**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет №3

«Системы управления, информатика и электроэнергетика»

Кафедра 304

**Отчет по лабораторной работе №1**

по учебной дисциплине «Информатика»

на тему «Символьные данные»

Вариант № 15

Руководитель работы

\_\_\_\_\_\_\_\_\_\_\_\_ Давыдкина Е. А.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2022 г.

\_\_\_\_\_\_\_\_\_\_\_\_ Секретарев В. Е.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2022 г.

Исполнитель

обучающийся группы М30-110Б-21

\_\_\_\_\_\_\_\_\_\_\_\_ Осминнов Н. М.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

\_\_\_\_\_\_\_\_\_\_\_\_Пысларь А. И

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Москва 2022

**Содержание.**

1. Задание……………………………………………….......................................................................3
2. Блок-схема………………………………………………………………………………………………………………4
3. Псевдокод……………………………………………………………………………………………………………….12
4. Код программы……………………………………………………………………………………………………….16
5. Тесты программы…………………………………………………………………………………………………...20
6. Вывод……………………………………………………………………………………………………………………….22

**Задание.**

Кафедра 304 Курс: Информатика

***Задание 3: Символьные данные***

**ВАРИАНТ № 15**

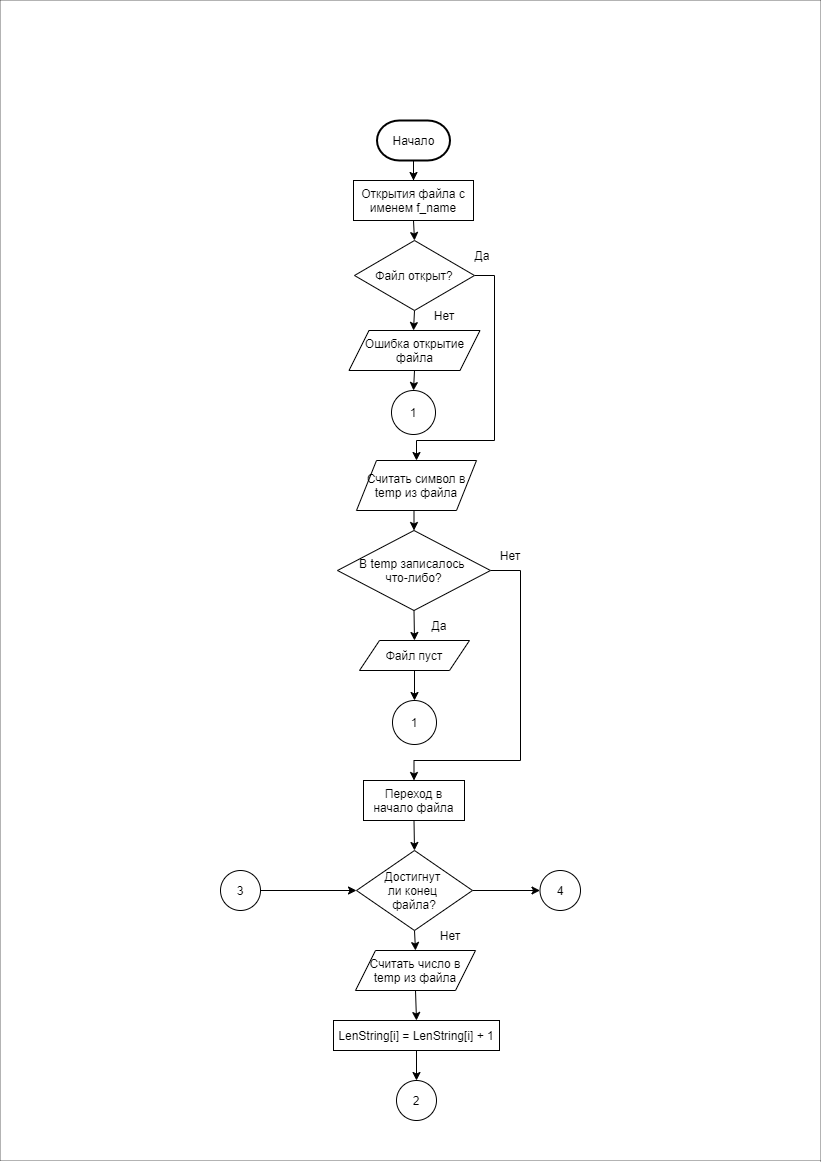
В файле исходных данных задаются две строки символов. Написать программу, включающую две процедуры, выполняющие следующие действия:

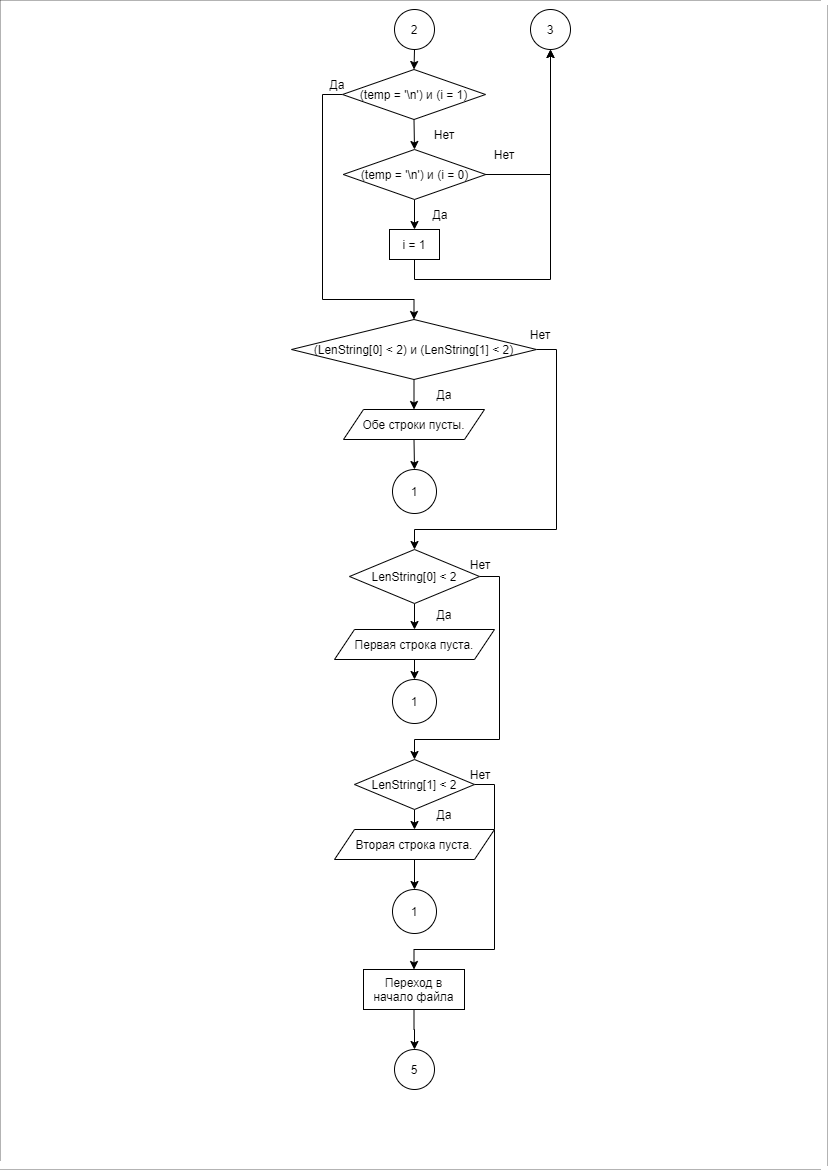
1. Определение и печать слов, встречающихся в обеих строках.

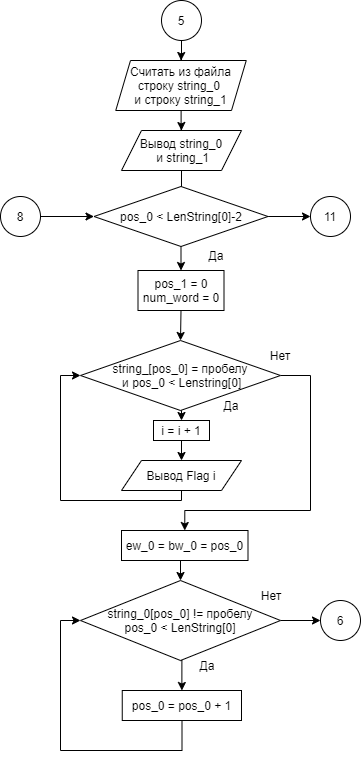
2. Объединение символов двух строк в выходную строку, в которой символы исходных строк чередуются через четыре.

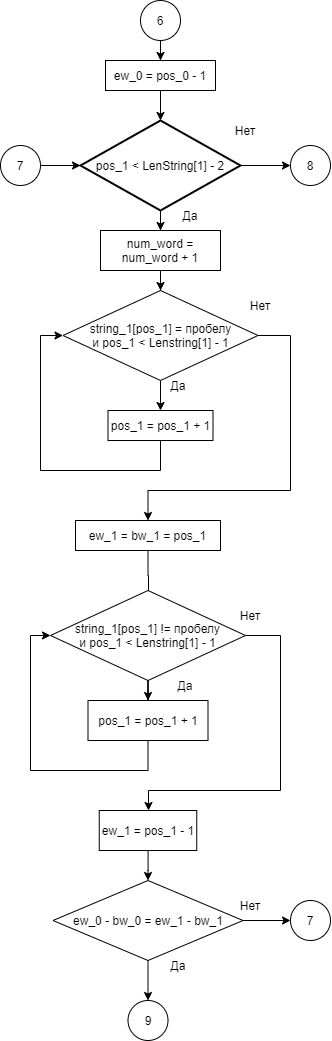
Чтение данных их файла производить с использованием функций ввода/вывода языка С++.

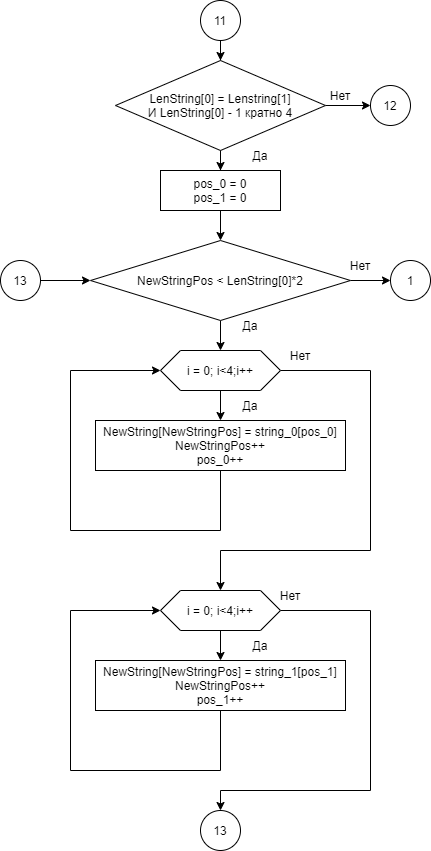
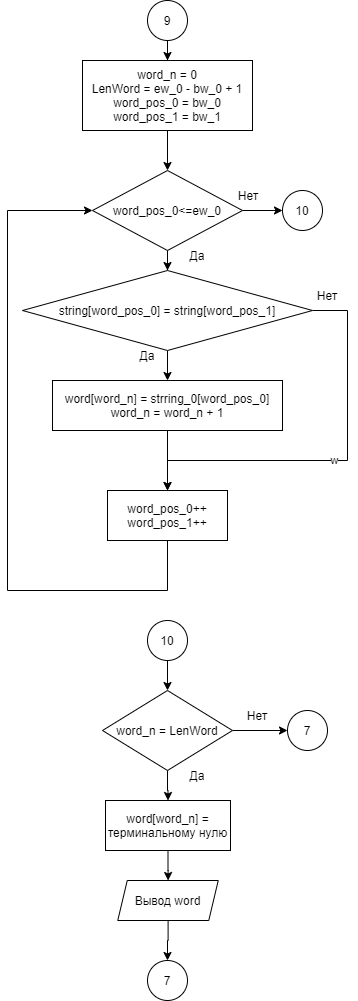
Алгоритм должен быть параметризован; обмен данными с подпрограммой должен осуществляться только через параметры; каждый из наборов исходных данных хранится в отдельном файле.

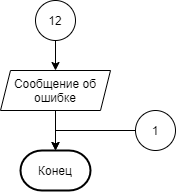
**Блок-схема.** 











**Псевдокод.**

НАЧАЛО

Целочисленный Массив LenString из 2-х элементов (0,0)

ЦЕЛОЧИСЛЕННЫЕ:

bw\_0 = 0

bw\_1 = 0

ew\_0 = 0

ew\_1 = 0

LenWord = 0

pos\_0 = 0

pos\_1 = 0

word\_pos\_0

word\_pos\_1

word\_n = 0

NewStringPos = 0

i = 0

СИМВОЛЫ:

temp = пробел

УКАЗАТЕЛИ НА СИМВОЛЬНЫЕ МАССИВЫ

word

string\_0

string\_1

NewString

МАССИВ СМВОЛОВ:

file\_name

ОТКРЫТЬ ФАЙЛ file\_name

ЕСЛИ файл НЕ ОТКРЫТ

ТО

| ВЫВОД сообщения об ошибке

| ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

КОНЕЦ

СЧИТАТЬ символ с файла file\_name в temp

ЕСЛИ в temp не записались символы

ТО

| ВЫВОД сообщения об ошибке

| ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

КОНЕЦ

ВЕРНУТЬСЯ в начало файла file\_name

ПОКА в файле file\_name НЕ ДОСТИГНУТ КОНЕЦ

| СЧИТАТЬ символ с файла file\_name в temp

| LenString[i] = LenString[i] + 1

|

| ЕСЛИ temp = концу строки И i = 1

| ТО

| | ВЫЙТИ ИЗ ЦИКЛА

| КОНЕЦ

|

| ЕСЛИ temp = концу строки И i = 0

| ТО

| | i = 1

| КОНЕЦ

КОНЕЦ

ЕСЛИ LenString[0] < 2 И LenString[1] < 2

ТО

| ВЫВОД сообщения об ошибке

| ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

ИНАЧЕ

| ЕСЛИ LenString[0]<2

| ТО

| | ВЫВОД сообщения об ошибке

| | ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

| КОНЕЦ

|

| ЕСЛИ LenString[1]<2

| ТО

| | ВЫВОД сообщения об ошибке

| | ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

| КОНЕЦ

КОНЕЦ

ВЕРНУТЬСЯ в начало файла file\_name

ЗАДАТЬ РАЗМЕР LenString[0] массиву string\_0

ЗАДАТЬ РАЗМЕР LenString[1] массиву string\_1

СЧИТАТЬ из файла file\_name LenString[0] символов в string\_0

СЧИТАТЬ из файла file\_name LenString[1] символов в string\_1

ПОКА pos\_0 < LenString[0] - 2

|

| pos\_1 = 0

| num\_word = 0

|

| ПОКА string\_0[pos\_0] = пробелу И pos\_0 < LenString[0]

| |pos\_0 = pos\_0 + 1

| КОНЕЦ

|

| ew\_0 = bw\_0 = pos\_0

|

| ПОКА string\_0[pos\_0] НЕ = пробелу И pos\_0 < LenString[0]

| |pos\_0 = pos\_0 + 1

| КОНЕЦ

|

| ew\_0 = pos\_0 - 1

|

|

| ПОКА pos\_1 < LenString[1] - 2

| |

| | num\_word = num\_word + 1

| |

| | ПОКА string\_1[pos\_1] = пробелу И pos\_1 < LenString[1]

| | |pos\_1 = pos\_1 + 1

| | КОНЕЦ

| |

| | ew\_1 = bw\_1 = pos\_1

| |

| | ПОКА string\_1[pos\_1] НЕ = пробелу И pos\_1 < LenString[1]

| | |pos\_1 = pos\_1 + 1

| | КОНЕЦ

| |

| | ew\_1 = pos\_1 - 1

| |

| | ЕСЛИ ew\_0 - bw\_0 = ew\_1 - bw\_1

| | ТО

| | | word\_n = 0

| | | LenWord = ew\_0 - bw\_0 + 1

| | |

| | | ЗАДАТЬ размер LenWord + 1 массиву word

| | |

| | | word\_pos\_0 = bw\_0

| | | word\_pos\_1 = bw\_1

| | |

| | | ПОКА word\_pos\_0 <= ew\_0

| | | |

| | | | ЕСЛИ string\_0[word\_pos\_0] = string\_1[word\_pos\_1]

| | | | ТО

| | | | | word[word\_n] = string\_0[word\_pos\_0]

| | | | | word\_n = word\_n + 1

| | | | КОНЕЦ

| | | |

| | | | word\_pos\_0 = word\_pos\_0 + 1

| | | | word\_pos\_1 = word\_pos\_1 + 1

| | | КОНЕЦ

| | |

| | | ЕСЛИ word\_n = LenWord

| | | ТО

| | | | word[word\_n] = терминальному нулю

| | | | ВЫВЕСТИ в консоль word

| | | КОНЕЦ

| | КОНЕЦ

| КОНЕЦ

КОНЕЦ

ЕСЛИ LenString[0] = LenString[1] и кратны 4

ТО

| ЗАДАТЬ размер LenString[0]\*2 - 1 массиву NewString

| pos\_0 = 0

| pos\_1 = 0

|

| ПОКА NewStringPos < LenString[0]\*2

| |

| | ДЛЯ i = 0 ПОКА i < 4 ДЕЛАТЬ i = i + 1

| | | NewString[NewStringPos] = string[pos\_0]

| | | NewStringPos = NewStringPos + 1

| | | pos\_0 = pos\_0 + 1

| | КОНЕЦ

| |

| | ДЛЯ i = 0 ПОКА i < 4 ДЕЛАТЬ i = i + 1

| | | NewString[NewStringPos] = string[pos\_0]

| | | NewStringPos = NewStringPos + 1

| | | pos\_0 = pos\_0 + 1

| | КОНЕЦ

| КОНЕЦ

| NewString[NewStringPos - 1] = терминальному нулю

| ВЫВЕСТИ в консоль NewString

|

ИНАЧЕ

| ВЫВЕСТИ в консоль сообщение об ошибке

КОНЕЦ

КОНЕЦ ПРОГРАММЫ

**Код Программы.**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Курс ИНФОРМАТИКА \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Project type :Win64 Console Application \*

\*Project name :Lab1 (2 sem) \*

\*File name :Лабораторная работа.exe \*

\*Language :CPP, MSVS 2021 \*

\*Programmers :Осминнов Никита Михайлович, Пысларь Александр Игоревич, М3О-110Б-21 \*

\*Modified By : \*

\*Created :18.03.2022 \*

\*Last revision:01.04.2022 \*

\*Comment :Двумерные массивы \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

    setlocale(LC\_ALL, "ru");

    int LenString[2] = {0,0};

    int bw\_0 = 0; //Начало слова из 1 строки

    int bw\_1 = 0; //Начало слова из 2 строки

    int ew\_0 = 0; //Конец слова из 1 строки

    int ew\_1 = 0; //Конец слова из 2 строки

    int LenWord = 0; //Длина слова

    int pos\_0 = 0;   //Позиция в 1 строке

    int pos\_1 = 0;   //Позиция во 2 строке

    int word\_pos\_0;  //Позиция внутри слова в 1 строке

    int word\_pos\_1;  //Позиция внутри слова во 2 строке

    int word\_n = 0;  //Номер следующего элемента слова (куда записывать символ)

    int num\_word = 0;   //Количество слов во 2 строке (Использовано для отладки)

    int NewStringPos = 0;  //Позиция в обьединенной строке

    char temp = ' '; //Временная переменная (Нужна для проверок)

    char\* word;      //указатель на массив со словом

    char\* string\_0;  //указатель на массив с 1 строкой

    char\* string\_1;  //указатель на массив со 2 строкой

    char\* NewString; //указатель на массив с обьединенной строкой

    //Имя открываемого файла

    const char file\_name[] = "correct\_01.txt";

    short i = 0;    //Просто переключатель для массива с размерами строк.

    ifstream ifs(file\_name);

    if (!ifs.is\_open())  //Проверка открытия файла

    {

        cout << "Ошибка открытия файла\n";

        exit(1);

    }

    ifs >> temp;

    if (temp == ' ')   // если в temp ничего не записоалось, то файл пуст

    {

        cout << "Файл пуст.\n";

        exit(3);

    }

    ifs.clear();

    ifs.seekg(0);

    while (!ifs.eof())  //Считаем количество символов в первых двух строках.

     {

        ifs.get(temp);      //Считываем данные

        LenString[i]++;     //Счетчик

        if ((temp == '\n') && (i == 1)) //Наткнулись на конец второй строки - завершаем подсчет

        {

            break;

        }

        if ((temp == '\n')&&(i == 0)) //Наткнулись на конец первой строки - начинаем считать длину второй.

        {

            i = 1;

        }

    }

    if ((LenString[0] < 2) && (LenString[1] < 2)) //Проверяем размеры строк

    {

        cout << "Обе строки пусты.\n";

        exit(2);

    }

    else {

        if ((LenString[0] < 2)) {

            cout << "Первая строка пуста.\n";

            exit(2);

        }

        if ((LenString[1] < 2)) {

            cout << "Вторая строка пуста.\n";

            exit(2);

        }

    }

    ifs.clear();

    ifs.seekg(0);

    string\_0 = new char[LenString[0]];    //Задаем размеры массивов, в которых будут храниться строки

    string\_1 = new char[LenString[1]];

    ifs.getline(string\_0, LenString[0]);                //Считываем строки

    ifs.getline(string\_1, LenString[1]);

    cout << "Первая строка:\n" << string\_0              //Вывод строк

        << "\nВторая строка:\n" << string\_1 << endl;

    while (pos\_0 < LenString[0] - 2)   //Начинаем поиск и сравнение слов

    {

        pos\_1 = 0;                     //Обнуляем счетчики

        num\_word = 0;

        while ((string\_0[pos\_0] == ' ') && (pos\_0 < LenString[0] - 1)) //Пропуск пробелов

        {

            pos\_0++;        }

        ew\_0 = bw\_0 = pos\_0; //Первый символ, отличный от пробела - начало слова.

        while ((string\_0[pos\_0] != ' ') && (pos\_0 < LenString[0] - 1)) // "Проход" по слову

        {

            pos\_0++;

        }

        ew\_0 = pos\_0 - 1;    //Раз наткнулись на пробел либо на конец строки, слово закончилось.

        while (pos\_1 < LenString[1] - 2) //Начинаем посик второго слова и сравнение

        {

            num\_word++;

            while ((string\_1[pos\_1] == ' ') && (pos\_1 < LenString[1] - 1)) //Снова пропуск пробелов

            {

                pos\_1++;

            }

            ew\_1 = bw\_1 = pos\_1;

            while ((string\_1[pos\_1] != ' ') && (pos\_1 < LenString[1] - 1)) //И нахождение конца слова

            {

                pos\_1++;

            }

            ew\_1 = pos\_1 - 1; //Снова корректировочка

            if ((ew\_0 - bw\_0) == (ew\_1 - bw\_1)) //Если длины найденных слов равны, то начинаем сравнение

            {

                word\_n = 0;    //Обнуляем счетчик

                LenWord = ew\_0 - bw\_0 + 1; //И считаем размер слова

                word = new char[LenWord + 1]; //Далее задаем размер массиву, в котором будет слово

                word\_pos\_0 = bw\_0;            //Копируем данные в подпрограмму для того, что бы не вызывать сдвиг

                word\_pos\_1 = bw\_1;

                cout <<"bw\_0 = " << bw\_0 << endl <<"ew\_0 = " << ew\_0 << endl;

                while (word\_pos\_0 <= ew\_0) // И начинаем сравнение

                {

                    if (string\_0[word\_pos\_0] == string\_1[word\_pos\_1])//Срабатывает только когда символы равны

                    {

                        word[word\_n] = string\_0[word\_pos\_0];

                        word\_n++;

                    }

                    word\_pos\_0++; word\_pos\_1++; //Наши счетчики позиций в строках

                }

                if (word\_n == LenWord) //Если длина полученного слова совпала с ожидаемой - выводим.

                {

                    word[word\_n] = '\0';

                    cout << "word = " << word << endl;

                }

            }

        }

    }

    if ((LenString[0] == LenString[1]) && !((LenString[0] - 1) % 4)) //Проверяем размеры строк на критерии задания

    {

        NewString = new char[(LenString[0] \* 2) - 1]; // Задаем размер массива с результатом

        pos\_0 = 0; //Идем с начало строк.

        pos\_1 = 0;

        while (NewStringPos < (LenString[0]\*2)) //Идем пока не кончатся строки

        {

            for (int i = 0; i < 4; i++) //Вставляем по 4 символа за раз

            {

                NewString[NewStringPos] = string\_0[pos\_0];

                NewStringPos++; pos\_0++;

            }

            for (int i = 0; i < 4; i++) {

                NewString[NewStringPos] = string\_1[pos\_1];

                NewStringPos++; pos\_1++;

            }

        }

        NewString[NewStringPos - 1] = '\0'; //Терминальный ноль ставим

        cout <<"Новая строка:\n" << NewString << endl; //И выводим в консоль

    }

    else {

        cout << "Размеры массивов не соответствуют условию." << endl;

    }

}

**Тесты программы.**

* + - 1. **Некорректные тесты**

**1.1**  Цель: проверить работоспособность программы при отсутствии файлов.

Ожидаемый результат: Ошибка открытия файла

Полученный результат:



1.2 Цель: проверить работоспособность программы при пустых файлах.

Ожидаемый результат: Файл пуст.

Полученный результат:



1.3 Цель: проверить работоспособность программы при отсутствии первой строки

Ожидаемый результат: Отсутствует строка

Полученный результат:



1.4 Цель: проверить работоспособность программы при отсутствии второй строки

Ожидаемый результат: Отсутствует строка

Полученный результат:



* + - 1. **Некорректные тесты**

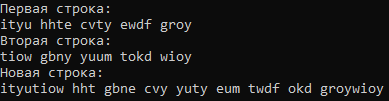
2.1 Цель: проверить работоспособность программы при отсутствии одинаковых слов и кратных 4-м массивов

Исходные данные:

ityu hhte cvty ewdf groy

tiow gbny yuum tokd wioy

Ожидаемый результат:



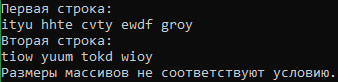
2.2 Цель: проверить работоспособность программы при отсутствии одинаковых слов и не кратных 4-м массивов

Исходные данные:

ityu hhte cvty ewdf groy

tiow yuum tokd wioy

Ожидаемый результат:



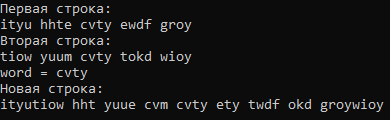
2.3 Цель: проверить работоспособность программы при наличии одинаковых слов и кратных 4-м массивов

Исходные данные:

ityu hhte cvty ewdf groy

tiow yuum cvty tokd wioy

Ожидаемый результат:



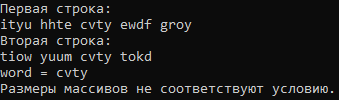
2.4 Цель: проверить работоспособность программы при наличии одинаковых слов и не кратных 4-м массивов

Исходные данные:

ityu hhte cvty ewdf groy

tiow yuum cvty tokd

Ожидаемый результат:



**Вывод по работе.**

Разработка программы завершена на том основании, что:

1. Полученные результаты совпали с ожидаемыми.

2. считаем набор тестов полным.