrayFromFileStream(std::string filename1, std::string filename2)
{
    ofstream result_stream;
    result_stream.open("result_dynamic.txt", std::ios_base::app);
    int *A = nullptr;
    int A_len = 0;
    ifstream massiv1;
    massiv1.open(filename1.c_str(), std::ios_base::in);
        if (massiv1.eof())
    {
        std::cout << "Файл 1 пуст, упс";
        result_stream << "Файл 1 пуст, упс" << std::endl;
        return;
    }
    std::string tmp;

    while (getline(massiv1, tmp))
        ++A_len;
    A = new int[A_len];
    if (A == nullptr)
    {
        std::cout << "Память не выделилась, упс";
        result_stream << "Память не выделилась, упс" << std::endl;
        return;
    }
    massiv1.close();
    massiv1.open(filename1.c_str(), std::ios_base::in);
    int counter = 0;
    int tmp1 = 0;
    int *p_a = A;
    while (!massiv1.eof())
    {
        massiv1 >> tmp1;
        *p_a = tmp1;
        ++p_a;
    }
    massiv1.close();
    int *B = nullptr;
    int B_len = 0;
    ifstream massiv2;
    massiv2.open(filename2.c_str(), std::ios_base::in);
    if (massiv2.eof())
    {
        std::cout << "Файл 2 пуст, упс";
        result_stream << "Файл 2 пуст, упс" << std::endl;
        return;
    }
    while (getline(massiv2, tmp))
        ++B_len;
    B = new int[B_len];
    if (B == nullptr)
    {
        std::cout << "Память не выделилась, упс";
        result_stream << "Память не выделилась, упс" << std::endl;
        return;
    }
    massiv2.close();
    massiv2.open(filename2.c_str(), std::ios_base::in);
    counter = 0;
    int tmp2 = 0;
```

```
    int *p_b = B;
    while (!massiv2.eof())
    {
        massiv2 >> tmp2;
        *p_b = tmp2;
        ++p_b;
    }
    if (A_len == B_len)
    {
        std::cout << "Массивы не должны быть одного размера!" << endl;
        result_stream << "После обработки " << filename1 << " и " << filename2 << "  Массивы не должны быть одного размера!" << endl;
        return;
    }
    int result = 0;
    p_a = A;
    p_b = B;
    for (int i = 0; i < A_len; i++)
    {
        for (int j = 0; j < B_len; j++)
        {
            if ((*p_a) * (*p_b) > 0) // то же самое было бы if ((*(A + i)) * (*(B + j))) > 0)
            {
                result++;
            }
            ++p_b;
        }
        ++p_a;
    }
     std::cout << result << " (result dynamic + file)" << endl;
    result_stream << "После обработки " << filename1 << " и " << filename2 << " результат с динамич. массивами составил: " << result << std::endl;
    result_stream << " Массив 1: "
                  << "(Длина " << A_len << ")" << endl;
    for (int k = 0; k < A_len; k++)
    {
        result_stream << A[k] << endl;
    }
    result_stream << endl;
    result_stream << " Массив 2: "
                  << "(Длина " << B_len << ")" << endl;

    int *pB = B;
    for (int k = 0; k < B_len; k++)
    {
        result_stream << *(pB + k) << endl;
    }
    result_stream << endl;
    result_stream << endl;
    delete[] A; // delete[] для удаления памяти выделенной под массив (а не под один инт)
    delete[] B;
}

int main(int argc, char *argv[])
{
    {
        ofstream result_stream;
        result_stream.open("result_static.txt", std::ios_base::trunc);
    }
    {
        ofstream result_stream;
        result_stream.open("result_dynamic.txt", std::ios_base::trunc);
    }
    StaticArrayFromFileStream("static_array1.txt", "static_array2.txt");
    StaticArrayFromFileStream("static_array2.txt", "static_array1.txt");
    StaticArrayFromFileStream("static_array1.txt", "tests1.txt");
    StaticArrayFromFileStream("tests9.txt", "tests11.txt");
    DynamicArrayFromFileStream("dynamic_array1.txt", "dynamic_array2.txt");
    DynamicArrayFromFileStream("dynamic_array1.txt", "testd1.txt");
    DynamicArrayFromFileStream("testd1.txt", "dynamic_array2.txt");
    DynamicArrayFromFileStream("testd11.txt", "dynamic_array2.txt");
    return 0;
}
```

Результаты работы программы

1 Версия

После обработки tests9.txt и tests11.txt результат с статич. массивами составил: 3

Массив 1: (Длина 3)

1

1

1

Массив 2: (Длина 2)

-1

3

После обработки static_array1.txt и tests1.txt Файл пуст, упс

После обработки static_array2.txt и static_array2.txt Массивы не должны быть одного размера!

2 Версия

После обработки testd7.txt и testd7.txt Массивы не должны быть одного размера!

После обработки testd1.txt и dynamic_array2.txt результат с динамич. массивами составил: 0

Массив 1: (Длина 0)

Массив 2: (Длина 5)

4

1

2

-1

-88

После обработки testd11.txt и dynamic_array2.txt результат с динамич. массивами составил: 8

Массив 1: (Длина 3)

2

-2

4

Массив 2: (Длина 5)

4

1

2

-1

-88

Вывод о проделанной работе

В ходе выполнения задания была освоена работа с одномерными массивами, и улучшен навык работы с файлами, для применения их в программах.