МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №11**

по дисциплине: Основы программирования

тема: «Обработка типизированных файлов»

Выполнил: ст. группы ПВ-201

Машуров Дмитрий Русланович

Проверил:

Притчин Иван Сергеевич

Белгород 2020 г.

**Лабораторная работа №11**

**«Обработка типизированных файлов»**

**Цель работы:** получение навыков работы с файлами прямого доступа

**Задания для подготовки к работе:**

1. Изучите организацию работы с типизированными файлами и возможности Турбо Паскаля для организации прямого доступа к записям файла.
2. Разработайте алгоритм и составьте программу для решения задачи соответствующего варианта с использованием прямого доступа к записям файлов. Во всех случаях, если возможно, не используйте вспомогательные файлы.
3. Разбейте задачу на подзадачи. Если задача небольшая, выделите самостоятельной подзадачей обработку файла.
4. Опишите блок-схему алгоритма решения задачи в укрупненных блоках.
5. Для каждой подзадачи опишите используемые структуры данных, если необходимо, и спецификацию, и блок-схему алгоритма.
6. Опишите блок-схему алгоритма решения задачи с использованием блоков «предопределенный процесс».
7. Опишите процедуру для вывода содержимого файла на экран.
8. В программе обработки файла выведите содержимое файла на экран до и после обработки файла
9. Закодируйте алгоритм.
10. Файлы для обработки создайте с помощью отдельной программы.
11. Подберите наборы тестовых данных с обоснованием их выбора.

**Задания к работе:**

1. Наберите программу, отладьте ее, протестируйте.
2. Выполните анализ ошибок, выявленных при отладке программы

**Задание варианта №17:**

Дан символьный файл. Сократить число пробелов между словами до одного.

**Выполнение работы:**

1. **Выделение подзадач**

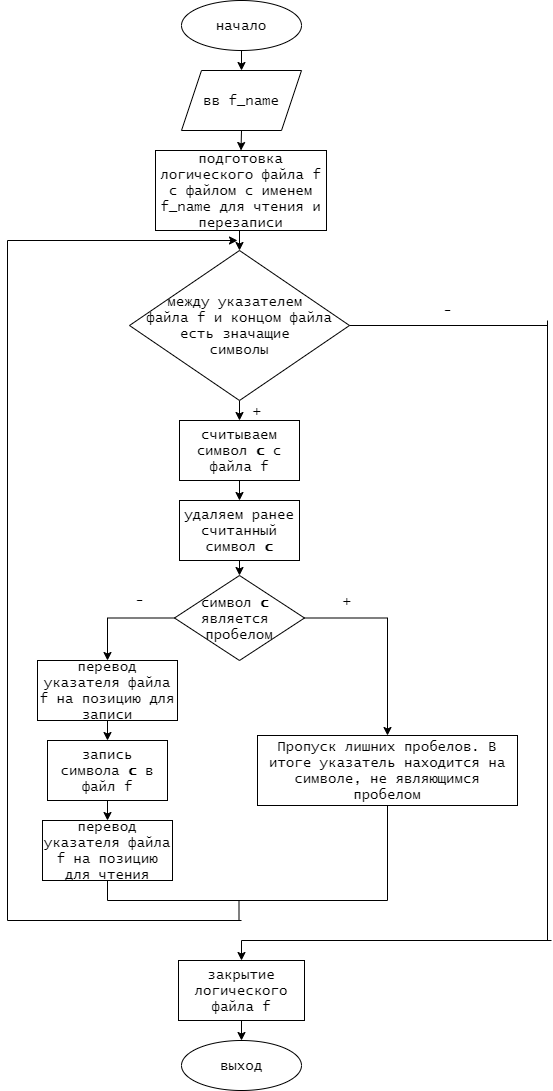
Суть алгоритма заключается в том, что мы считываем по одному байту, и, если этот байт нужно оставить в файле, то переводим указатель с позиции чтения файла на позицию записи (которая будет расположена ближе к началу файла), записываем байт, затем переводим указатель на позицию для чтения и продолжаем считывание.

Выделим следующие подзадачи:

* Пропуск лишних пробелов

Дальнейшее описание алгоритма даётся в блок схеме

1. **Блок-схема алгоритма в укрупнённых блоках**

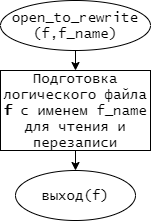
****

1. **Описание подпрограмм**

Спецификация процедуры open\_to\_rewrite

1. Заголовок: procedure open\_to\_rewrite(var f: file of char; f\_name: string)
2. Назначение: подготавливает логический файл f с именем f\_name к чтению и перезаписи
3. Входные данные: f, f\_name
4. Выходные данные: f

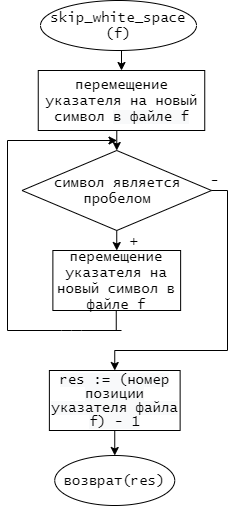
Блок-схема:



Спецификация функции skip\_white\_space

1. Заголовок: functionskip\_white\_space(f: file of char) : integer
2. Назначение: возвращает номер позиции с которого начинается значащий символ в файле f
3. Входные параметры: f
4. Выходные параметры: нет

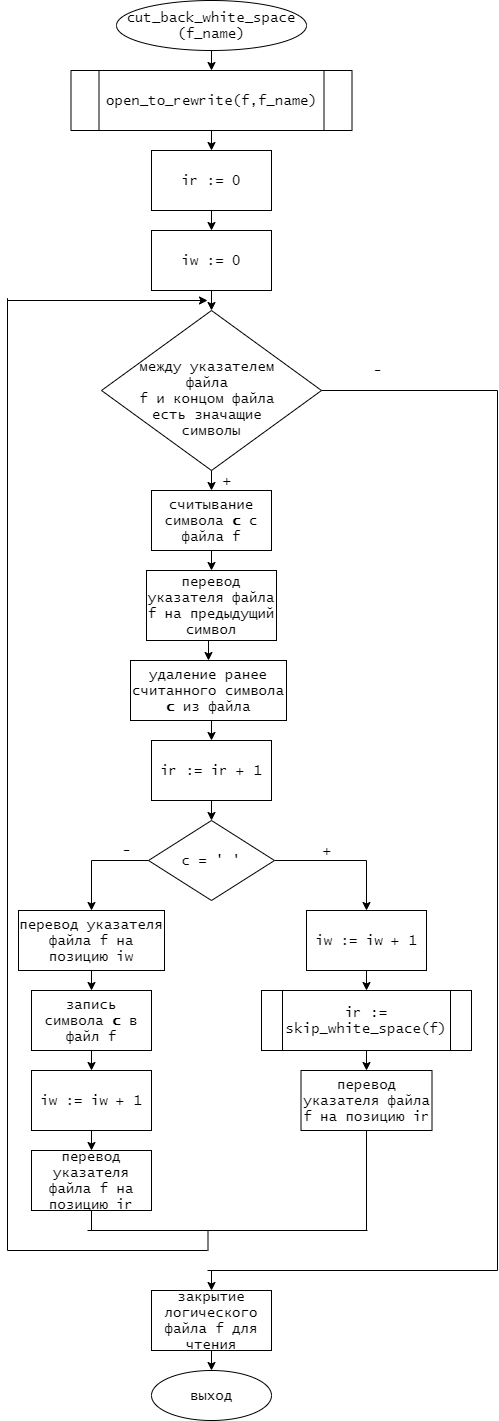
Блок-схема:



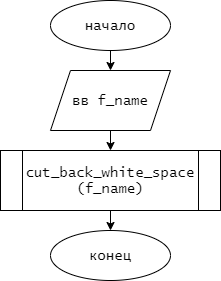
Спецификация процедуры cut\_back\_white\_space

1. Заголовок: cut\_back\_white\_space(f\_name: string)
2. Назначение: создаёт логический файл f с именем f\_name. Затем сокращает число пробелов между символами в файле f до одного. После закрывает логический файл f
3. Входные параметры: f\_name
4. Выходные параметры: нет

Блок-схема:



1. **Блок-схема с блоками предопределённый процесс**

****

1. **Тестовые данные**

|  |  |  |
| --- | --- | --- |
| № | Вход | Выход |
| 1 |  |  |
| 2 | C:\Users\Shmouk\Desktop\рвх2.PNG | C:\Users\Shmouk\Desktop\рвх2.PNG |
| 3 | C:\Users\Shmouk\Desktop\рвх3.PNG | C:\Users\Shmouk\Desktop\рвы3.PNG |

1. **Текст программы**

//Дан символьный файл. Сократить число пробелов между словами до одного.

**procedure** open\_to\_rewrite(**var** f: **file of** char; f\_name: string);

**begin**

assign(f,f\_name);

reset(f);

**end**;

**function** skip\_white\_space(f: **file of** char) : integer;

**var** c: char;

res: integer;

**begin**

Read(f,c);

**while** (c = ' ') **and not** Eof(f) **do**

**begin**

Read(f,c);

**end**;

res := FilePos(f) - 1;

skip\_white\_space := res;

**end**;

**procedure** cut\_back\_white\_space(f\_name: string);

**var** f: **file of** char;

c: char;

ir, iw: integer;

**begin**

open\_to\_rewrite(f,f\_name);

ir := 0;

iw := 0;

**while not** Eof(f) **do**

**begin**

read(f,c);

seek(f,FilePos(f)-1);

write(f,' ');

ir := ir + 1;

**if** (c = ' ') **then**

**begin**

iw := iw + 1;

ir := skip\_white\_space(f);

seek(f,ir);

**end**

**else**

**begin**

seek(f,iw);

write(f,c);

iw := iw + 1;

seek(f,ir);

**end**;

**end**;

close(f);

**end**;

**var** f\_name: string;

**begin**

writeln('Введите имя файла');

read(f\_name);

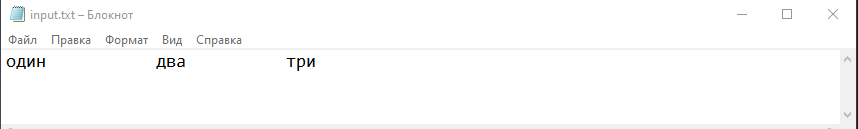
cut\_back\_white\_space(f\_name);

**end**.

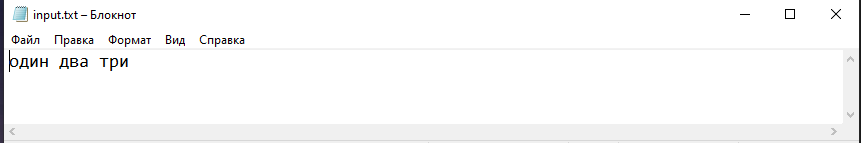
1. **Результаты работы программы**

*Пример 1:*

*Вход:*

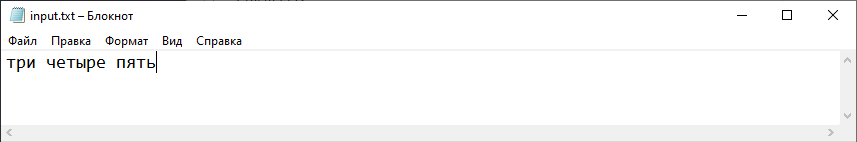


*Выход:*

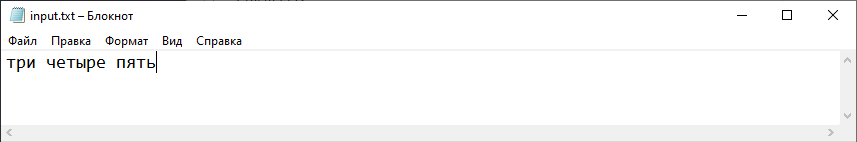


*Пример 2:*

*Вход:*



*Выход:*

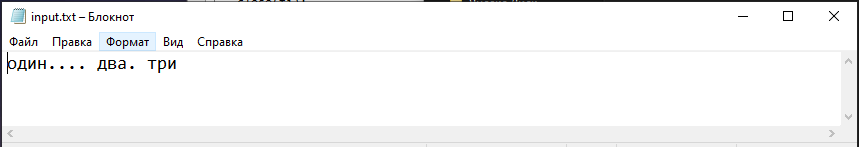


*Пример 3:*

*Вход:*



*Выход:*



1. **Анализ совершённых ошибок**

* Упущена ; после вызова процедуры cut\_back\_white\_space