**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР**

**ОТЧЁТ**

**По лабораторной работе №3**

**По дисциплине «Программирование»**

Студент гр. 3351 \_\_\_\_\_\_\_\_\_\_\_\_ Морозов А.А.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_ Рыжов Н.Г.

Санкт-Петербург

2024

цель работы

Составить программу, реализующую функцию текстового редактора по форматированию текста абзаца по ширине:



Текст абзаца вводится по желанию пользователя с клавиатуры или из файла. Абзац может состоять из нескольких строк. Каждая строка состоит из произвольного количества слов, разделенных пробелами. Если абзац вводится с клавиатуры, то строка заканчивается нажатием клавиши Enter. Абзац считается полностью введенным, если в конце строки стоит точка.

Написать программу форматирования абзаца, позволяющую изменять ширину текста (число знаков ширины абзаца вводится с клавиатуры) и выравнивать текст по ширине (одновременно по левой и правой границе). При изменении ширины абзаца слова должны переноситься из последующих или предыдущих строк так, чтобы в каждой строке помещалось максимальное количество слов. Полученный текст вывести в файл и на экран.

**Текст программы**

#include <iostream>

#include <fstream>

#include <cstring>

#include <cstdlib>

#include <cctype>

using namespace std;

// Функция для форматирования абзаца по ширине

// Принимает:

// text - строка, содержащая текст абзаца

// width - максимальная ширина строки

// fout - поток вывода, куда будет записываться результат форматирования

void formatParagraph(char\* text, int width, ofstream& fout) {

    // Длина строки

    int len = strlen(text);

    // текущая позиция в строке

    int pos = 0;

    // текущая ширина строки

    int linewidth = 0;

    // начало текущего слова

    int startword = 0;

    // конец текущего слова

    int endword = 0;

    // количество пробелов между словами

    int numspaces = 0;

    // обработка первой строки

    while (isspace(text[pos])) {

        // пропуск начальных пробелов

        pos++;

    }

    // начало текущего слова установлено на текущую позицию

    startword = pos;

    // нахождение конца текущего слова

    while (pos < len && !isspace(text[pos])) {

        pos++;

    }

    // конец текущего слова установлен на текущую позицию

    endword = pos;

    // нахождение количества пробелов между текущим и предыдущим словами

    numspaces = 0;

    while (pos < len && isspace(text[pos])) {

        numspaces++;

        pos++;

    }

    // вcrement ширины текущей строки на количество символов в текущем слове + количество пробелов

    linewidth += endword - startword + numspaces;

    // запись текущего слова в выходной поток

    fout.write(text, endword - startword);

    // добавление пробелов в выходной поток

    while (numspaces > 0) {

        fout << ' ';

        numspaces--;

    }

    // обработка оставшихся строк

    while (pos < len) {

        // начало следующего слова установлено на текущую позицию

        startword = pos;

        // нахождение конца следующего слова

        while (pos < len && !isspace(text[pos])) {

            pos++;

        }

        // конец следующего слова установлен на текущую позицию

        endword = pos;

        // нахождение количества пробелов между текущим и предыдущим словами

        numspaces = 0;

        while (pos < len && isspace(text[pos])) {

            numspaces++;

            pos++;

        }

        // если ширина текущей строки + длина следующего слова + количество пробелов больше ширины строки, то перенос слова на новую строку

        if (linewidth + endword - startword + numspaces > width) {

            // перенос слова на новую строку

            fout << endl;

            // обновление ширины текущей строки

            linewidth = 0;

        }

        else {

            // добавление пробелов в текущую строку

            int spacestoadd = width - linewidth;

            // если количество пробелов больше количества пробелов между текущим и предыдущим словами,

            // то использовать количество пробелов между текущим и предыдущим словами

            if (spacestoadd > numspaces) {

                spacestoadd = numspaces;

            }

            while (spacestoadd > 0) {

                fout << ' ';

                spacestoadd--;

            }

            // обновление ширины текущей строки

            linewidth += endword - startword + numspaces;

        }

        // запись следующего слова в выходной поток

        fout.write(&text[startword], endword - startword);

    }

    // выравнивание последней строки

    int lastlinewidth = linewidth;

    while (lastlinewidth++ < width) {

        fout << ' ';

    }

    fout << endl;

}

int main() {

    setlocale(LC\_ALL, "RU");

    int inputType;

    cout << "Выберите способ ввода данных:" << endl;

    cout << "1. Из файла" << endl;

    cout << "2. Из консоли" << endl;

    cin >> inputType;

    char text[100000];

    if (inputType == 1) {

        char filename[100];

        cout << "Введите имя входного файла: ";

        cin >> filename;

        // Открытие входного файла

        ifstream fin(filename);

        // Проверка открытия файла

        if (!fin.is\_open()) {

            cerr << "Ошибка открытия входного файла." << endl;

            // Завершение программы с кодом ошибки

            return 1;

        }

        // Создание массива для хранения текста из файла

        char text[100000];

        // Считывание символа из файла и запись его в массив

        char ch;

        int i = 0;

        while (fin.get(ch)) {

            text[i++] = ch;

        }

        // Запись символа конца строки в конец массива

        text[i] = '\0';

        // Закрытие файла

        fin.close();

    }

    else if (inputType == 2) {

        cout << "Введите текст: " << endl;

        cin.ignore();

        cin.getline(text, sizeof(text));

    }

    else {

        cerr << "Неверный выбор способа ввода данных." << endl;

        return 1;

    }

    // Ввод ширины абзаца

    int width;

    cout << "Введите ширину абзаца: ";

    cin >> width;

    // Открытие выходного файла

    ofstream fout("output.txt");

    // Проверка открытия файла

    if (!fout.is\_open()) {

        cerr << "Ошибка открытия выходного файла." << endl;

        // Завершение программы с кодом ошибки

        return 1;

    }

    // Форматирование абзаца

    formatParagraph(text, width, fout);

    // Закрытие выходного файла

    fout.close();

    cout << "Отформатированный текст записан в файл output. ";

    return 0;

}

**Описание программы**

Программное обеспечение – Microsoft Visual Studio 2022. Язык программирования – C++. Программа принимает на вход текст из файла или из консоли на выбор пользователя. После этого программа принимает значение ширины текста. Далее с помощью функции formatParagraph текст обрабатывается и форматируется по заданной ширине. Результат работы программы записывается в отдельный файл output.txt.

**Руководство пользователя**

Программа просит пользователя выбрать вариант ввода текста: из файла или из консоли. При выборе ввода текста из файла пользователю необходимо ввести название входного файла. Если выбрать вариант из консоли, то текст можно будет ввести прямо там же.

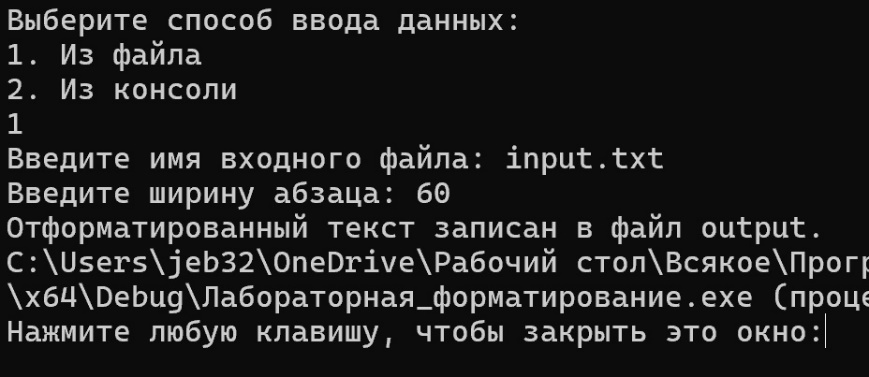


Рис. 1 – Ввод текста из файла input.txt

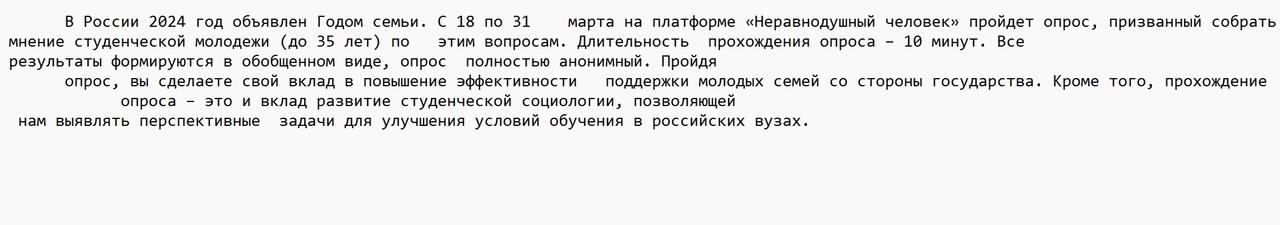


Рис. 2 – Содержимое файла input.txt

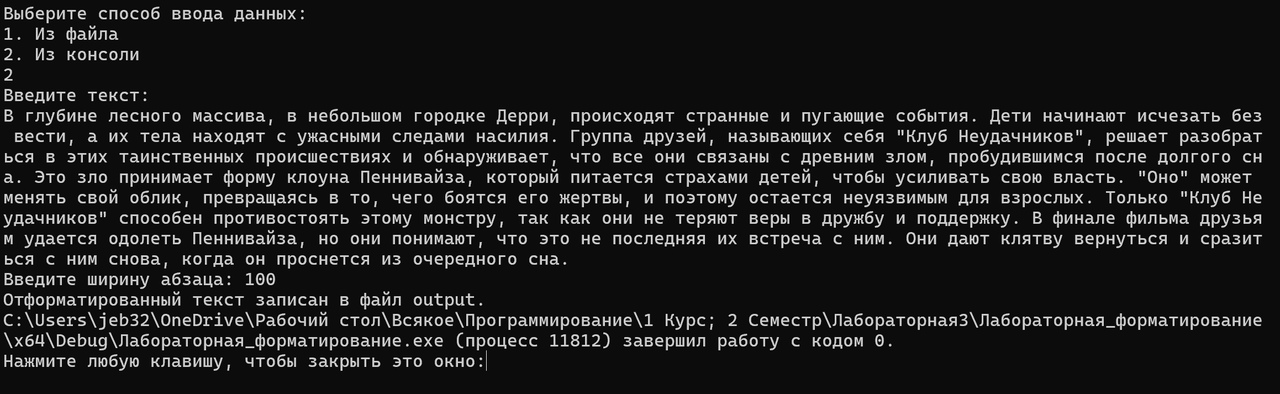


Рис. 3 – Ввод текста с помощью консоли

После ввода текста программа с помощью функции formatParagraph форматирует данные. Обработка первой строки: программа пропускает первые пробелы, сохраняя отступ абзаца; происходит перебор слов и выставление пробела после каждого. Обработка оставшихся строк, кроме последней, происходит аналогично. Также добавляются дополнительные пробелы между словами, чтобы текст был выровнен по ширине. Последняя строка не выравнивается по ширине, но в ней также удаляются лишние пробелы между словами.

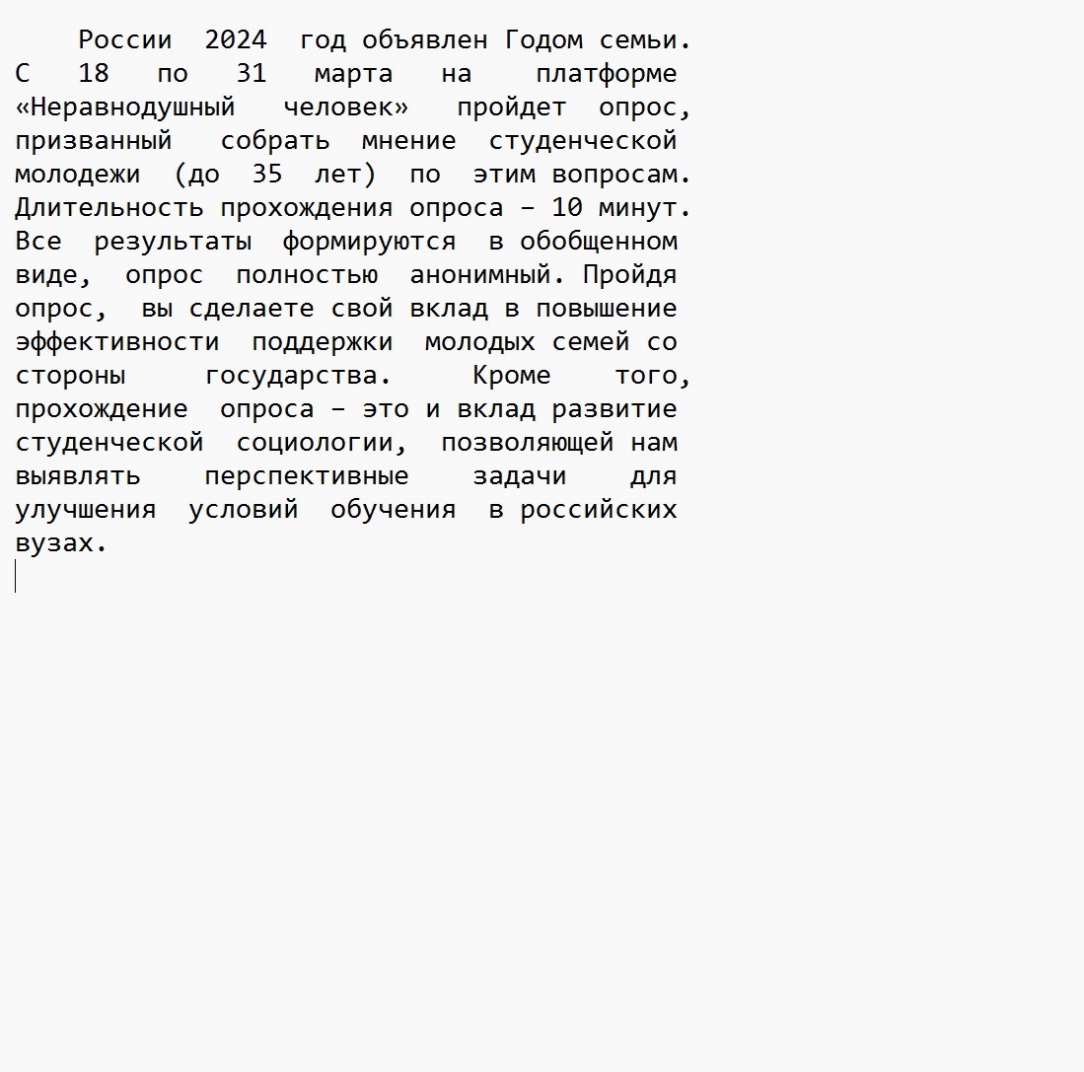


Рис. 4 – Выравнивание текста из файла input.txt

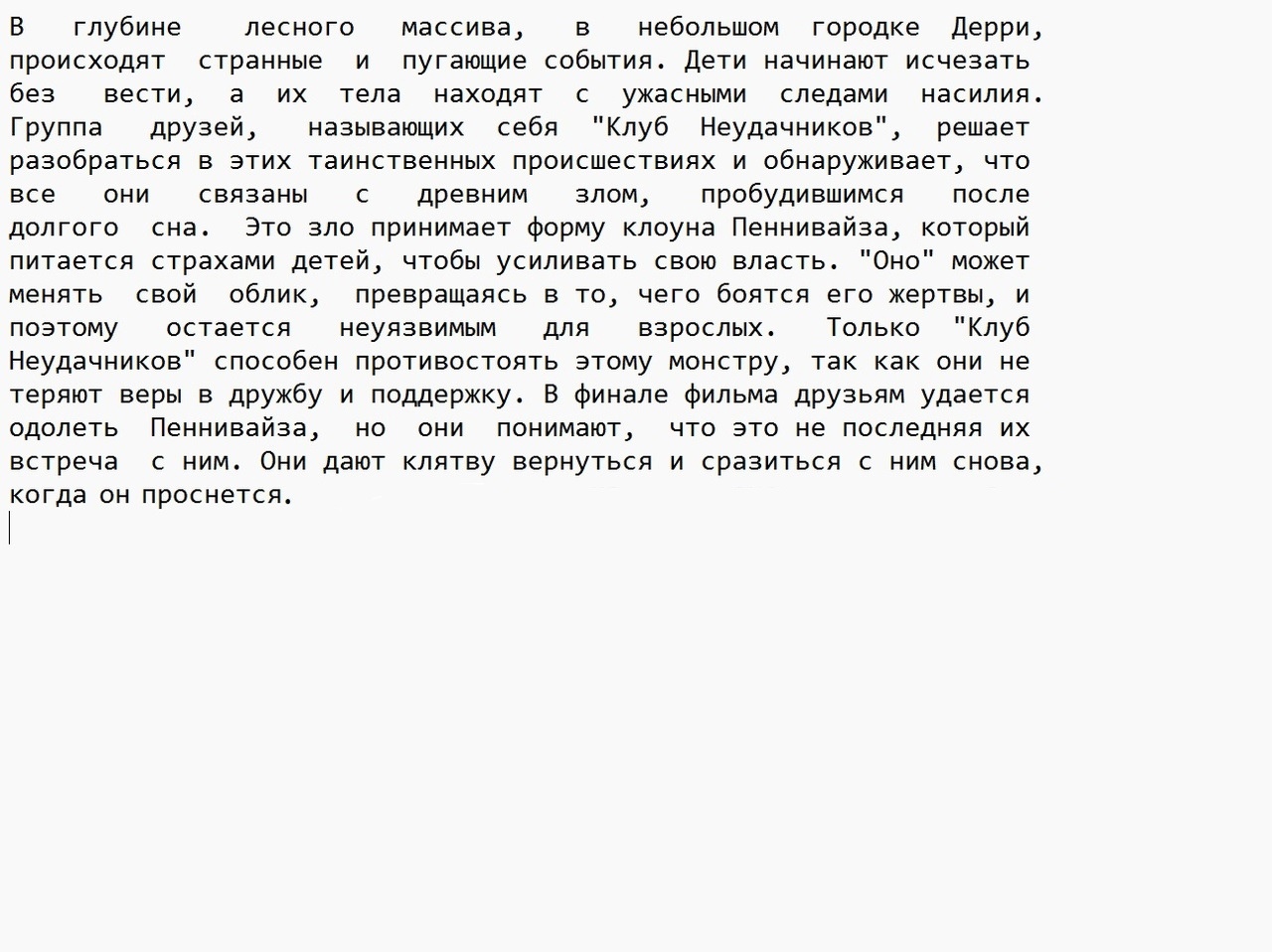


Рис. 5 – Выравнивание текста из консоли

В любом из вариантов ввода текста программа сохраняет результат своей работы в отдельный файл output.txt, который находится в файле программы.

**Пути дальнейшего улучшения программы**

1. Улучшение визуальной части программы
2. Добавление новых вариантов выравнивания текста: по левому краю, по правому краю, по середине